universität
wien

MASTERARBEIT / MASTER’S THESIS

Titel der Masterarbeit / Title of the Master’s Thesis

„Development of a water analysis support model and system in cooperation with the Environment Agency Austria“

verfasst von / submitted by

Ivan Igic, BSc (WU)

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien, 2023 / Vienna, 2023

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	UA 066 926
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Masterstudium Wirtschaftsinformatik UG2002
Betreut von / Supervisor:	Univ.-Prof. Dipl.-Ing. Dr. Dr. Gerald Quirchmayr

Acknowledgements

I would like to express my deepest appreciation to Univ.-Prof. Dipl.-Ing. Dr. Dr. Gerald Quirchmayr, without his guidance and support the completion of this master's thesis would not have been possible. The invaluable patience and feedback that the professor gave me, were indispensable for writing this master's thesis.
I am also grateful to my colleagues at the Environment Agency Austria for providing me generously with knowledge and professional expertise. First and foremost, I would like to thank Mag. Baker Abu-Naim for coming up with the idea to write my master's thesis on the topic of water analysis. Additionally, this journey would also not be possible without the expert advice and support of Mag. Gerald Hochedlinger. Thanks for providing me with in-depth knowledge about water analysis and helping me to define the specific topic of the master's thesis. I am also thankful to Mag. Harald Loishandl-Weisz, who together with Gerald Hochedlinger gave me constant feedback on the support model design and prototype implementation. From a technical point of view, I would like to thank Michael Schalk for introducing and describing me the technical aspects of the H2O Expert database. Special thanks, should also go to Denis Masinovic for advising me with conceptual concerns of the prototype design and implementation and constantly helping me with concrete programming problems. The technical advice you were able to give me were irreplaceable for the successful implementation of the prototype.
Lastly, I would like to thank my family, my friends and especially my partner. Their emotional support and belief in me, gave me the strength to keep going. My partner always supported me when things didn't go as planned and permanently motivated me to finally finish this master's thesis.

Abstract

Austria is one of Europe's most water-wealthy countries with an amount of annual precipitation to 92 km^3 of water. The protection and sustainable use of water as a resource is one of the most important environmental protection goals of our time. To keep up the water quality, all rivers, lakes, and ground waters of Austria must be monitored at regular intervals. These professional and administrative implementation of monitoring programs is performed by the Austrian Federal Ministry for Climate Action, Environment, Energy, Mobility, Innovation and Technology. The Environment Agency Austria on the other hand, is responsible for data management, provision of data to the public and the development of query and evaluation tools. The goal of this master's thesis is to develop a support model for the withdrawal of water samples and implementing a prototype. This prototype should support the person who collects the water sample by providing the validity of the collected on-site parameters, checked based on measured data from selected sample, using statistical methods and thus identify certain significant deviations. Additionally, in the case of deviating data the system should give the sampler a recommendation provided by a rule-based system (Prolog) and thereby advise him how to proceed. Finally, this master's thesis is an attempt to ensure data quality and prevent erroneous data collection at an early stage and thus support the sampler in his local investigations, so that he can find the real cause for the deviation.

Keywords: Water analysis, Support model, Knowledge-based system, Prototype, Web app

Kurzfassung

Mit einer jährlichen Niederschlagsmenge von 92 km^3 Wasser ist Österreich eines der wasserreichsten Länder Europas. Der Schutz und die nachhaltige Nutzung der Ressource Wasser ist eines der wichtigsten Umweltschutzziele unserer Zeit. Um die Wasserqualität aufrechtzuerhalten, müssen alle Flüsse, Seen und Grundwässer Österreichs in regelmäßigen Abständen überwacht werden. Die fachliche und administrative Umsetzung der Überwachungsprogramme obliegt dem österreichischen Bundesministerium für Klimaschutz, Umwelt, Energie, Mobilität, Innovation und Technologie. Das Umweltbundesamt hingegen ist für das Datenmanagement, die Bereitstellung der Daten für die Öffentlichkeit und die Entwicklung von Abfrage- und Auswertungstools verantwortlich. Das Ziel dieser Masterarbeit ist die Entwicklung eines Unterstützungsmodells für die Entnahme von Wasserproben und die Implementierung eines Prototyps. Dieser Prototyp soll den/der Probenehmer/in, dabei unterstützen, die Gültigkeit der vor Ort gesammelten Parameter sicherzustellen. Dies passiert anhand der Überprüfung von historischen Messdaten der ausgewählten Probe mithilfe statistischer Methoden, um so signifikante Abweichungen zu identifizieren. Darüber hinaus soll das System dem Probenehmer bei abweichenden Daten eine Empfehlung eines regelbasierten Systems (Prolog) liefern und ihm so Hinweise zum weiteren Vorgehen geben. Abschließend stellt diese Masterarbeit den Versuch dar, die Datenqualität sicherzustellen und fehlerhafte Datenerhebungen frühzeitig zu verhindern und so den/der Probenehmer/in bei seinen/er Untersuchungen vor Ort zu unterstützen, damit er/sie in der Lage ist, die eigentliche Ursache für die Abweichung zu finden.

Schlagwörter: Wasseranalyse, Unterstützungsmodell, Wissensbasiertes System, Prototyp, Webanwendung

Contents

List of Figures

1. Introduction

Austria is one of Europe's most water-wealthy countries. The huge amount of annual precipitation corresponds to 92 km^3 of water. The protection and sustainable utilisation of the resource water for future generations is a prior environmental goal. To keep up the water quality, all rivers, lakes and ground waters of Austria must be monitored at regular intervals. The professional and administrative implementation of monitoring programs is performed by the Federal Ministry for Climate Action, Environment, Energy, Mobility, Innovation and Technology in close cooperation with the provincial authorities. [FMoAMc][FMoAMd][FMoAMa] The Environment Agency Austria is responsible for data management, provision of data to the public, development of query tools and evaluation tools and private civil engineers are in charge of performing water monitoring and water analysis. [Ausa]
The Agency has been dealing with the protection of water quality for over 30 years. They advise and support international institutions, national governments, regional and local authorities and companies. [Ausb] One key area of their expertise is also a technical support. Their IT department is specialised for the design and implementation of web-based information systems and the development of databases, data management and reporting. [Ausa] The Water Information System Austria - WISA is the central platform which provides access to data and information about the Austrian water industry. One central tool is the H2O Expert database, from this database water monitoring results of Austria (rivers, lakes, and groundwater) can be retrieved. [FMoAMb]
The digitisation and optimisation of the water sample taking process and data validation for ensuring data quality is the research question of this master's thesis.

1.1. The problem

For providing a professional water status monitoring an accurate analysing of water objects must be accomplished. Therefore, commissioned private civil engineers collect several times a year samples from water measuring points and send them to the laboratory, where the samples were analysed and added to the system after validation. The currently implemented water monitoring system is a classic desktop (web) application which is not mobile capable, several parts of the collecting process are therefore harder to accomplish. How does the sample collect process works in detail? First, the engineer must locate the water body and the water measuring point. After locating the right measuring point the engineer records chemical and physical field parameters - measured directly on site. Some parameters are being measured by a measuring instrument (e.g., temperature, oxygen level) and other parameters are subjective estimations like smell or colour. These

measured parameters are recorded by hand in a report. Once the engineer returns from the field back to his office, the data is entered into the system. Finally, does the sample taker know if the recorded data is "fit for use"? [WS96, p. 6]
It can clearly be seen that this process can be optimised and digitised in order to facilitate the work of the engineers and to secure data quality. Studies from the past have shown that companies and their databases are not error-free and contain a surprisingly large number of errors. [Arn92] Data quality problems go beyond accuracy and include also other aspects like completeness and accessibility. Poor data quality can have substantial and economic impacts. To improve data quality, it must be understood what data quality means to the data consumer in our case the sample taker. [WS96, p. 6]
In order that, water analysis can be performed correctly and accurately, data quality must be ensured directly on the spot. The aim of this master's thesis is to develop a support model for the withdrawal of water samples and to implement a prototype. This prototype should be a platform independent web application and should support the sample taker to validate the taken sample data, and in this way secure data quality.

1.2. The solution

The planned measure is to support the sample taker with the withdrawal and analysis of the drawn water sample directly on the spot.
The support should take place in following scenario: Statistical checks by reference to the on-the-spot-measurement of the drawn sample. The plausibility checks, provide the validity of the currently collected on-site parameters checked on the basis of measured data from selected sample using statistical methods and thus identify certain significant deviations. The initial analysis takes place based on of existing data from former measurement periods (H2O Expert database). In addition to the statistical analysis, the system should give the sampler a recommendation provided by a rule-based system. With the recommendation and results of the analysis the sample taker can identify deviating measurement results such as outliers and detect errors, such false measuring spots or measurement mistakes. In the case of deviating data not caused through human error or defective equipment, it is conceivable that with the help of the system the engineer is also able to distinguish if implausible measuring data has its causes, due to other reasons caused by nature, humans or industry (e.g., local and seasonal conditions like pollution through heavy rains, flood, washouts etc.).
Finally, should these indications early identify data quality problems, avoid environmental pollution and support the sample taker with his local investigations.

1.3. Research Questions

The goal of the master's thesis is to answer the following three research questions:

Question 1 - How can a system based on a support model help the sample taker validate the measured data of the on-site water parameters: On the basis of the water data analysis results the user should identify if the data is sound and correct or if significant deviations are present. As a result, it should be determined if data quality can be ensured.

Question 2 - In case of deviation, how can a rule-based system support the sample taker in determining alternative actions and in finding the reason for the deviation: In case of a deviation pointing to a possible contamination the system shall advise the sampler how to proceed.

Question 3 - What are the limitations and strengths of the model: Through applying various statistical methods on the water data the explanatory power of the on-site parameters must be analysed, which are the foundation of the support model. Finally, the validation of the model-requirements will be executed through a scenario-based approach.

1.4. Challenges

This chapter describes the challenges which arise for answering the previously mentioned research question. Following issues must be explored.

1.4.1. Challenges at the methodological level

Plausibility checks: One big issue is it to investigate which statistical methods exists and which are appropriate for this specific problem. Currently used methods by the Environment Agency and others must be analysed. It is useful to go also beyond the scope of the problem and to check solutions of related problems, because there is always a chance to improve the current processes. The biggest challenge is here to find the optimal statistical method and to validate that the explanatory power of the measured "on spot parameters" are high enough? Moreover, is it possible and necessary to integrate more complex methods like regression analysis or machine learning methods?

Data Quality: As already mentioned in the introduction, securing data quality is a central part of the master's thesis. Information quality is a critical concern of organisations from Management Information Systems (MIS). The direct access of information from various sources by information users have increased the awareness for high quality information in organisation. Information quality research activities have significantly increased over the years and information quality has been rated in the industry as a top concern. [LSKW02, p. 133] Various information quality problems like contradiction of data, fully missing data or partially missing data etc. can have substantial social and economic impacts and must be prevented. [WS96, p. 6] Moreover, following scientific key principles for computing can be a way to design a system which secures aspects of data quality. Analysing this computing framework called "Great Principles framework" revealed that the categories computation, communication, recollection, evaluation and design are ideal for our purposes: [Den03][DM15]

Computation (meaning and limits of computing): means that computing systems are built from processing elements that transform and store information. An important computation principle is "Finite representations of real processes always contain errors". [DM15, p. 7] It must be clear that errors can always occur, if this is understood during the development process, errors can also be prevented.

Communication (reliable data transmission): means processing elements exchange information. A necessary principle for communication is "Data communication always takes place in a system consisting of a message source, an encoder, a channel, and a decoder". [DM15, p. 7] The challenge for the system is here to establish a reliable data transmission and to cover this mentioned components.

Recollection (storage and retrieval of information): same like computation it means also that computing systems are built from processing elements that transform and store information. An essential recollection principle is "Access to stored objects is controlled by dynamic bindings among names, handles, addresses, and locations". [DM15, p. 8] The challenge for the system is here to access and store data properly.

Evaluation (performance prediction and capacity planning): means that humans predict the speed and capacity of systems. An indispensable principle for evaluation is "The principal tools of evaluation are modelling, simulation, experiment, and statistical analysis of data". [DM15, p. 9]. Evaluation of the whole system is a fundamental part of this thesis, and therefore the development process includes evaluation tools like modelling, simulation and statistical analysis of data.

Design (building reliable computing systems): means humans decompose systems into processing elements and organize their construction. A relevant design principle is "Abstraction, information hiding, and decomposition are complementary aspects of modularity". [DM15, p. 9]. The art to design computing systems efficient is an important challenge, hence the support model is designed from scratch.

The categories are like windows into one computing knowledge space and should not be seen as separate pieces. [DM15, p. 6] Most computing technologies are containing principles from all categories, and many technologies have factors in common. [DM15, p. 6] The aim is to design the support system by following also this framework of principles, so that challenges can be overcame.

1.4.2. Technological challenges

Technologies of the system: Another issue is to find the best technological solution for the given problem. The most important requirements are that the application must be platform independent and that it must be able to cover all water analysis requirements. Moreover, a statistical method for the data analysis suited for the specific data must be found. Furthermore, it is also important to find a technology for implementing a rule-based system suitable for this prototype. Finally, it is also important to choose technologies which fit into the systems of the environment agency.

Integration of data sources: Another challenge is the integration of data sources. How is the data being delivered? In which format and which condition is the data? Must the data be transformed in a suitable format or even be corrected? What problems can occur when data is changed too much, are the results still accurate?

Integration of various technology: Finally, the combination of various frameworks (Prolog, Statistics etc.) or programming languages (Java, JavaScript, SQL) can also be a challenge as compatibility must be assured. If it is not possible to combine certain technologies other ways must be find. (External services)

1.5. Description of the working environment

The idea for implementing software for the water analysis field comes directly from employees of the IT department and water monitoring specialists of the water quality department. There is always a demand for digitisation within the Environment Agency Austria and the whole environment protection sector. Many environmental monitoring processes are already digitised or are being digitised as prescribed by Austrian law or directives of the European Union. The topic of this master's thesis is not influenced by political decisions. As already mentioned, the recording of the water sample data is currently done by hand, a digitisation in this area would be a great improvement.
I have been working in the Environment Agency Austria for 10 years as a software tester and I could acquire comprehensive know-how and bright understanding and knowledge in the environment protection field. The idea to develop a support model for the withdrawal process come out after detailed discussions with specialists from the water protection department. After understanding the problems of the sampling process it was possible to specify the topic of the master's thesis and to create a possible solution concept.
During solution development it is wanted to have a close cooperation with the specialists of the water protection department, because of the reason that their knowledge is substantial. Their know-how and deep understanding for this specific problem is an important guidance through the developing process. Therefore, it is desired to hold regular meetings and discussions to involve the water department in process design. This gathered feedback can be used to detect potential problems in design or implementation in an early stage or even prevent them.
As stated before, it is also important to choose technologies which fit into the systems of the Environment Agency Austria. Using technologies, the Environment Agency is using can be a benefit, as synergies can be used. Furthermore, it also important to have independence as too many dependencies to frameworks and systems of the agency can make the system too complex.

1.6. Structure of the work

The master's thesis is structured in following chapters and each chapter has a specific focus and objective:

1. Introduction: The topic of the master's thesis is introduced, especially the environment protection field, the Environment Agency Austria, and the withdrawal process. Furthermore, the research questions together with the challenges and working environment were presented.
2. State of the art: In this chapter a scientific literature analysis and practice review is carried out. In our case classic and modern water analysis methods and the description of the spot parameters. Finally, various mobile software were analysed too.
3. Research approach: This chapter describes the methods which were applied to solve the specific problem of the master's thesis. The spiral model for the development process, the process validation through model comparison and Knowledge Representation are great approaches to develop the prototype.
4. Model: The model development chapter discusses the design and development of the model in theory and gives a visual representation of the use-cases, processes and data through diagrams.
5. Prototype: The next chapter illustrates all technical aspects for implementing the model. The architecture of the prototype is described too.
6. Test and validation: The validation and tests are represented in this chapter by a scenario-based simulation. Finally, the results of the validation and the limitations of the prototype are discussed.
7. Discussion of results: In this chapter the results of the thesis were discussed in detail.
8. Outlook and conclusion: The last chapter sums up the outcome of the master's thesis, answers the research questions and gives a proposal for future work.

2. State of the Art

This chapter describes methods of water analysis, technical aspects, and other approaches. Before we look of the state of the art, lets point out how important water analysis are in general.

2.1. Importance of water analysis

The author Ralf Pohling explains in his book "Chemical Reactions in Water Analysis", why water analysis is so important. [Poh15] The author states, that water is the food number one and the foundation of all life on earth. The shortage of clean water has increased in the last decades and finds its reasons in climate factors and in progressing human made pollution, what is clearly a big global problem. For this reason, water analysis is more important than ever. Thanks to such analyses, the water quality can be evaluated, the quality of the wastewater can be controlled, and the degree of pollution can be estimated. As a result, people can keep waters clean and can avoid pollution. Additionally, water analyses are also important for drinking water treatment plants, because to guarantee their correct operating, some parts of this plants must be monitored every hour. As a consequence, it appears that there is a correlation between people's health and water analysis. In this context the responsibility of water analysis must be highlighted. [Poh15]
Finally, we can see how necessary water analyses are and therefore, the importance of continual improvement of such analysis processes. The next chapter describes classic methods for water analysis.

2.2. Classic methods for water analysis

This chapter describes methods and other solutions which are used in the water analysis area. The book "Wasseranalysen - richtig beurteilt" written by Walter Kölle gives a good overview about approved methods in the water analysis sector. [Köl17] These methods will have an essential role in the water analysis part of the model.

2.2.1. Limit values, concentrations, concentration-gradients

Limit values and concentrations are important measurements for the analysis of water. The level of limit values or concentrations of a specific parameter can be a direct indicator for the estimation of the pollution degree. Every limit value has his history and meaning. There exist limit values, which can exclude acute diseases or chronic effects. Other limit values have preventive characters and other are, derived because of the interactions between water and materials, finally some have just an aesthetic character. [Köl17, p. 24]

2.2.2. Geogenic or anthropogenic

The question if a bad influence is caused by nature or humans is a very important one. This cannot be solved that easy, sometimes are geogenic and anthropogenic factors both involved. An important decision criterion is here that the measurement should not contain "anomalous changes", such conditions should in any cases be clarified and if possible be revoked. [Köl17, p. 28]

2.2.3. Dealing with big data and data preparation

Dealing with big data sets will be an important part for the implementation of the prototype as historical data must be used for the statistical method. The advice of the author is here to save the data into data bases as the data must often be used for all kind of functions (e.g., statistical calculations). He also points out that metadata is very important, because when the information is lost to which measuring point the sample data belongs, then the sample data is useless. In our case it would be interesting if historical data misses some important information. Moreover, metadata is also data which can come from other authentic sources like laboratory diaries or sample taking diaries. [Köl17, p. 31]
In addition, the author also stated that there are various ways to prepare data for other functions: [Köl17, p. 31]

- Creating times series for the concentration of specific parameters
- Visualisation of geographical concentration dependence (concentration is location-dependent)
- Detection of correlation between different parameters (concentration is temperature-dependent)
- Detection of correlation between concentration and the flow of flowing waters (concentration decreases by increasing flow of water)
- Detection of complicated dependencies (e.g., Monitoring of flowing water with much industrial effluent during holiday time, many companies are closed)
- Calculation of frequency distribution (e.g., Gaussian distribution)
- Calculation of mean values

2.2.4. Statistical methods for water analysis

This sub-chapter describes statistical methods which are established in the field of water analysis.

2.2.4.1. Frequency distribution

When dealing with big data of numeric values for a specific parameter it is interesting to analyse the distribution for the values. [Köl17, p. 32] Following distribution are interesting for water analysis: [Köl17, p. 32f]

- Gaussian distribution: Is a normal distribution and occurs often.
- Multimodal distribution: Two or more peaks can occur when on the location of the measuring point more water sources with different nature encounter each other but do not mix too much.
- Logarithmic distribution: Is asymmetric and can occur for the smell parameter which is by nature logarithmic. It happens when measuring values can increase infinitely. For example, the concentration of undissolved substance (iron oxide) or in the case of turbidity, microorganisms. The solution for this is using geometric mean.

2.2.4.2. Arithmetic mean

Is an important part of the analytic and are used in many analysis cases and should not be applied on multimodal or logarithmic distributions. [Köl17, p. 35]

2.2.4.3. Geometric mean

As already mentioned, the geometric mean is important for logarithmic distributions. [Köl17, p. 36]

2.2.4.4. Median, percentile

The median on the contrary to the mean ignores outliers. With the arithmetic mean, geometric mean and the median there are three ways to deal with extreme values. This depends on the situation and on the parameters which are analysed. The median is a special case of the percentile and corresponds to the 0,5 percentile. Depending on the case other percentile types can also be useful to analyse with. [Köl17, p. 36f]

2.2.4.5. Dealing with outlier and faults

Outlier is data which differs so much from the rest of the data set, that it is obvious that they don't belong to this data. However, for the analysis for water, outlier must not be removed. Strong variation of measurement values demands highest attention. At last, it is exactly this parameter that vary the most, which can detect in bacteriological data a danger of epidemic or in chemical data an environmental scandal. Anomalous measured data can either be right or wrong. Ignoring such data is very dangerous, but reacting unreasonably when there is no threat, can be very embarrassing, especially when public is involved. Therefore, it is very important immediately to carry out accurate control investigation. In addition, one should also be carefully with correction of data, because historical data should not be overridden. It is suggested to perform version control. [Köl17, p. 37f]

2.3. Important on-site parameters for water analysis

This chapter describes the on-site parameters which must be recorded on the spot, and which are the basis for our plausibility checks and frequency analysis. When the sample taker arrives at the measuring point several on-site parameter values must be measured. This includes all information and parameters which can't be determined in the laboratory correctly: [TFMoAM15, p. 26]

- Physical or physico-chemical parameters: Temperature, pH-value, electric conductivity, dissolved oxygen
- Organoleptic parameters: Smell, colour, turbidity, water surface

2.3.1. Physical or physico-chemical parameters:

Physical or physico-chemical parameters are being measured with the aid of a measuring device and can yield important information about the condition of the water. [TFMoAM15, p. 32]

2.3.1.1. Temperature

The parameter temperature can give conclusions about the origin, possible pollution or influences caused by humans. Toxic effects of specific substances in the water are also depended on temperature and pH-values. Temperature is also an important factor for the colonisation of aquatic organisms. The temperature reacts sensitive on environmental influences like sunlight, earth frost and snow, which also should be minimised. [TFMoAM15, p. 32]

2.3.1.2. pH-value

The pH-values characterizes the acid and base content of a aqueous solution. It reacts very sensitively on temperature changes, degassing or chemical precipitation of insoluble compounds. For this very reason is the measuring of the pH-Value on the field very important, as analysis later in the laboratory can't be representative anymore. Normally the measurement takes place through glass electrodes, which are very sensitive and must be maintained regularly. Several factors can lead to a considerably interference in the measuring accuracy and to a replacement of the electrode. [TFMoAM15, p. 33]

Coatings like mud, chalk, or gesso (gypsum).

Chemicals like concentrated acid or base, acetone, or alcohol.

Mechanic loadings like cracking, particles, or vibrations.

Overaging extended stabilization time.

2.3.1.3. Electric conductivity

The electric conductivity [µS\cm] is a measure for the content of freely moving charge carrier (ions) in the water, which can conduct electricity. The electric conductivity of water is also a measure for the salt content (total mineralisation) and must because of its temperature dependence always be corrected to a reference temperature. The reference temperature is 20°C for groundwater and 25°C for surface water. The determination of the conductivity on the spot gives information about the representativity of the drawn sample. If the conductivity distinguishes from measuring points in the proximity, could this be a suggestion of different origins of the contamination. From a environmental perspective it makes more sense to immediately verify the measurement or to repeat the measuring after weeks or months. The electric conductivity can change over time, because of precipitation. Furthermore, pre-treatment of the sample (acidification) can also change the conductivity. [TFMoAM15, p. 35] An overview of the electric conductivity of different water types shows figure 2.1. On the left we can see the different types of water and on the right the electric conductivity. [TFMoAM15, p. 35] (Reinstes Wasser = clean water; Destilliertes Wasser = distilled water; Regenwasser = rainwater; Grundwasser = ground water; Leitungswasser = tap water; Mineralwasser = mineral water; Meerwasser = sea water)

Wasserart	**Elektrische Leitfähigkeit [µS/cm]**
„Reinstes" Wasser	0,055
Destilliertes Wasser	< 3
Regenwasser	5 - 100
Grundwasser	100 - 1900 (in Österreich im Mittel: 696)
Leitungswasser	150 - 800
Mineralwasser	150 - 2.000
Meerwasser	ca. 50.000

Figure 2.1.: Electric Conductivity
[TFMoAM15, p. 35]

2.3.1.4. Dissolved oxygen

The amount of the dissolved oxygen is proportional to the local atmospheric pressure and has a reverse correlation with temperature and salinity. As oxygen regulates the solubility of many trace metals and the bacterial substance conversion of organic compounds in ground waters, the measurement of dissolved oxygen enables to draw conclusions about the reduction of organic substances, origin of the water or mobilisation of metals. The oxygen concentration in one sample changes with time. Enrichment of the sample with oxygen through air contact or degassing by shaking, leads to a faulty measurement. Methods for measuring on the field are normally amperometric (Clark electrode) or spectrometric procedures (colour changing after adding chemicals). [TFMoAM15, p. 37]

Usual devices enable the measurement of following parameters: [TFMoAM15, p. 38]

- Concentration of the dissolved oxygen (mg\l)
- Relative concentration of the dissolved oxygen (%, pressure and temperature depended)
- Temperature

2.3.2. Organoleptic parameters

The organoleptical investigations include the qualitative inspection of colour, turbidity, smell, and sediment. The investigations take place when the parameters are stable and a transparent vessel with at least one litre volume is being used. The measuring must be done within a black or white background. Noticeable problems with the water sample or the measuring station can be determined beforehand. [TFMoAM15, p. 32] Following parameters can just be measured on the spot: [TFMoAM15, p. 32]

Colour: Identifying the colouring, intensity, and possibly temporal progress of the coloration. Drinking water must be colourless.

Turbidity: Is caused by a fine, suspended mineral or organic material. The documentation of colour and intensity of the turbidity must be performed.

Smell: Identifying the type and intensity of smell, e.g., earthy, musty or smell of certain chemicals. Drinking water should be odour-free.

Water surface: Identifying if the surface is polluted, e.g., film of oil, floating matters or foam formation.

2.4. Plausibility checks used by the Environment Agency Austria

This chapter gives an overview about the practical usage of methods bye the Environment Agency Austria. There are five types of analysis methods being used: [HLW18, p. 4]

1. Analysis based on chemical rules
2. Analysis based on statistic characteristics
3. Analysis based on limit values
4. Analysis based on ionic balance
5. Analysis based on electric conductivity

2.4.1. Analysis based on chemical rules

This analysis validates the measurement values against fixed defined rules. Following chemical parameters are validated: [HLW18, p. 4ff]

- Total hardness
- Carbonate hardness
- Oxygen
- Biochemical oxygen demand
- Chemical oxygen demand
- Dissolved organic carbon
- Total organic carbon
- O-Phosphate
- Total phosphor
- Other parameters: iron, manganese, nitrite, ammonium, pH-value

2.4.2. Analysis based on statistical characteristics

This analysis validates the measurement values against statistical characteristic values from historical data of the same measuring point. Depending what kind of water is being tested, the time series are between five (flowing waters) to eight (ground water) years in the past. [HLW18, p. 7] Following statistical measures are being used for this analysis: [HLW18, p. 8ff]

- Arithmetic mean
- Standard deviation
- 33% Percentile
- 66% Percentile
- Distribution

2.4.3. Analysis based on limit values

This analysis validates the measurement values against limit values, the validation data is not allowed to exceed these limits. This parameters and limit values are stated in the groundwater chemistry regulation, the national environmental quality standards, and the drinking water ordinance. [HLW18, p. 11f]

2.4.4. Analysis based on ionic balance

This analysis validates with the aid of an ionic balance. Therefore, cation parameters (Natrium, Potassium, Calcium etc.) and anion parameters (Chloride, Sulphate, Hydroxide etc.) are checked here. [HLW18, p. 13]

2.4.5. Analysis based on electric conductivity

This analysis calculates the electric conductivity for the measured parameters. Therefore, cation parameters (Calcium, Magnesium, Natrium, Potassium) and anion parameters (Nitrate, Nitrate, Sulphate, Hydrogen carbonate) are checked too. [HLW18, p. 16]

Finally, we can see that the Environment Agency Austria uses many methods and parameters to perform the water analysis on the drawn samples. We must keep in mind that on the spot not all of these parameters are available. Let us now look at other water analysing methods and water monitoring tools.

2.5. Modern methods for water analysis

In the previous chapters we have seen that the usage of statistical methods is established in the water analysis field. This chapter analyses different approaches of water analyses and methods.

The article "Efficient Water Quality Prediction Using Supervised Machine Learning" introduces a good machine learning approach and has many similarities with our problem setting. [ea19d] The authors state that, water quality is currently being estimated through expensive and time-consuming lab and statistical analyses, which are not a good solution for today's real-time monitoring systems. Therefore, their research explores machine learning algorithms to estimate the water quality index (WQI), which is a singular index to describe the general quality of water. Their algorithms are fed with four input parameters, namely, temperature, turbidity, pH and total dissolved solids. The results showed that gradient boosting and polynomial regression predict the WQI most efficiently. Whereas multi-layer perceptron classifies the WQI Class most efficiently. The proposed methodology achieves reasonable accuracy using not many parameters to validity if such methods can be used in real time water quality detection systems. [ea19d]
Here we can see similarities with our problem, because on the spot we analyse the same

parameters except of dissolved solids, in our case it is dissolved oxygen. It must be investigated in detail if such machine learning methods can be integrated into the prototype and if the water quality index is a good indicator for our analysis.

One interesting analysis is the machine learning approach for anomaly detection of water quality, written by Muharemi, Logofătu and Leon. [MLL19] In their work they analysed if their approaches can identify changes or anomalies occurring on water quality time series data. Moreover, the authors also discuss and propose a solution to some challenges when dealing with time series data. The performance evaluation was applied using F-score and the models which were applied to water quality data are: logistic regression, linear discriminant analysis, support vector machines (SVM), artificial neural network (ANN), deep neural network (DNN), recurrent neural network (RNN) and long short-term memory (LSTM). Each algorithm was performed in a simulation to check the F-score. The results show that all algorithms have their problems. Algorithms which tend to have less problems are, SVM, ANN and logistic regressions, on the contrary DNN, RNN and LSTM are very vulnerable. [MLL19, p. 294] Their experiment showed the weakness of machine learning algorithms when applied on highly imbalanced data set. They also mentioned that solving this imbalanced data set problem can be achieved by resampling the training set. [MLL19, p. 305]
As we can see there is a relation between the problems. In our case we will also have to create times series of the historical data, so that we can perform analyses. It must be investigated in detail if the H2O database also has problems with data anomalies. Furthermore, it should be explored if machine learning methods in such form are necessary.

Another water quality analysis is the survey "Comparison of Machine Learning Algorithms for Retrieval of Water Quality Indicators in Case-II Waters: A Case Study of Hong Kong", which was written by Hafeez and others. [ea19c] In this case study machine learning approaches for the retrieval of water quality indicators over the coastal waters of Hong Kong were analysed. The aim was to estimate the concentrations of suspended solids (SS), chlorophyll-a (Chl-a), and turbidity using water reflectance from both a hand-held spectroradiometer and satellite data. Following machine learning techniques were tested: Artificial Neural Network (ANN), Random Forest (RF), Cubist regression (CB), and Support Vector Regression (SVR). The results indicate that the NN-based machine learning approaches perform better and, thus, can be used for improved water quality monitoring with satellite data in optically complex coastal waters. [ea19c, p. 1]
For this method it has to be checked if such analyses using spectroradiometer and satellite data, are performed on the side of the Environment Agency Austria. Furthermore, it should be explored if machine learning methods in such form can improve the support model.

2.6. Water quality tools

This chapter analyses different types of water quality tools. The aim is to investigate the importance of other tools and devices to identify synergies.

One interesting open-source tool is the AkvaGIS module which is included in QGIS (QGIS = Open Source Geographic Information System [QGI]). AkvaGIS supplies a standardized workflow for the storage, management, visualisation, and analysis of hydrochemical and hydrogeological data. The main function simplifies the characterization of groundwater bodies for building data-based environmental conceptual models, which are important in Europe, because of the Water Framework Directive. Especially for data-based groundwater management, AkvaGis can prepare input files for numerical models of groundwater flows in all the available formats in QGIS. The tool was applied in Belgium (Walloon Region) and the results support a better understanding of the hydrochemical relationship among aquifers in this region. Moreover, it can be used for the development of new analysis, for nitrate vulnerable zones and management of the monitoring network to control chemical spatial and temporal evolution. [ea19b, p. 1]
This add-on could be very interesting for the survey issue, because the environment agency currently uses QGIS. It should be investigated if using AkvaGIS is relevant for the scope of the master's thesis.

Another open-source tool is EDAM (Environmental Data Acquisition Module), which is a generic template framework that facilitates timeseries data acquisition and integration. Data acquisition and integration is an important step of the environmental data lifecycle. Moreover, is heterogeneous data a big issue, as systems are using different protocols and data is stored under diverse formats. EDAM deals with syntactic heterogeneity in environmental timeseries data and describes different dataset syntaxes via templates. The templates apply programming language-agnostic semantics and can be used for input and output data. Reusing data can also be enabled via transformation across different formats. EDAM demonstrates its generality in scraping online data, extracting observations from database, or aggregating historical timeseries stored in local files. It is used in many domains, including meteorology, agriculture, urban air quality and hydrology. [SPA19, p. 1]

The next tool is not a software application, but a highly sensitive, portable photonic device for pervasive water quality analysis, called WaterSpy. The device is a cost-effective device for pervasive water quality sensing, operating in the mid-IR spectral range. The device applies advanced Quantum Cascade Lasers (QCLs) applying the Vernier effect as light source. Moreover, it uses fibre-coupled, fast and sensitive Higher Operation Temperature (HOT) photodetectors as sensors. In addition, WaterSpy also supplies optimised laser driving and detector electronics, laser modulation, signal conditioning and signal processing techniques. [ea19a, p. 1] There is also a possibility to add pattern analysis and machine learning techniques. The devices include following advantages: cost-effectiveness,

modularity, leveraging existing technologies and know-how, maximization of open software usage and compatibility with other technology and European regulations. [ea19a, p. 7]
WaterSpy can effectively detect the presence of three pathogens, which measurements are obligated by the European legislation. The three microbiological parameters are heterotrophic plate counts (HPC) and the two bacterial indicators Escherichia coli and Enterococcus spp. [ea19a, p. 2]
For this device it must be checked if such microbiological parameters analyses, are performed on the side of the environment agency and if they are part of the research problem. It should be explored if such extra device could be a benefit for the water sampling process.

2.7. Technology

This chapter gives an overview about the implementation of the prototype focused on properties of various web software. As already mentioned in the introduction chapter, there are some system requirements the prototype must fulfil, the most important ones are mobility and platform independence.

2.7.1. Mobile software

Nowadays it is possible for mobile software development teams to choose between several different development strategies, like native apps, mobile web apps, hybrid apps and progressive web apps. [Mal16]

2.7.2. Why progressive web apps?

Progressive web apps also called PWAs, are a special kind of mobile web apps. Their characteristics is progressive enhancement, support for low or no network connectivity, background processing capabilities, push notifications support, and security benefits. They work with HTTPS and can be initially accessed as a standard web app via a browser, no install is required. After the first access the user can choose if he wants to install the PWA. [Mal16] PWAs combine the benefits of a native app with the low friction of the web. [Ate17]
Native mobile apps on the contrary are distributed via an app store and must be installed. In addition, the use of platform-specific technologies leads to challenges, as code written for one platform cannot be used on another platform. The code must be written for every platform. [Mal16]
The third approach is a mobile web app, they are developed with web technologies and act like mobile-optimized websites accessed with the browser. The advantage is that they can be started in every browser on every platform, but the disadvantages are handling heavy graphics and no access to low level features like background services management. [Mal16]
Another approach is a web-based hybrid mobile app, which allow developers to create a cross-platform web-based mobile app by providing a native wrapper for containing the

web-based code and a generic JavaScript API that bridges all the service requests from the web-based code to the corresponding platform API. The advantages are that through the native wrapper, the app can be packed and distributed for any supported platform, but disadvantages are that the platform APIs can just be accessed via the JavaScript bridge and this creates a performance overhead. Moreover, every platform has its own characteristics so the code must be adjusted for every platform. (e.g., Android back button, whereas iOS has back on screen) [Mal16]

Finally, when we compare the requirements on our problem to the properties of the mobile app types, we can clearly see that the best solution for our prototypes is a progressive web app. All the advantages of PWAs help us to fulfil our requirements. In the book "Building Progressive Web Apps: Bringing the Power of Native to the Browser" written by Tal Ater, are these benefits described in detail. [Ate17] Following benefits will help us to implement a progressive and modern web application: [Ate17]

- Availability regardless of connection
- Fast load times
- Push notification
- Home-screen shortcut
- Native look

2.7.3. React js

The author Scott Domes describes in his book why React js is a great solution for the front-end technology. [Dom17] So why React? React has quickly become good solution for front-end web applications, because it is fast, elegant and makes managing large applications easy and complexity simple. For PWA one could also use other technologies, but one major benefit of React is, its component pattern. The user interfaces are split into distinct pieces, therefore the component pattern breaks the interface into small chunks of code, to decrease the preceding JavaScript downloading issue. Another advantage of React is that is an in-demand skill and future-oriented, because when pairing the knowledge of React with experience with PWAs, one will be future-ready as one can be in the fast-moving web development world. [Ate17]

2.7.4. Interview with front-end developers of the Environment Agency Austria

This interview took place on the 28.10.2019 with two front-end developers (Walter Huimann and Denis Masinovic) of the Environment Agency Austria. The aim was to experience which kind of web technologies are developed by the IT department of the Environment Agency and why? Furthermore, the two developers should assess which web solution suits the best for the water sampling problem. [HM19]

2.7.4.1. Why Web App?

The developers stated following advantages of web apps and why they are a good solution for this specific problem.
Web apps are per default platform independent, on the contrary native applications need to be adjusted for every platform. Web apps enable fast implementation and rapid prototyping, what leads to quicker results. One special form of a web app is the progressive web app. A PWA is a symbiosis of a responsive web page and an app. They have a great performance and use service workers for optimised caching to supply an offline mode.

2.7.4.2. Why JavaScript?

In this sub-chapter the developers stated that non-JavaScript techniques are very reliant on server communication (bandwidth latency). For every user action, the application sends or recalls the whole web page. JavaScript web pages load the whole web site into the browser cache and work just with the required data. When performing a user action, just necessary data is loaded, there is no redundant loading of code. As a result, this makes web applications written in JavaScript so well-performing. Finally, we can see that these JavaScript properties suit the sampling problem very well, because it could be very likely that the sample taker is facing such an situation that he has a low internet coverage, therefore it is important that he is able to continue to operate the application.

2.7.4.3. Why React js

The developers explained that in 2017 three JavaScript frameworks were very popular: Angular, VUE and React. Back then Angular was very famous in Europe but not globally. For React a trend was recognised in the USA and globally. In contrast VUE was very new and had a small community. Therefore, the developers made an evaluation and came to the conclusion that React js would probably be the best solution, because of following advantages:

- On-boarding: Getting employees and new people to work with React. One reason is that the basics are easier to learn and knowledge faster to achieve, than with Angular.
- React is a rendering library and closer to standard JavaScript, so it's easier to learn React than Angular, because Angular is a framework and thus very specific to learn.
- Dataflow is easier to understand in React, which makes defect-debugging easier.
- Breaking changes (in 2017): Angular was very radical when launching a new Angular API (Angular 1 to Angular 2 brought radical changes) this brings very much effort for refactoring. On the other side React was always very carefully in changes, older versions were always compatible (Very important for the Environment Agency).
- The React community was always big and still is. Their environment provides many tools for the framework.

2.7.4.4. Why to use React js for this specific problem?

As already mentioned, the internet coverage in the field can be very bad and the system should still work, therefore a JavaScript framework is very suitable. Cross-platform functionality and mobility is accomplished through the properties of the web app or PWA.

2.7.4.5. Why React DOM and not React Native

As React Native must run on various platforms, for certain use-cases there must be implemented an own logic for each platform. React Native is platform specific and requires modification of the code to gain platform independence. A big advantage of React Native is that it feels like a real app (native app), because of performance and user experience, but this is not important for the prototype. Finally, to make a React Native App it requires more effort, on the contrary DOM is per default platform independent and is easier to develop.

3. Research approach

This chapter describes the research approach of this master's thesis. The aim is to apply a software development process which suits the development of the water analysis model and prototype.

3.1. Applied methods

3.1.1. The Software Development Process - The Spiral Model

The spiral model is based on the experience with various refinements of the waterfall model and it serves a more robust foundation for a software development environment than other models in that time. The model contains of several cycles and is designed in such a way that every cycle goes through the same steps, for each portion of the product and for each level of elaboration. This goes from the conceptual phase down to coding itself. [Boe86, p. 24] The spiral model and its cycles are illustrated in Figure 3.1.

3.1.1.1. Cycles of the Spiral Model

Each cycle typically starts with the identification of the functionalities of a portion of the product (performance, functionality etc.). The next step is to find alternatives for implementing this portion (design A, design B etc.) and to consider the constrains (cost, schedule etc.) of each alternative. The next step is to evaluate objectives and constraints of the alternatives. This process identifies areas of uncertainty which are important sources for the risk analysis. A cost-effective strategy for resolving the risk must be formulated and certain risk-resolution techniques like prototyping, simulation, administering user questionnaires or analytic modelling must be applied. After evaluating the risks, the next step is determined by the relative risks remaining. If risks are still left, an evolutionary development step must be executed: specifying the nature of the product, planning the next level of prototyping, and developing a more detailed prototype to resolve the major risk issues. If previous prototyping has already resolved all the performance or user-interface risks, the next step follows the basic waterfall approach, modified in incorporate incremental development. [Boe86, p. 24ff]

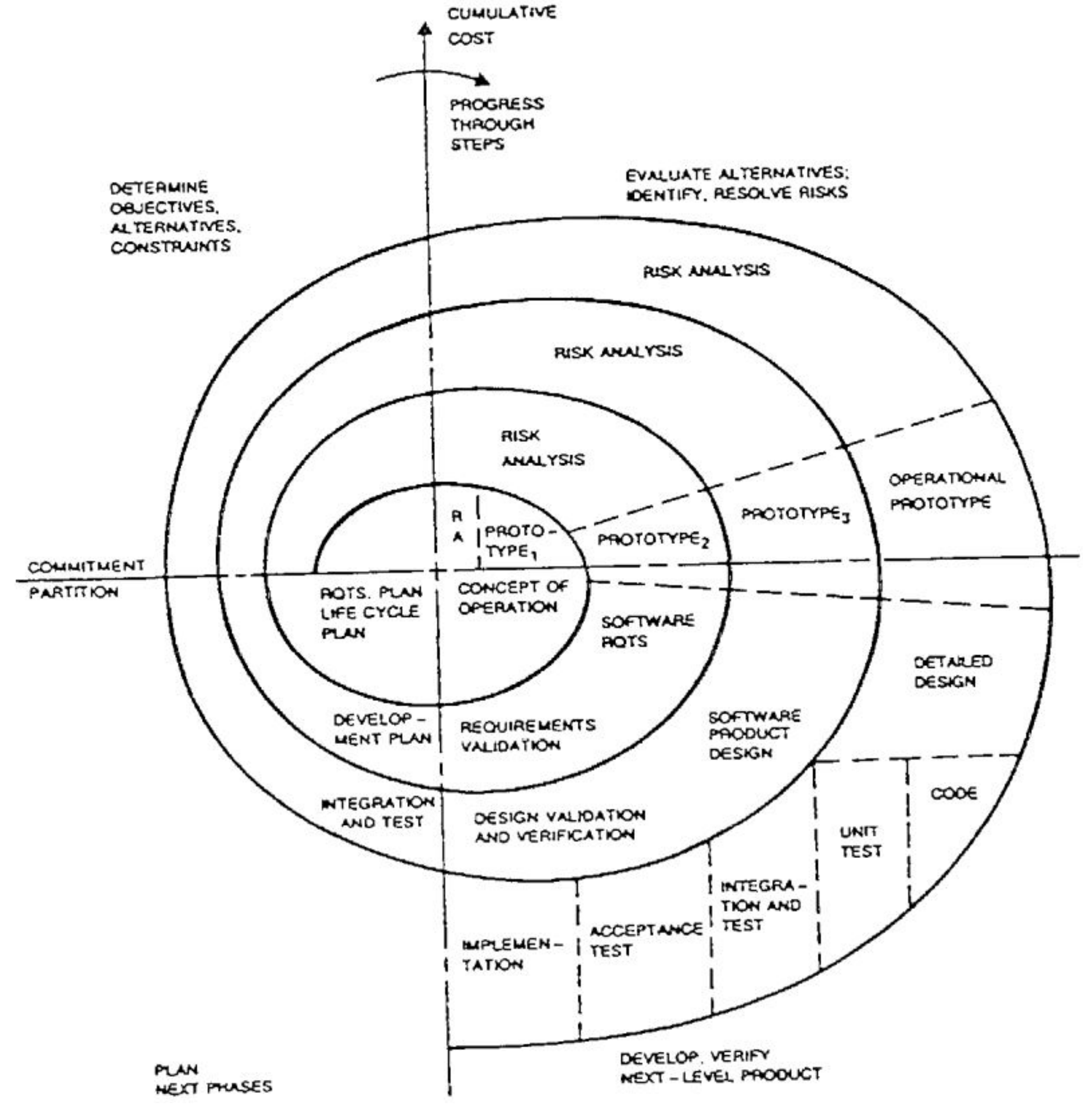


Figure 3.1.: Spiral Model of the Software Process
[Boe86, p. 25]

3.1.1.2. Applying the Spiral Model

When applying the spiral model for this specific problem, every cycle of the spiral model should produce a new version of a prototype and apply new statistical methods for the water analysis part or rule-based system part. The result of the first iteration of a cycle must generate a base application, the first prototype version of the water analysis tool which is the foundation for the data analysis. The cycle starts with a planning and concept phase. The aim is to identify the technical requirements and to model the prototype. The design alternatives must be analysed so that objectives and constraint of the first version of the prototype can be evaluated. The alternative which fulfils the requirements and can be implemented at low-risk should be chosen. At the end of the first cycle the prototype must contain basic functionality like loading, saving, and updating data of measuring points and parameters. A base application is mandatory for applying a water analysis method or a rule-based system. If problems with data processing or other risks are identified, then a new cycle must be started to adjust and correct the prototype.
After implementing the base, the next cycles can be executed. The aim of the following cycles is to find water analysis and rule-based systems alternatives until the scientific problem is solved. At the end of the last cycle a software process validation (like implementation test, scenarios test, acceptance tests etc.) must be executed. [Boe86, p. 25] The method for the validation and verification of the prototype is described in the next chapter.

3.1.2. Software Process Validation - Comparing Process and Practice Models

To assure quality in software processes, various types of models play an important role. Process models represent the officially defined software development processes in organizations. Practice models on the other hand describe how it is really done and are considered as important for validating the software process. In reality, the software development process often differs considerably from the official process. Another validation approach are formal methods, they compare processes and practice models in various combinations. The method detects differences between these models easily and allows software developers, managers and engineers to interpret these differences and to make recommendations for software process improvement. [DMD06, p. 171]
Analysing and comparing of formal models which are specifying requirements, analysis, architectural design and so on, are an integral activity in the software development process. Therefore, it is normal for participants to consider formal models in order to validate the software development process itself. [DMD06, p. 172]

3.1.2.1. A Conceptual Model of Software Process Validation

The starting point is the intended process, the point in which one must envision the software process. However, at the end the actual work often differs from the intended

process. How can this process be validated? Validated in this way means any approach that can evaluate differences between models. [DMD06, p. 172]
Figure 3.2 presents the conceptual model of software process validation with 4 types of validation: (1) first-hand observation, (2) process-model validation, (3) practice-model validation, and (4) model comparison. [DMD06, p. 172f]

Types of validation: [DMD06, p. 172ff]

1. **First-hand observation:** Is a classic kind of validation like when the manager makes a visit in the office and checks whether the work is progressing. In large companies and projects is such a validation not sufficient anymore.

2. **Process-model validation:** Is a validation that compares management intentions with the process model. A process model describes the plan how a business process should be conducted. The description is a (informal) meta-model of a development process, but the actual intention of the management is often different. For this reason, the needs of the management often change in unintended ways during the specification of the process model. One way to validate the process model is to hire external consultants and have discussions based on analysis of the process with industry standards (e.g., CMM - Capability Maturity Model).

3. **Practice-model validation:** The third type of validation is between the actual work performance of the staff members and their individual practice models. This validation can be useful to allow employees to improve their skills, by letting them reflect between what they are doing and what they think is their optimal work performance. This learning process can be supported by techniques as videotaping, logging command sequences etc.

4. **Model comparison:** The fourth validation is a model comparison between the official process models and individual practice models. Due to many participants with very diverse interests and responsibilities, the theoretical practices (process models) and the individual real practices (practice models) are often far away from each other. [Win95] When old patterns and experiences no longer work, new organizational patterns must be deployed by staff.[KS03]

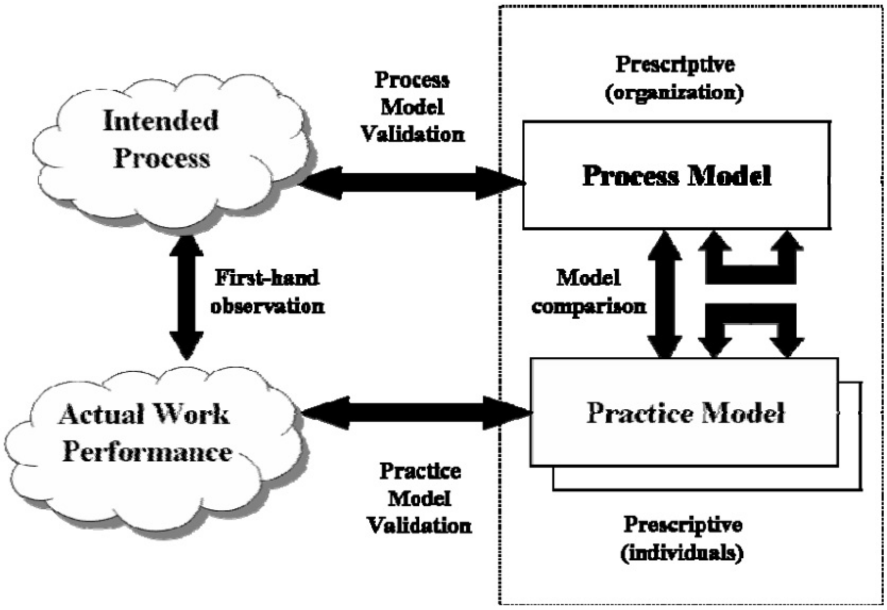


Figure 3.2.: A Conceptual Model of Software Process Validation [DMD06, p. 173]

3.1.2.2. A Formal Method for Software Process Validation

How can model comparison efficiently be facilitated? A good solution are formal methods, as they include many advantages like: (1) Reducing ambiguity (2) Preventing conflicts (3) Prerequisite for many sophisticated analysis techniques. [DMD06, p. 174] [VDA98] The formal method works as a top-level software process validation ontology, where core instruments are workflow, step, and activity. One key attribute of a software development process is that it consists of workflows (like requirements acquisition) which lead to the final deployment of software. The core of the analysis are activities in which an object is produced. All activities must be initiated, executed or evaluated by an organisational actor, which are called control processes or simply controls. Norms also come with the controls, and they describe if an actor has a responsibility, a privilege, or a prohibition. Moreover, the ontology contains of a number of activity patterns. Definitional patterns represent activities which must have concepts and relationships. Schematic patterns on the contrary represent ones which can have concepts and relationships but must not have. The method steps are outlined in Figure 3.3. [DMD06, p. 174f]

• **Create activity patterns for the models**
• **Create two models**
o Models can be either process or practice models
o Identify model steps
o Build models by instantiating or specialising general activity patterns for each step
• **Compare the models**
o Identify activities in models, and delineate steps in a sequence
o Compare steps between the models
o Create difference graphs
• **Interpret differences between the models**
o Present difference graphs to software professionals
o Adapt process and/or practice models
o Launch change program

Figure 3.3.: A Formal Method for Software Process Validation [DMD06, p. 175]

3.1.2.3. Applying the Model Comparison Validation

In the article "Software Process Validation: Comparing Process and Practice Models" the authors focused on one validation process that compares two process models from the aerospace domain. Searching for differences between models is important for identifying problems, opportunities and improvements to ensure quality control. They used a conceptual graph theory model, which is a powerful knowledge formalism to compare the complex knowledge structures such as process and practice models. Interpretation and evaluation of these differences were carried out by domain experts such as software engineers and managers. The method was found useful by its intended users and provided new insights at relatively low cost. For each difference, they were able to identify new actions or suggest further examinations and showed that the technique helps software development practitioners to improve process quality. [DMD06, p. 178f]
Applying this method on a software from the aerospace domain brought successful results, therefore it would also make sense to apply it on software from the environment protecting domain. In our case applying the model comparison should usually compare a process model and a practice model. However, it can also occur that more solutions are modelled or even designed, then comparisons between variants of process models or comparisons between practise models are possible. Interpretation and evaluation of the model differences should be carried out together with the domain experts of the Environment Agency Austria, as they have a comprehensive know-how and deep understanding and knowledge in the water analysis domain. The aims are to validate the different solutions (models), to identify problems, opportunities and improvements and as a result improve process quality and finally data quality.

3.1.3. Knowledge-based system

3.1.3.1. Knowledge representation

The notion of Knowledge Representation (KR) has to do with writing down, in a language or communicative medium, description or pictures that correspond in some way to the world or a state of the world. In Artificial Intelligence (AI) we are writing down descriptions so that an intelligent machine can come to new conclusions through manipulating these descriptions. AI is knowledge-based, not because that they behave as if they had knowledge nor that it takes knowledge to write them, but rather that their architecture includes explicit knowledge (direct symbolic encodings in the knowledge of the system). [BL85]
The methodological assumption for this finding is called "Knowledge Representation Hypothesis" and is presented in the next sentence.

"Any mechanically embodied intelligent process will be comprised of structural ingredients that (a) we as external observers naturally take to represent a propositional account of the knowledge that the overall process exhibits, and (b) independent of such external semantical attribution, play a formal but causal and essential role in engendering the behaviour that manifests that knowledge." [Smi82, p. 2]

To have an explicit knowledge base, a system must contain a well-specified language for encoding its beliefs, a knowledge representation language. The system must include more than just the literal set of sentences (or frames, rules, etc) to represent what it knows. Moreover, it must provide access to facts implicit in the knowledge base, it must perform automatic inferences for the user. Next to the representation language and the inference scheme the third ingredient is the domain knowledge. They say that the power of an AI system lies in the knowledge itself, which means that the ability of an intelligent system to diagnose or to troubleshoot something, depends on the detailed explicit knowledge base that represents the system's understanding of its domain. Finally, to develop good systems for this three ingredients is the main problem of AI, the representation problem. [BL85] In other words it can also be described like this:

"All parties to the debate agree that a central goal of research is that computers must somehow come to "know" a good deal of what every human being knows about the world and about the organisms, natural or artificial, that inhabit it. This body of knowledge-indefinite, no doubt, in its boundaries-goes by the name "common sense." The problem we face is how to impart such knowledge to a robot. That is, how do we design a robot with a reasoning capacity sufficiently powerful and fruitful that when provided with some subbody of this knowledge, the robot will be able to generate enough of the rest to intelligently adapt to and exploit its environment?" [Isr83, p. 37]

This hypothesis can be seen more as a working hypothesis about a reasonable way to develop intelligent systems, than a explanation of a rational cognitive behaviour.

[BL85] Knowledge-based systems are virtually synonymous with AI systems, and both are those systems that satisfy the knowledge representation hypothesis by design. The study of interpreted symbolic structures and their formal manipulation in a truth-respecting way is the foundation of AI. The application of logic to reasoning was pursued in the field of knowledge representation and reasoning. This subfield is concerned with the forms in which knowledge can be expressed, the meaning of such forms and the reasoning with them. [Bra88, p. 8f]

3.1.3.2. Logic Programming

The main idea of logic programming is to make it possible to provide procedural semantics for statements in certain logical formalisms. Therefore, programming is reduced to provide a program specification that is represented as a set of axioms. This set is hand over to a sound and complete Theorem Prover [Bun97, p. 121], which constructs a proof for a solution. This approach is justified in this way that, classical logic is well understood, and hence systems can be provided which have a great, theoretical foundation. However, this system can only work when this formalism is sufficiently powerful and desirable that it is also sufficiently expressive. Normally are the programs written in the Horn clause form of first-order Predicate Logic [Bun97, p. 92] and the best-known example to develop such a system is by using Prolog. [Bun97, p. 95] [Bun97, p. 67f]

3.1.3.3. Applying the knowledge representation - Prolog

Prolog is a very simple but powerful programming language founded on symbolic logic, as a practical tool for "logic programming". Pattern matching process ("unification") is used as basic computational mechanism, which operates on general record structures ("terms" of logic). [WPP77, p. 109] Prolog differs from most programming languages, as there are two ways to understand its semantics. The procedural semantics is the more conventional way and describes the sequence of states passed through when executing a program. Additionally, the Prolog program can also be understood as a set of descriptive statements about a problem. The declarative semantics from logic provides a formal basis for such a reading. [WPP77, p. 110]
The declarative semantics defines (recursively) the set of terms which are true according to the program. A term is true only if the head of some clause instance and each of the goals of that clause instance is true. Finally, this declarative aspect of Prolog is responsible for promoting clear, rapid, accurate programming, because it allows a program to be broken down into small, independently meaningful clauses. [WPP77, p. 110]
The procedural semantics on the other side, describes how a goal is executed. The objective of executing is to produce true instances of the goal. It is important to notice that the ordering of clauses in a program and ordering goals in a clause, represent important control information for the procedural semantics. [WPP77, p. 110]
Finally, what speaks in favour of using Prolog is that it allows a program to be formulated into smaller units, each of it having a natural declarative reading. Furthermore, it gives the developer generalised record structures with a great mechanism for manipulating

them. Additionally, it allows programs with similar behaviour to be coded without having to use machine- or implantation-oriented concepts. [WPP77, p. 113]

3.2. Why not neural networks and deep Learning

While neural network-based technique's, such as deep learning have their merits, they are not suitable for the tasks supported by the system. The problem in this context is the lacking explainability of the conclusions and recommendation produced by neural networks. If the water sample is classified as polluted it must be possible to reproduce the underlying experiment, including the reasoning done by the system. That is why this technology is considered as out of scope.

3.3. Conclusion

In this chapter we have seen some interesting software development approaches. The spiral model suits the development process very efficient, as various fields, technologies and frameworks can be implemented in separately cycles. Through these cycles the prototype can be expanded from cycle to cycle. It is also possible to integrate the model validation methods described in the previous chapter into each cycle, and so validate each prototype version. One can clearly see that the combination of both approaches is a efficient way to develop the support model and the prototype. Furthermore, knowledge-based systems were introduced through knowledge representation and logic programming. Knowledge representation has to do with writing down, in a language or communicative medium, description or pictures that correspond in some way to the world or a state of the world. Knowledge based system and artificial intelligence share similar characteristics as both architectures include explicit knowledge, in other words direct symbolic encodings in the knowledge of the system. Both work in that way that they are fed with descriptions which they manipulate to come to new conclusions. This concept is very important for the support model. Finally, when applying such a knowledge-based system, logic programming with Prolog seems to be the best solution. The Prolog knowledge-based system should be integrated into the support system as independent module or container.

4. Model

This chapter explains the support model in detail and how the problems described in the introduction can be solved. In the following sub-chapters, the solution concept is brought closer by the different types of modelling. Through the various types of models, the scenario should be visualised from several points of view. The approach is, to start with describing the scope, focus and challenges of the model and after that proceed with a model-based approach by presenting the model primarily from an use-case and process view and secondly from a technical view too.

4.1. Scope and focus: goals and non-goals

This sub-chapter describes the scope, focus, goals, and non-goals of the support model. As shortly explained in the "Introduction", the focus of this support model is to develop a system which can support the sample taker by his investigations. The model which will be described in this chapter should provide a solution how this support can be designed. The support should in any case help the sample taker to save valid and correct data during measuring and minimize or completely prevent errors. Additionally, if the system measures a deviation, it should give the sample taker a recommendation about the next actions he should carry out. Therefore, the objectives of the model are, saving data quality, recognising deviations and providing reliable recommendations. Due to the fact, that false data recorded during sample-taking can cause problems for interpreting the situation and in the worst cases let to not detecting environmental pollution. Finally, through a scenario-based model validation it should be confirmed that the purpose of the model is achieved.
The next sub-chapters "Coverage" and "Core model requirements" will illustrate the characteristics of the model further.

4.1.1. Coverage

Coverage of the model: From the functional point of view, the model will focus on water bodies of the type streaming water (rivers, streams etc.), especially these types which have a normal Gaussian distribution. In addition, the model will cover the most important recommendations which are important for these types of water-bodies. Finally, as already mentioned before, the statistical analysis will concentrate just on the parameters measurable directly on the spot.

On the field test: It must be clear that it is not possible to execute a validation or real test on the field since the water data would likely not contain any deviations. Such an experiment would probably demonstrate that no outliers were detected, and that the measured on-the-spot parameter values are valid. Finally, such a real user test would involve a great deal of effort, for the Environment Agency Austria and for the Ministry too.

Ability to cover the specifics of all water bodies: As this model describes a prototype, it is not possible to cover the specifics of all water bodies. In other words, every water body type has special characteristics. Including all water body types with all the characteristics would be too complex. However, it would be interesting to expand all types of water bodies for future work.

Ability to cover a large number of recommendations: As this system is a prototype, it is not possible to include a very large number of recommendations, but the recommendations must cover the validation scenarios. Including an extensive number of recommendations would be too complex. However, designing the model in such a way, that expanding the system easily, is desirable.

4.1.2. Core model requirements

This sub-chapter describes the core requirements driving the development of the model and its components.

Statistical methods: The model must contain statistical methods to be able to analyse historical data of the water body. The results of these statistical calculations are the basis for the rule-based system and the computation of its recommendations. Therefore, it is important that the methods can provide reliable results on what the rule-based system can build on.

Rule-based system: The rule-based system must be able to provide reliable and correct recommendations to the sample taker. In addition, the system should also be extendable with new rules. Through expandability the system can be upgraded and supplied with new recommendations for the sample taker. However, when expanding the system, the maintainability must be considered, as unorganised updates have a particular tendency to become unmaintainable and inflexible over time (e.g., avoid contradiction).

Simulation scenarios: The model and its functionality must be validated trough simulation scenarios. These tests must validate that the statistical calculations can detect deviations and that the rule-based system is able to give the right recommendation for the specific scenario. After executing the different scenarios and elaborating the results, a direct comparison between the expected results and the actual results must be performed for every scenario.

4.2. Challenges

This subsection describes the challenges of developing the model:

Modelling at the right level of detail: In the beginning it is essential to find a sufficient level of detail for modelling the support application. The model should be described in a level of detail so that all scenarios are covered and fully understandable. When the level of detail is too low, the model and the described scenarios are covered superficially. A too high detail level, on the contrary is a never-ending story of modelling.

Complexity of the water analysis domain: The water analysis domain is an inherently complex area with massive data of all types and different schemes making it hard to integrate information from different sources. At the modelling level this results in having to combine a wide variety of data models, functional models, and use-cases. The current situation corresponds to stage 3 and 4 from Richard L. Nolans paper "Managing the Computer Resource: A Stage Hypothesis". In this paper a descriptive stage hypothesis for data processing is presented. Nolans suggests that planning, organizing, and controlling activities of computer resources will change in character over a period and describes them in following stages. Stage 1 "computer acquisition", stage 2 "intense system development", stage 3 "proliferation of controls" and stage 4 "user/service orientation". [Nol73][p. 399] Stage 1 (Initiation stage) is about the introduction of the computer in the organization. Computer-based systems got required and old manual systems disappear. The managerial commitment marks the transition to stage 2 and raising the responsibility and reporting level are techniques used reacting to problems. Stage 2 (Contagion stage) is characterized by creating strategies to encourage alienated users to investigate the potential for computing. [Nol73][p. 401f] In the case of the environment agency both stages are already surpassed, but the situation for designing this water analysis model is similar to stage 3 and 4. Stage 3 (Control stage) describes the moment when entering into the result of crisis. Management mobilizes a set of tasks to control expenditures for computing, where inefficiencies in computer applications are usually the first target. Planning tasks are executed in all aspects of the computing organization. The aim is to evaluate the systems plan and to establish priorities for the future. The tendency is generally re-centralising the computing activity and developing of standards for programming, documentation, and operations. [Nol73][p. 403] In the last stage 4 (Integration stage) refinement of the control tasks and the elimination of the unnecessary tasks are performed. Project control systems and the management reporting systems become integral components of the management function. Moreover, planning tasks become more comprehensive and are linked to budgeting processes and to corporate planning activities. The most important characteristic is rethinking the role of the computer resource in organisational goal achievement, by upsurging user empathy and by the need to design solutions which effectively satisfies user needs. [Nol73][p. 403] In the case of developing the support model, it can clearly be seen that this developing process belongs to phase 3 and 4. As described in phase 3, the phase starts when problems occur. To handle problems, control and planning mechanisms must be executed. The challenge is here to plan and design the model in a effective way which handles the general aim of the model and the already mentioned problems of having to combine a wide variety of data models, functional models and use-cases. For stage 4 applies, planning and modelling a system which suits the needs of the user, by creating a necessary and use-full solution for the future. Finally, all these challenges must be considered in this modelling phase.

Technical feasibility: The designed model should also be implementable. Mistakes made in the modelling phase which are discovered during the implementation phase can cause problems e.g., a specific modelled scenario is not implementable for various technical reasons. Reasons can be that specific technologies, frameworks and modules are not compatible with each other or very difficult and complex to integrate into the system.

4.3. Model

This chapter describes the model through various modelling types. The use-case and process diagram should explain the scenario of the model. The component, entity relationship and user-interface diagram specify the model from a technical view.

4.3.1. Use-case diagram

The use-case "Add sample data" starts with the intention that the user (sample taker) wants to carry out a withdrawal of a water sample. This activity includes that the measured values of the on-spot parameters must be validated through a water analysis or plausibility check. The interpretation of the water analysis results is performed by the rule-based system and when no deviation occurs, then it must be assumed that the data is correct, and the sample can be saved. If a deviation exists, the activity "Take other actions if deviation exists" must be performed by the sample taker. For this activity various cases occur. The second use-case is "View sample data", in this scenario the user can view sample data from previous withdrawals, for information purposes. Figure 4.1 demonstrates the use-case diagram.

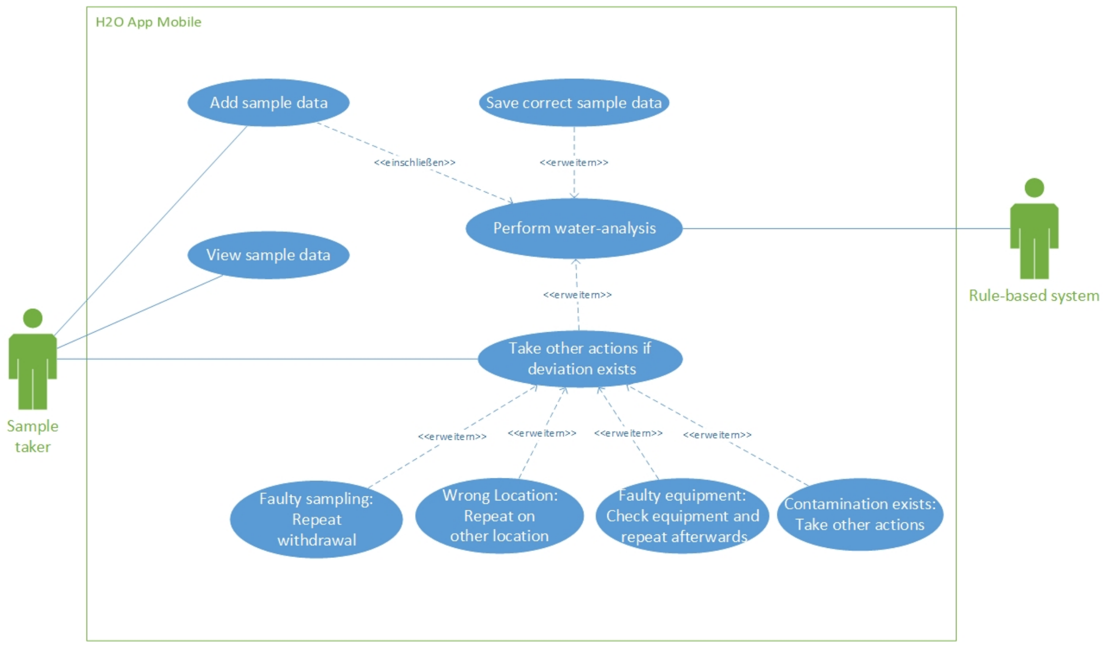


Figure 4.1.: Use-case diagram

4.3.2. Process diagram

This process diagram describes the previous presented use-case from a more detailed view.

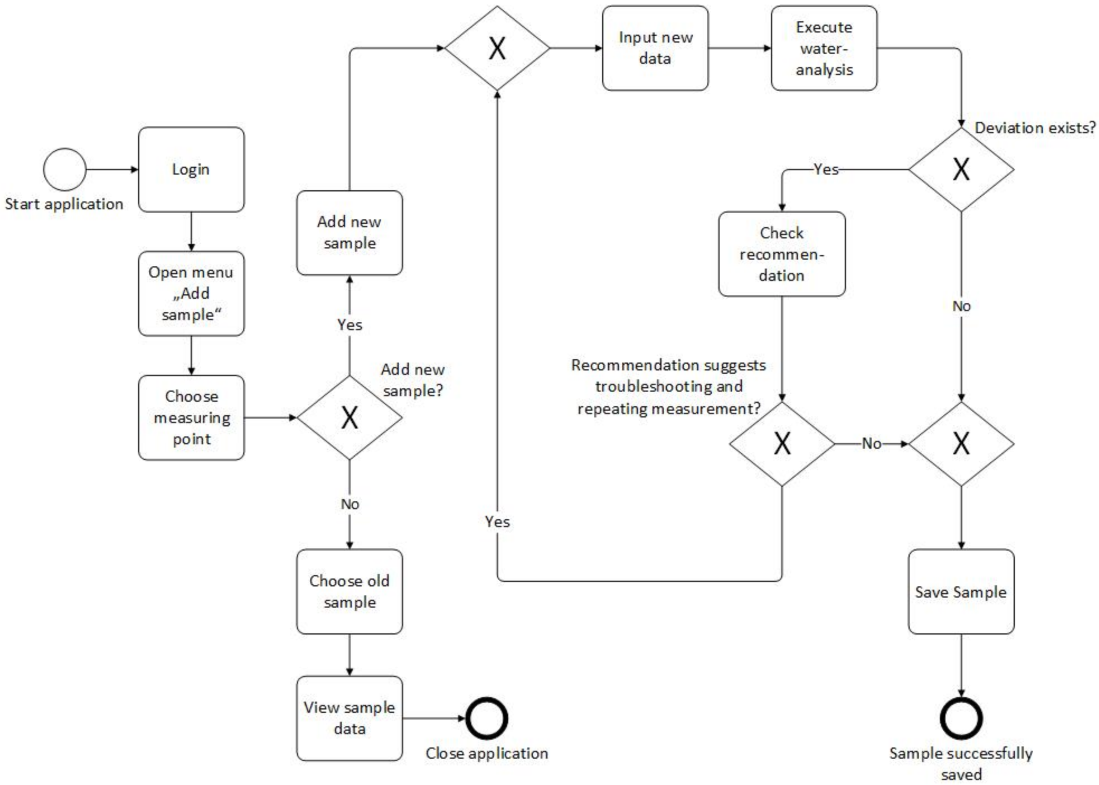


Figure 4.2.: BPMN diagram

4.3.3. Component diagram

This diagram presents the components of the system from an architectural view. The exact description can be found in chapter 5.6 "Prototype architecture".

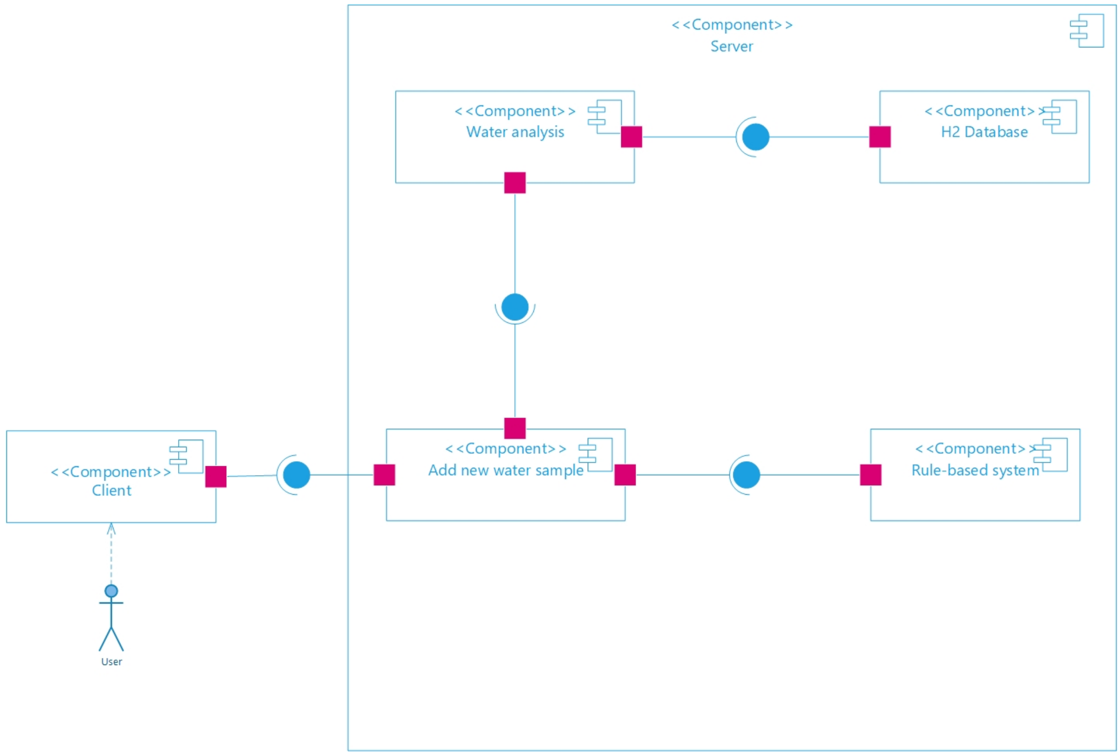


Figure 4.3.: Component diagram

4.3.4. Entity relationship diagram

This chapter describes the tables and relations of the database used by the prototype. The ER diagram is the default modelling language within the informatics department of the Agency for presenting data models. Therefore, the data view of the prototype is also designed as ER diagram.

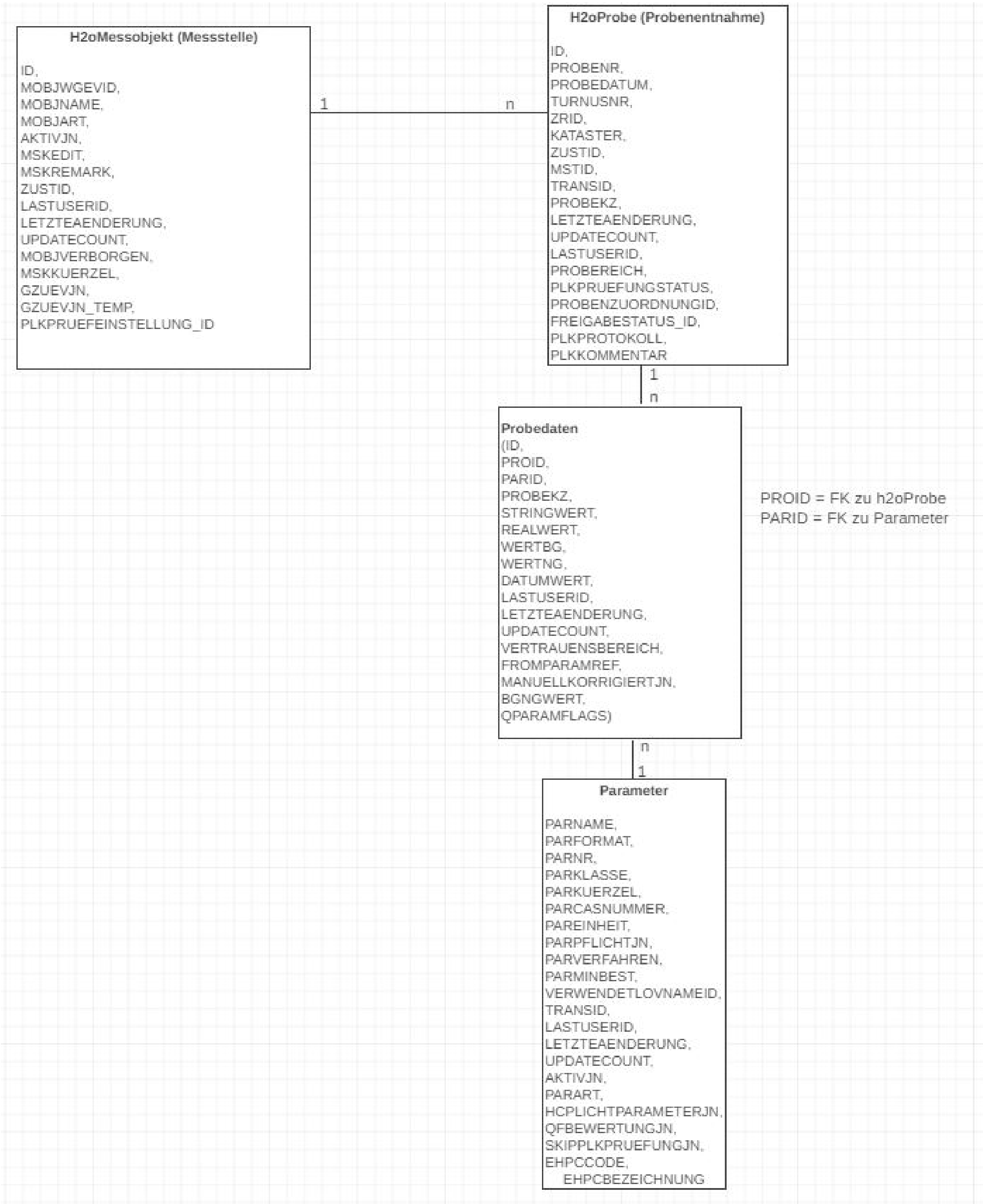


Figure 4.4.: Entity relationship diagram

4.3.5. User-interface diagram

4.3.5.1. UI01: Login page

The login page is the first page which opens when the application is requested. The user can log into the system and proceed to the main menu (UI02).

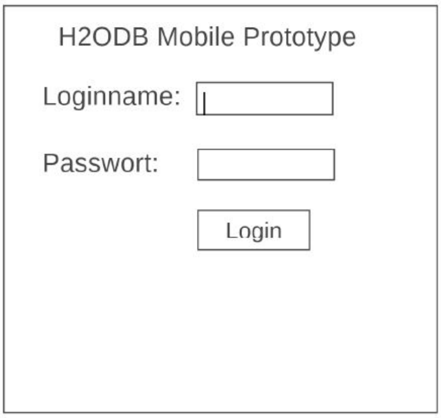


Figure 4.5.: UI01: Login page

4.3.5.2. UI02: Main menu

The main menu opens after login or when pushing the button "Home". On this page the user is welcomed and buttons to the sub-menus were shown. Moreover, other information like news, or announcing updates could also be presented on this page. Pushing the button "Log out", logs the user out and returns him to the login-page (UI01). Click on the Menu "Home" refreshes the main menu (UI02). When pressing "Probenahme" the user is directed to the sample menu (UI03). When clicking on "Daten hochladen" the user is directed to the data upload menu (UI05).

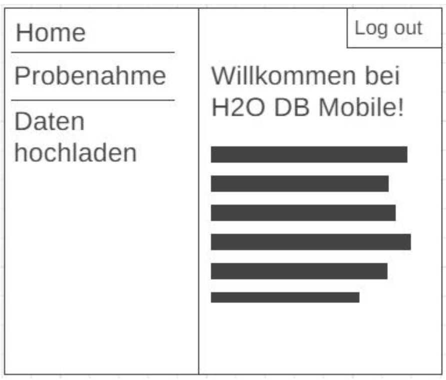


Figure 4.6.: UI02: Main menu

4.3.5.3. UI03: Measuring-points page

When pushing the button "Probenahme" in the main-menu (UI02) the measuring-points page opens. The user gets a list of all measuring points saved in the database. The list presents information about the measuring points and on the right, two buttons were displayed. Button "Proben betrachten" opens the page (UI03.1) where all samples of the specific measuring point where listed. Button "Probe hinzufügen" opens the page (UI04.2) where a new sample can be added.

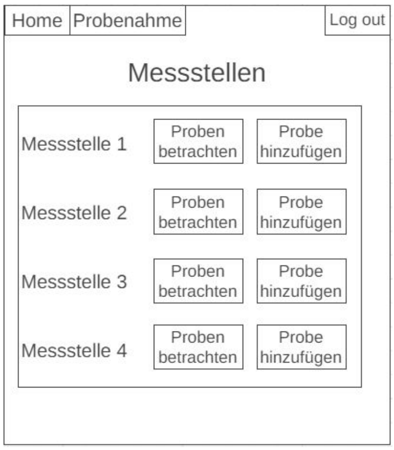


Figure 4.7.: UI03: Measuring-points page

4.3.5.3.1. UI03.1: Sample-list page

When pressing the button "Proben betrachten" in the measuring-points page a list with all available samples of the chosen measuring point were loaded. On the right of every sample row, there is a button "Probe betrachten", where the user can open the sample page (UI04) where the details of a single sample can be viewed.

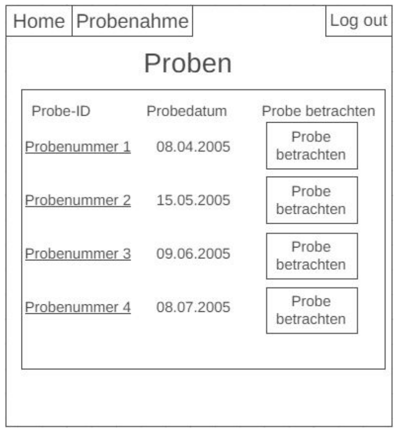


Figure 4.8.: UI03.1: Sample-list-menu

4.3.5.4. UI04: Sample page

The sample page shows all details of a specific sample. This mask can be opened when older samples are viewed or when a new sample is being added. Moreover, the water analysis can be also executed. In figure 4.9 both versions are portrayed.

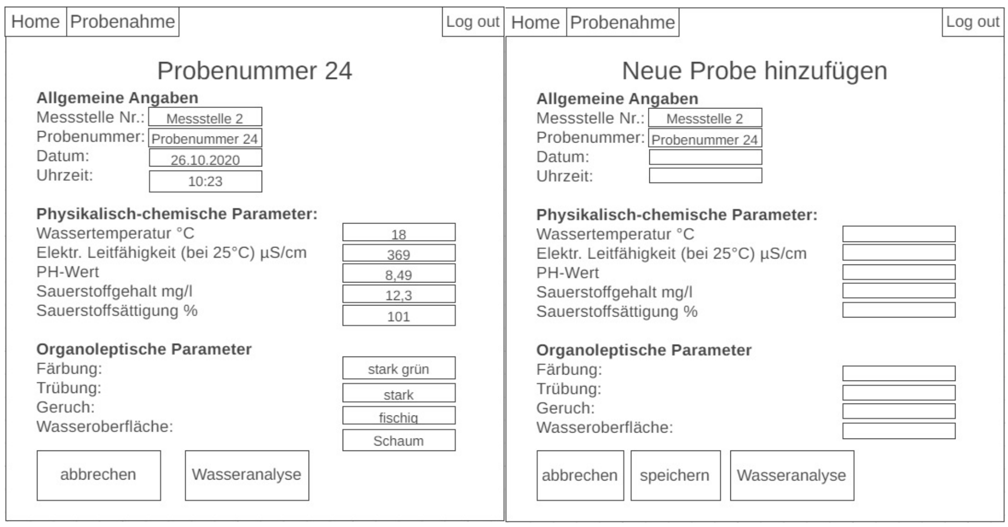


Figure 4.9.: UI04: Sample page

4.3.5.4.1. UI04.1: Sample details

This paragraph shows the details of an old sample. As the text-fields are "read-only" it is not possible to change the values. However, the user is also able to execute the water analysis.

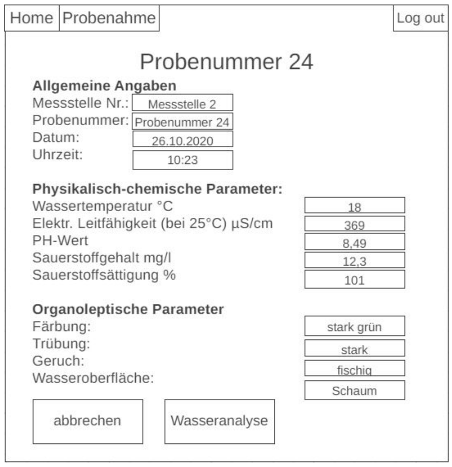


Figure 4.10.: UI04.1: Sample details

4.3.5.4.2. UI04.2: Add new sample

When adding a new sample, the same mask as in UI4.1 opens, but with the difference that the text-fields are editable and must be filled. The user can perform the water analysis and to save the entry.

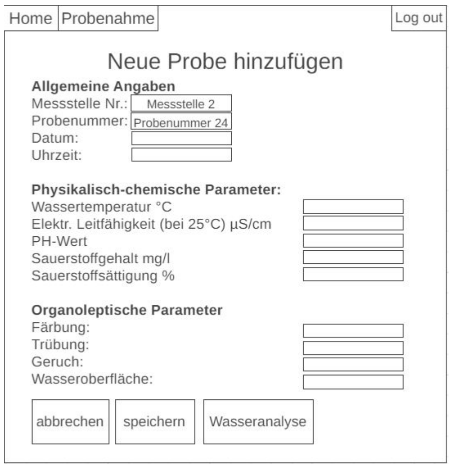


Figure 4.11.: UI04.2: Add new sample

4.3.5.4.3. UI04.3: Water analysis result page

When pressing the water analysis button, the water analysis were executed. The results of the analysis were displayed in an expanded view of the mask.

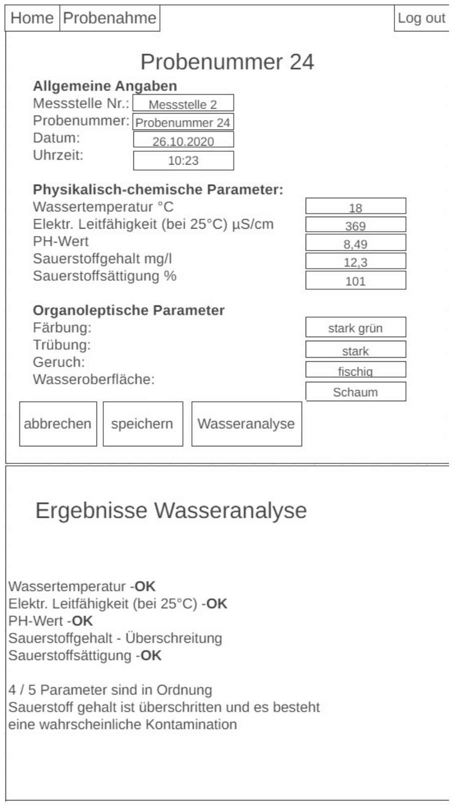


Figure 4.12.: UI04.3: Water analysis result page

4.3.5.5. UI05: Upload new data

When clicking the button "Datei auswählen" a file upload window opens where the user can select an excel file. After selecting the file, the user can upload the data per button "Upload!".

Figure 4.13.: UI05: Upload new data

4.4. Implementation Strategy

As already mentioned, it would be a benefit to implement the prototype application within the software environment of the Environment Agency Austria or at least use similar technology. Therefore, the first approach would be to investigate if an abstraction of an Environment Agency application is possible, if not than a standalone application must be implemented.
The best way to get fast to a base application is to implement a good practice example of a tutorial and reconstruct it. After setting up the technological base for the application, the model can be implemented. The database must be designed an integrated into the back-end, after that the REST-API endpoints can be defined and the GUI-masks developed. When the basic functionality like retrieving and saving data works. The water analysis and rule-based system part can be integrated into the system.

5. Prototype

The aim in this chapter is to explain the implementation of the previously described model. The developed prototype will be used to answer the research question of the master's thesis.

5.1. Applications of the Federal Environment Agency of Austria

The Federal Environment Agency of Austria provides IT solutions for different environmental fields, but also in the water pollution domain. The agency is servicing and managing the applications and the environment data, just as developing web applications and other IT solutions. The H2O Expert database and the H2O web applications is also maintained by the Agency. Due to this fact it would make sense to design the prototype application also within this environment or at least use similar technology. Many parts of the applications developed by the agency are standardizes, because of specific frameworks, style guides, process models, assisted services and other applications which were integrated.

5.2. Standalone application vs. framework application

After a failed try to abstract a web application together with developers of the Agency, I decided to create the prototype as standalone application from scratch. You could see quickly that using the Environment Agency frameworks increased the complexity of the application and the dependencies to other system. As the applications of the Agency co-operates with other services and systems, many dependencies would flow in and cause trouble. The prototype on the other side deals with a specific problem in the water analysis domain, and therefore it is not necessary to integrate the whole Environment Agency framework stack.
Even if the prototype is not integrated into the Environment Agency it would still be important that it contains similar technology. A full stack application with Java Spring Boot back-end, Rest-API and React js as front-end is a great solution.

5.3. Implementation focus

This chapter describes the focus of the prototype from a technical point of view. The requirements of the model, which were described in the previous chapter must be considered during implementation and all be realised.
Finally, by using the spiral model as a development process model, it is possible to implement the core functionality in the first cycle, and the water analysis module and the rule-based system in the following cycles.

5.3.1. Focusing on the core requirements

Timely and complete data: Because of the fact, that the prototype is a standalone application and not integrated into the Environment Agency systems. It cannot be guaranteed that the data is up to date. Moreover, the aim of the prototype is to present the functionality through certain use-cases. For example, it is not necessary that all measuring points and their sample data is loaded into the system. What really is important that several measuring points which are needed for the certain scenarios are available and that this sampling data is plausible and can be used for the statistical calculations. Finally, it is also not necessary that data is automatically updated, as the prototype has his own data source and is not directly connected with the H2O database. Therefore, saving new data into the H2O database is also not possible.

Offline compatibility: The internet coverage in the field can be very bad and it would be very useful that the system would still work. As it is not realistic that a real on-the-field experiment could be executed, this functionality is not necessary for the prototype. For future work and if the application goes in production this feature would be very important.

Integration into the systems of the environment agency: As the future for this prototype remains uncertain it is currently not a requirement to integrate the application into the systems of the environment agency.

5.3.2. Implementation environment and resulting technology choice

This subsection describes the implementation environment:

Platform independent: The application should be platform independent. Therefore, implementing the application as progressive web app in using JavaScript is probably an ideal solution.

Plausibility checks: A big part of the master's thesis was to find an appropriate statistical method. The statistical methods are the base for the plausibility checks and these checks are mandatory data for the rule-based system. Therefore, it is useful and performant to run these analysis within the system and not receive or send the data to an external service. The plausibility checks can therefore be executed on the new input data in real time and thus deliver the rule-based system directly with input data.

Rule-based system: The knowledge-based system should be integrated into the system as independent module or container. Just as real-time performance applies to the plausibility checks, it also applies to the rule-based system. The rule-based system should also be able to be invoked all the time. To be on call nonstop secures that the rule-based system can wait for input data delivered by the statistical module and thereby execute the logic reasoning. As a result the knowledge-based system can give a recommendation or action to the sample taker.

Interoperability: Interoperability between all various technologies must be present. This feature is very important for the system as the different technologies, frameworks and modules must be able to communicate with each other, especially regarding to Prolog and Java.

5.4. Technology stack

Based on the requirements of the organisation and the implementation strategy defined in the previous chapter, the following technical components were deployed:

Front-End: React.js, Node.js (npm), JavaScript - The React web framework is used as front-end technology. React is cross-platform capable and already being used in the agency for new software projects. Older projects use the JavaServer Faces Framework from Java and are normally standard desktop web applications and not mobile-able, so this technology is out of the question.

Back-End: Java Spring Boot - The back-end technology is the Java framework "Java Spring Boot", as it used everywhere within the projects of the agency. Spring Boot is a great framework to build RESTful API.

Architecture: Rest API - The architecture is be constructed as Client-server-architecture with a RESTful Web service.

Database: H2 in-memory database - The application must use a database as the data is being imported as SQL-input from the H2O Expert database. The idea was to use an easy mountable one, therefore the in-memory H2 database which is easy to integrate into Java Spring Boot is a great solution especially in early development stages. Every re-start drops the tables and imports them new into the system, as a result it is not possible to have incorrect data in the database.

Statistical method: The statistical methods are based on simple mathematics, that's why special imports or frameworks are not necessary. The selected statistical method for the plausibility check which analyse physico-chemical parameters is limited to classical statistics and uses methods such as standard deviation and variance. The frequency analysis which calculates the frequency of the measured organoleptical parameters in old data, uses SQL queries and basic mathematics.

Rule-based system: Prolog - Prolog was integrated as rule-based system by using the external library JPL. JPL is a library using the SWI-Prolog foreign interface and the Java JNI interface providing a bidirectional interface between Java and Prolog, that can be used to embed Prolog in Java. [SP]

5.5. Prototype implementation using the spiral model

The base of the prototype was created with a tutorial from the online learning and teaching platform "Udemy". The tutorial is a video tutorial to learn how to build a Java Full Stack Application. The name of the course is "Go Java Full Stack with Spring Boot and React" by the lecturer "in28Minutes Official". [Kar] The application created with this tutorial provides basic functionality like reading, displaying, saving and modifying data. Of course, the functionality from this example application is very simple and had to be extended with the features of the water analysis prototype.
As described in the previous chapter, the spiral model is the desired strategy for the implementation of the prototype. The next sub-chapters describe the cycles of the spiral model in detail.

5.5.1. Cycle 1: Prototype basic functionality

The first cycle of the spiral model was to set up the base of the prototype by modifying and expanding the functionality of the application example of the video tutorial. The various masks and the input functions were adjusted to the desired appearance. Additionally, the database entities class were created and the import data for the prototype were also added to the resources. Finally, the aim of this cycle was to implement all menus and basic functionality like readings measuring points and measuring data and save new samples.

5.5.2. Cycle 2: Water analysis

In the second cycle the prototype was extended with the statistical calculation which were added through a Java class. The results of these calculations are sent as an request to the front-end, where they are displayed to the user. The outcome of these calculations is one part of the support the prototype can provide to the sample taker and the basis for the rule-based system.

5.5.3. Cycle 3: Prolog

In the third cycle Prolog was added to the system as an external library. In order that Prolog can work a Prolog database was added too, the database is fundamental for the rule-based system. The Prolog rules are executed at the same time as the water analysis and the Prolog results are displayed together with the water results as recommendations for the sample taker.

5.5.4. Cycle 4: Additional functionality

In the fourth cycle further improvements of the system were made. To enhance the overall power of the recommendations, a method has been added to analyse the organoleptic parameters. Finally, a new menu for uploading data was also added, with this functionality it is possible to keep the data up to date and add new measuring spots and measuring data to the system.

5.6. Prototype architecture

This chapter describes the prototype architecture in detail and how all components collaborate. Figure 5.1 shows again the structure of the system.

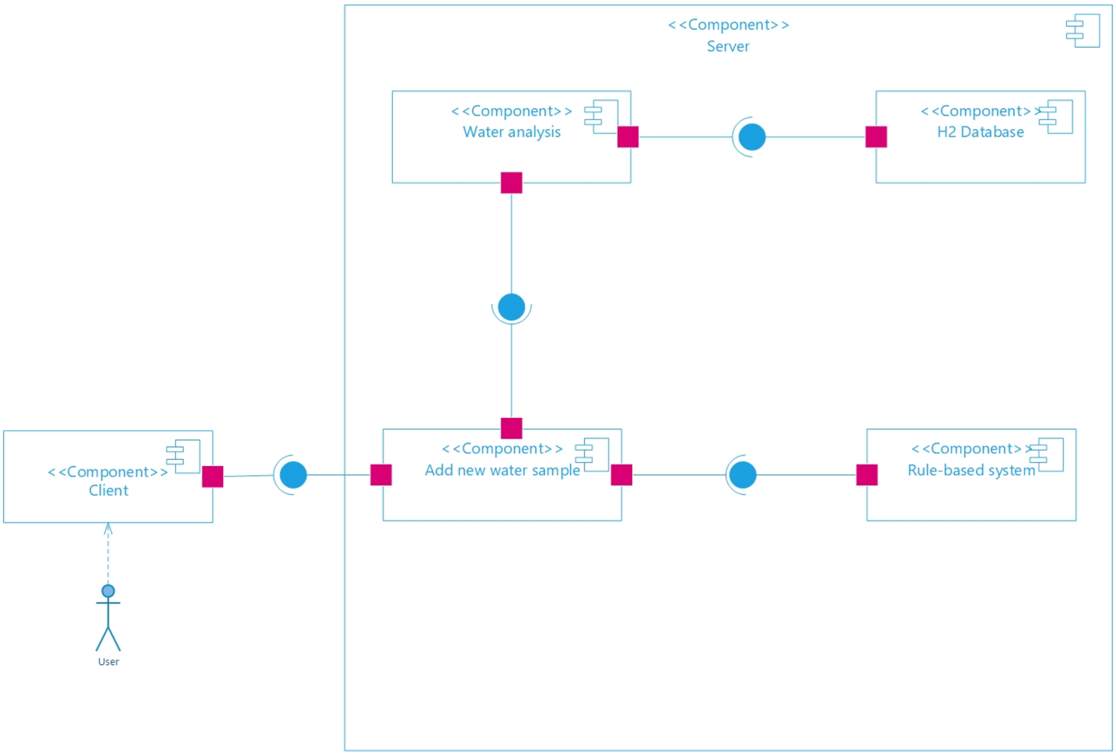


Figure 5.1.: Component diagram

5.6.1. Back-end: Java Spring Boot, H2 database and REST

The basis of the prototype is the Java Spring Boot back-end [VMwb], which operates with Maven [Foua], Hibernate [Hib], Spring Data JPA [VMwa], a H2 in-memory database [H2] and a REST API [RH]. Maven is used to build, configure, publish the application and all important configurations for dependencies and versions are configured in the "pom.xml". The database was generated through entity classes which were defined in Java. This Object Relational Mapping technique generates the whole database with its entities and relations automatically during application launch. The data import is also performed during start by executing a SQL file which contains SQL import statements. For accessing the data, JPA repositories were created for every table of the database. These repositories consist of important CRUD (Create, Read, Update, Delete) [Mar83] methods which must be executed to read or change data. The REST API is implemented in a Java class with a REST Controller where various HTTP methods with its endpoints are saved. The REST Controller is the central component for the communication between back-end and front-end, it is the contact point for sending data from back-end to front-end and vice versa. Finally, the back-end contains also classes which are important for the water analysis and recommendation system and a class where the data upload is handled.

5.6.2. Statistical method

The class "WaterAnalysisClassic" handles all calculations for the statistical part and as already mentioned, no special frameworks were included to the system, as only basic mathematical functions are used. The statistical values which are used for the plausibility check of the physico-chemical parameters are average, variance and standard deviation. The formula to calculate the lower test barrier for a parameter is (average value - test setting value * standard deviation) and for the upper test barrier (average value + test setting value * standard deviation). For the organoleptic parameters on the other side the plausibility check is not possible, the information which is here brought to the user is if this specific measured state already occurred for this water body and how often this situation occurred - Frquency analysis.

5.6.3. Prolog

Prolog needs three components to work: the Prolog library, a Prolog class and a Prolog database. The first step is integrating the external library JPL which works with the SWI-Prolog foreign interface and the Java JNI interface. It provides a bidirectional interface between Java and Prolog that can be used to embed Prolog in Java. [SP] This library was imported into the referenced libraries directory as external library and all the methods the library offers can be called by the second component the Prolog class. The Prolog class calls the Prolog methods which are necessary to connect to the Prolog database, the third important component. Through the Prolog methods, the system can write new facts into the Prolog database and also to check Prolog rules from the database, both during run-time. A connection during run-time is of course necessary as values of the parameters were delivered at the time the user uses the application. When the Prolog class is being called the delivered values are saved as facts into the database, after saving the facts the Prolog rules were checked. When the rule is true, the recommendation is returned and displayed on the screen. After that, all facts inserted into the Prolog database must be removed again, so that the Prolog database is restored to its original state, for the case when a new request is sent to the rule-based system. Keeping the old facts in the database would deliver an incorrect recommendation.

5.6.4. Front-end: React JS

React.js is used as front-end and is being launched with node.js [Foub] and npm[npm]. Node.js is a JavaScript run-time build and npm the package manager for JavaScript applications. The front-end is designed with component classes and these components are organised in the "H2oApp.jsx" class where all paths for the application are defined. For the style of the menu layouts, the Framework Bootstrap [Tea] and css is used. For the communication with the back-end the framework Axios [Sta] is used. In the service class "H2oService.js" all back-end endpoints are defined, which forward the requests over the Rest API to the back-end.

5.7. Implementation examples

This chapter presents code snippets of the most important components and describes the system in detail. The structure of this chapter for the description of the components corresponds to the order of chapter "5.6. Prototype architecture".

5.7.1. Implementation examples: Back-end: Java Spring Boot, H2 database and REST

This sub-chapter describes components and classes of the back-end in detail. The figure 5.2 shows the Package Explorer View in Eclipse IDE and how the system is built up. The directory "src/main/java" contains three packages where the main code is located. The directory "src/main/resources" is the location where important application properties and the data SQL file is saved. The "JRE System Library" is the Java Runtime Environment which is added automatically during creation and "Maven Dependencies" is the library directory where all important Maven classes and libraries are saved, and they were also added during installation. Jpl.jar is the Prolog Api which was added as external library and is stored in the directory "Referenced Libraries". Finally, the "pom.xml" is also saved in the project directory and contains all important dependencies and versions for the application.

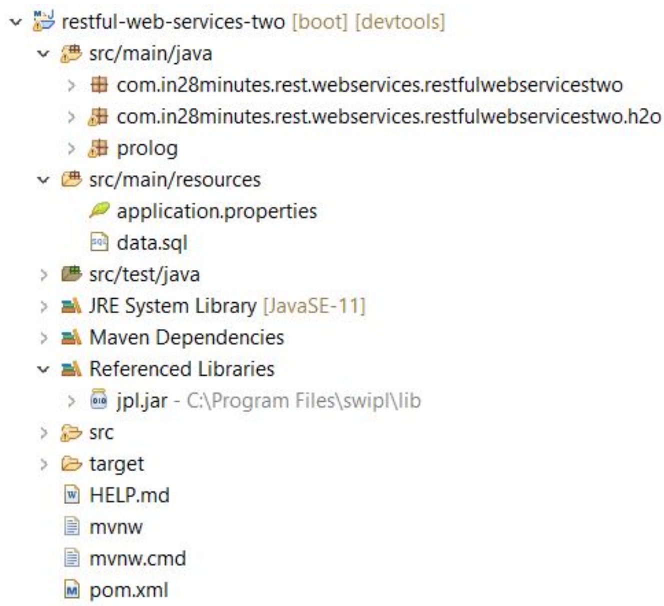


Figure 5.2.: Project structure

The figure 5.3 shows the classes which are included in the packages described before.

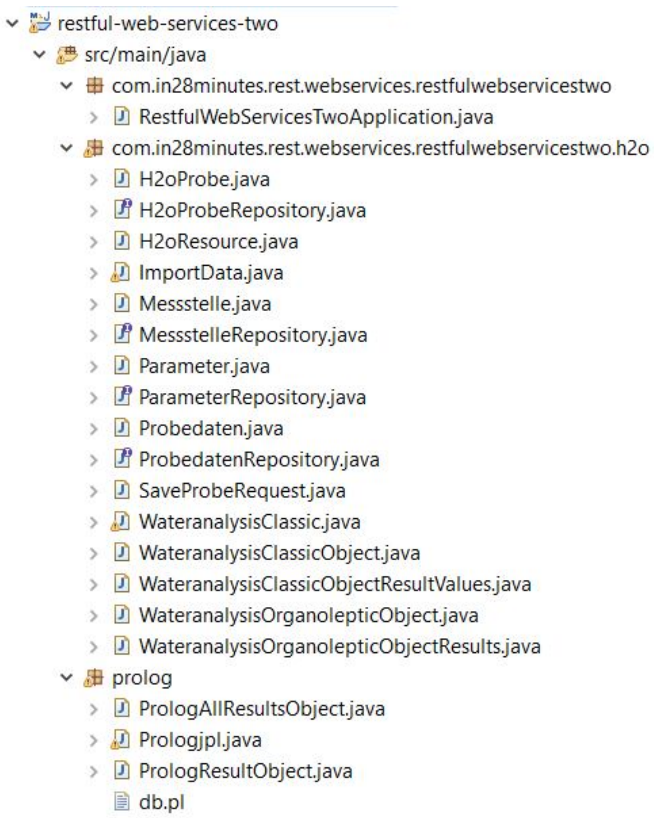


Figure 5.3.: Back-end classes

5.7.1.1. Java Spring Boot Application

The class "RestfulWebServicesTwoApplication.java" from the package "com.in28minutes.rest.webservices.restfulwe is the "void main class" which is defined and being run as an "Spring Boot Application".
The package "com.in28minutes.rest.webservices.restfulwebservicestwo.h2o" contains the database entity classes, Rest-Api (h2oResource.java) and classes which are also important for the water analysis component. (see figure 5.4)

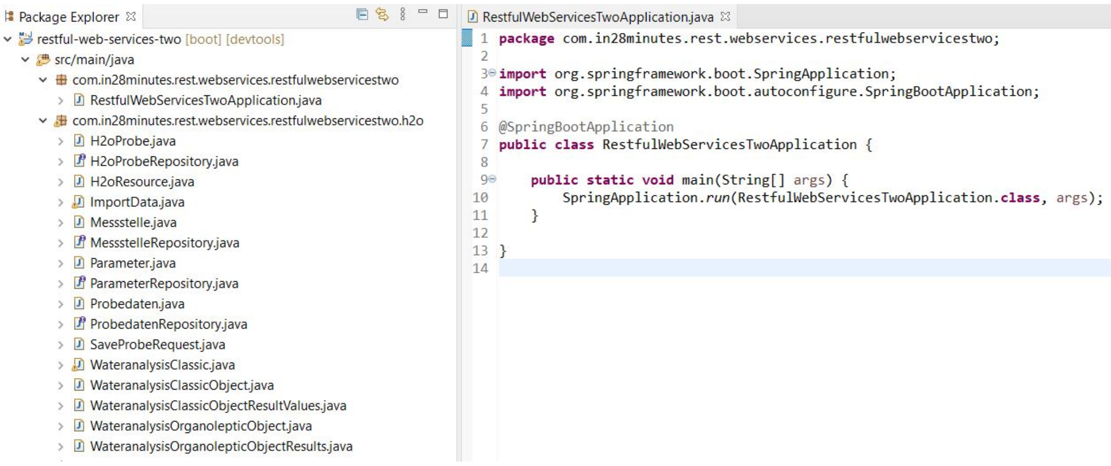


Figure 5.4.: Java Spring Boot Application

5.7.1.1.1. Maven - pom.xml In the "pom.xml" all important dependencies and information about the project is saved. (see figure 5.5) *pom.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   <modelVersion>4.0.0</modelVersion>
   <parent>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-parent</artifactId>
      <version>2.5.4</version>
      <relativePath />
      <!-- lookup parent from repository -->
   </parent>
   <groupId>com.in28minutes.rest.webservices</groupId>
   <artifactId>restful-web-services-two</artifactId>
   <version>0.0.1-SNAPSHOT</version>
   <name>restful-web-services-two</name>
   <description>Demo project for Spring Boot</description>
   <properties>
      <java.version>11</java.version>
   </properties>
   <dependencies>
      <dependency>
         <groupId>org.springframework.boot</groupId>
         <artifactId>spring-boot-starter-data-jpa</artifactId>
      </dependency>
      <dependency>
         <groupId>org.springframework.boot</groupId>
         <artifactId>spring-boot-starter-web</artifactId>
      </dependency>
      <dependency>
         <groupId>org.springframework.boot</groupId>
         <artifactId>spring-boot-devtools</artifactId>
         <scope>runtime</scope>
      </dependency>
      <dependency>
         <groupId>com.h2database</groupId>
         <artifactId>h2</artifactId>
         <scope>runtime</scope>
      </dependency>
      <dependency>
         <groupId>org.springframework.boot</groupId>
         <artifactId>spring-boot-starter-test</artifactId>
         <scope>test</scope>
      </dependency>
      <dependency>
         <groupId>org.apache.poi</groupId>
         <artifactId>poi</artifactId>
         <version>5.0.0</version>
      </dependency>
   </dependencies>
   <build>
      <plugins>
         <plugin>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-maven-plugin</artifactId>
         </plugin>
      </plugins>
   </build>
</project>
```

Figure 5.5.: pom.xml

5.7.1.1.2. H2 database - Database entity The H2 database columns are created through classes where the table structure is specified. The annotation "@Entity" from the "javax.persistence library" is used to define the database structure. Every database class includes all fields and relations the database table must have. Figure 5.6 shows the table "H2oProbe"

H2oProbe.java

```java
package com.in28minutes.rest.webservices.restfulwebservicestwo.h2o;

import java.util.Date;
import java.util.List;

import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.OneToMany;

import com.fasterxml.jackson.annotation.JsonIgnoreProperties;

@Entity(name = "h2o_probe")
public class H2oProbe {

    @Id
    public String ID;
    private String PROBENR;
    private Date PROBEDATUM;
    private String TURNUSNR;
    private String ZRID;
    private String KATASTER;
    private String ZUSTID;
    // private String MSTID;
    private String TRANSID;
    private String PROBEKZ;
    private Date LETZTEAENDERUNG;
    private Integer UPDATECOUNT;
    private String LASTUSERID;
    private String PROBEREICH;
    private String PLKPRUEFUNGSTATUS;
    private String PROBENZUORDNUNGID;
    private String FREIGABESTATUS_ID;
    private String PLKPROTOKOLL;
    private String PLKKOMMENTAR;
    private String BIOLOGIE_QUALITAETSELEMENT;

    @OneToMany(targetEntity = Probedaten.class, mappedBy = "h2oprobe")
    private List<Probedaten> probedaten;

    @ManyToOne
    @JoinColumn(name = "MSTID")
    // @JsonBackReference
    @JsonIgnoreProperties(value = { "applications", "hibernateLazyInitializer" })
    // hinzugefügt wegen fehler beim Speichern von neuer Probe:
    // https://stackoverflow.com/questions/52656517/no-serializer-found-for-class-org-hibernate-proxy-pojo-bytebuddy-bytebuddyinterc
    private Messstelle messstelle;

    protected H2oProbe() {

    }

    public H2oProbe(String ID, String PROBENR, Date PROBEDATUM, String TURNUSNR, String ZRID, String KATASTER,
            String ZUSTID, String MSTID, String TRANSID, String PROBEKZ, Date LETZTEAENDERUNG, Integer UPDATECOUNT,
            String LASTUSERID, String PROBEREICH, String PLKPRUEFUNGSTATUS, String PROBENZUORDNUNGID,
            String FREIGABESTATUS_ID, String PLKPROTOKOLL, String PLKKOMMENTAR, String BIOLOGIE_QUALITAETSELEMENT) {

        super();
```

Figure 5.6.: Database entity class: H2oProbe

5.7.1.1.3. H2 database - Repository The repositories are interfaces which provide methods for adding, updating, and retrieving data into and from a specific table. Every table has his own repository interface and depending on the needs also his own methods.

The technology used here is JPA, every interface extends from "JpaRepository" and the import for the JPA library is "org.springframework.data.jpa.repository.JpaRepository". Figure 5.7 demonstrates the repository interface for "H2oProbe". This example interface provides two methods which retrieve data from the table "H2oProbe", on by ID of a H2oProbe and one by ID of a specific measuring point.

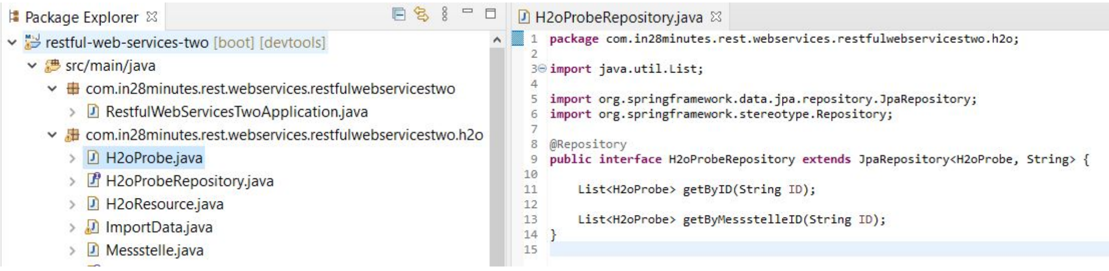


Figure 5.7.: Database repository: H2oProbe

5.7.1.1.4. H2 database - SQL data import file The H2 database is configured as an in-memory database and in order to load data into the application an import SQL file must be provided. The "data.sql file" is located in the resources package and contains SQL import statements which were executed whenever the application is restarted. The next figure presents a snippet of some insert statements.

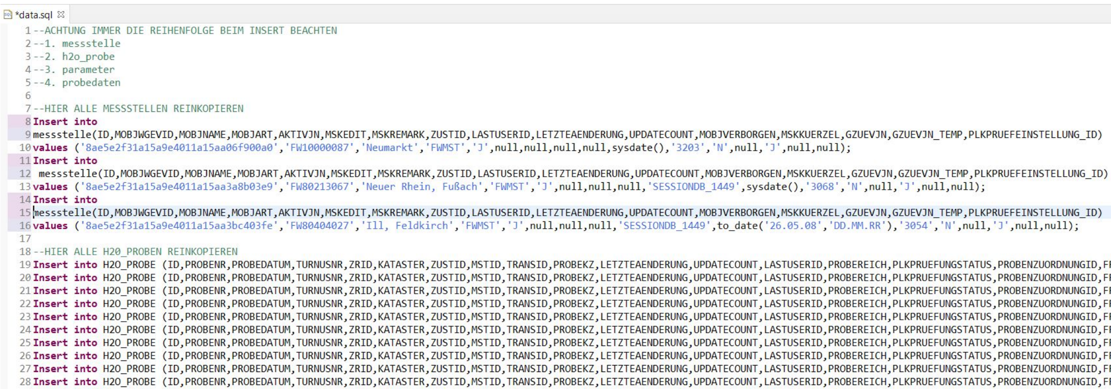


Figure 5.8.: SQL data import file

5.7.1.2. REST API

The REST API is defined in the class "H2oResource.java" and is located in the package "com.in28minutes.rest.webservices.restfulwebservicestwo.h2o". The annotation "@Rest-Controller" is essential for specifying a REST API. Figure 5.9 displays the "REST

Controller", important import statements, the initialisation of the repositories and demonstrates a back-end call for the REST operation "GET". This "GET" operation returns all measuring points in a list.

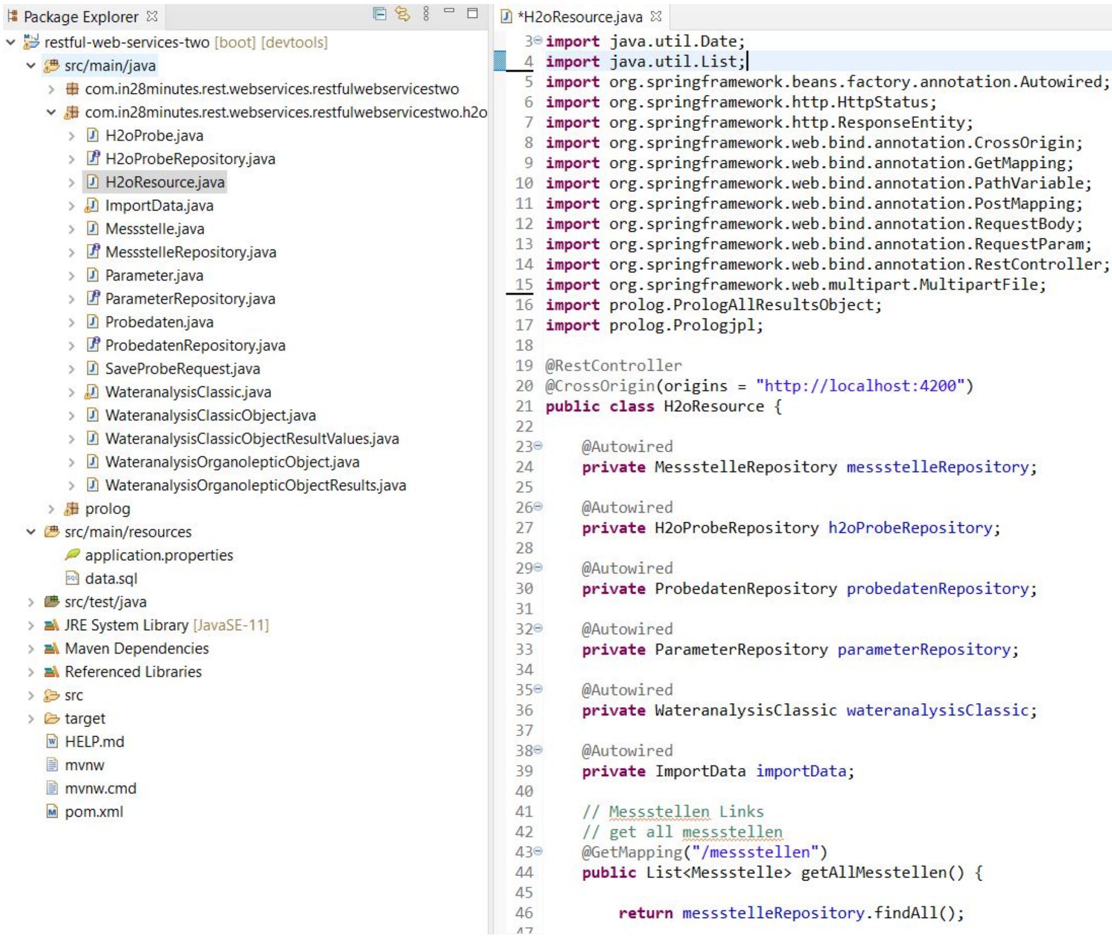


Figure 5.9.: Rest Controller

Figure (5.10) shows an example for a POST operation, in this case for saving a new measuring sample with sample data. When "addProbe" is executed, the request is supplied with a "PathVariable" which is the ID of the measurement point for which the new sampling data has to be stored. The second parameter is a "RequestBody" which is an object from the class "SaveProbeRequest.java". This object contains data of the new measuring sample (H2oProbe) and sample data (Probedaten) of every parameter measured in this measuring procedure. (see figure 5.11)

H2oResource.java

```java
    // Create Probedaten
    // createProbedaten(id, h2oProbe) {
    // return axios.post(`${API_URL}/messstellen/${id}/proben`, h2oProbe)
    // }
    @PostMapping("/messstellen/{id}/proben")
    public ResponseEntity<H2oProbe> addProbe(@PathVariable String id, @RequestBody SaveProbeRequest saveProbe) {
        // erstellen letzte Änderung für probe und für die probedaten
        saveProbe.probe.setLETZTEAENDERUNG(new Date());
        saveProbe.temperaturProbedaten.setLETZTEAENDERUNG(new Date());
        saveProbe.phwertProbedaten.setLETZTEAENDERUNG(new Date());
        saveProbe.leitfaehigkeitProbedaten.setLETZTEAENDERUNG(new Date());
        saveProbe.sauerstoffgehaltProbedaten.setLETZTEAENDERUNG(new Date());
        saveProbe.sauerstoffsaettigungProbedaten.setLETZTEAENDERUNG(new Date());

        // messstelle holen wegen Foreign Key und danach die Messtelle setzen
        Messstelle messstelle = messstelleRepository.getById(id);
        saveProbe.probe.setMessstelle(messstelle);
        H2oProbe createH2oProbe = h2oProbeRepository.save(saveProbe.probe);

        // Temperatur Parameter holen für Temperatur Foreign Key und danach die
        // parameter setzen
        Parameter temperaturParameter = parameterRepository.getById("8ae5e2f31a15860a011a1586224c0012");
        saveProbe.temperaturProbedaten.setParameter(temperaturParameter);
        probedatenRepository.save(saveProbe.temperaturProbedaten);

        // ph Parameter holen für Foreign Key und danach die parameter setzen
        Parameter phParameter = parameterRepository.getById("8ae5e2f31a15860a011a1586227b0014");
        saveProbe.phwertProbedaten.setParameter(phParameter);
        probedatenRepository.save(saveProbe.phwertProbedaten);

        // leitfähigkeit Parameter holen für Foreign Key und danach die parameter setzen
        Parameter leitfaehigkeitParameter = parameterRepository.getById("8ae5e2f31a15860a011a1586225b0013");
        saveProbe.leitfaehigkeitProbedaten.setParameter(leitfaehigkeitParameter);
        probedatenRepository.save(saveProbe.leitfaehigkeitProbedaten);

        // sauerstoffgehalt Parameter holen für Foreign Key und danach die parameter
        // setzen
        Parameter sauerstoffgehaltParameter = parameterRepository.getById("8ae5e2f31a15860a011a158623070019");
        saveProbe.sauerstoffgehaltProbedaten.setParameter(sauerstoffgehaltParameter);
        probedatenRepository.save(saveProbe.sauerstoffgehaltProbedaten);
```

Figure 5.10.: POST Operation

SaveProbeRequest.java

```java
package com.in28minutes.rest.webservices.restfulwebservicestwo.h2o;

public class SaveProbeRequest {

    H2oProbe probe;
    Probedaten temperaturProbedaten;
    Probedaten phwertProbedaten;
    Probedaten leitfaehigkeitProbedaten;
    Probedaten sauerstoffgehaltProbedaten;
    Probedaten sauerstoffsaettigungProbedaten;
    Probedaten faerbungProbedaten;
    Probedaten truebungProbedaten;
    Probedaten geruchProbedaten;
    Probedaten oelfilmProbedaten;
    Probedaten schaumbildungProbedaten;
```

Figure 5.11.: Class: SaveProbeRequest.java

5.7.1.3. Data upload

One functionality this application supplies is an additional possibility to upload data. Therefore, an excel file upload is provided for the user. To handle these excel files in the back-end, a class for interpreting excel files was created. The method reads the data from the excel and adds it into the database. For handling excel format the libraries of "org.apache.poi" were imported. (see figure 5.12)

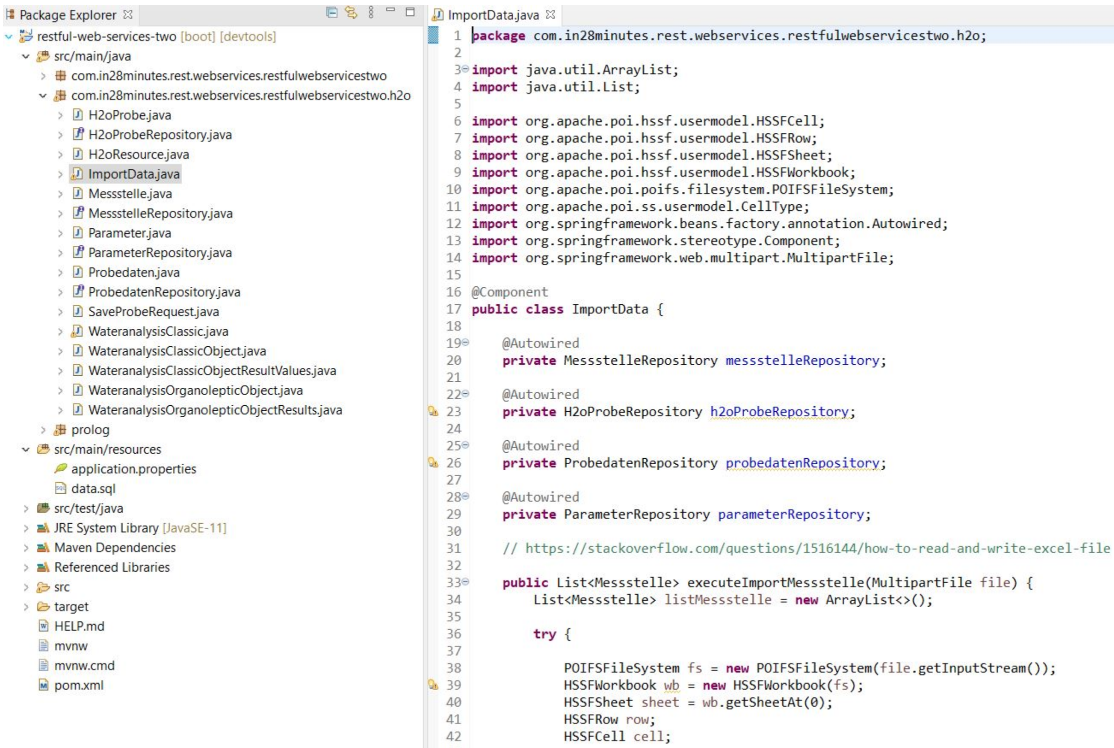


Figure 5.12.: Class: ImportData.java

When a data upload is performed by the user, the back-end call "@PostMapping("/upload")" with the method "addData(@RequestParam("file") MultipartFile file)" is executed. In this case the front-end sends the excel file to the back-end. Finally, the methods of the class "ImportData.java" where executed to get the data from the excel files. At the end the data is added to the database. (see figure 5.13)

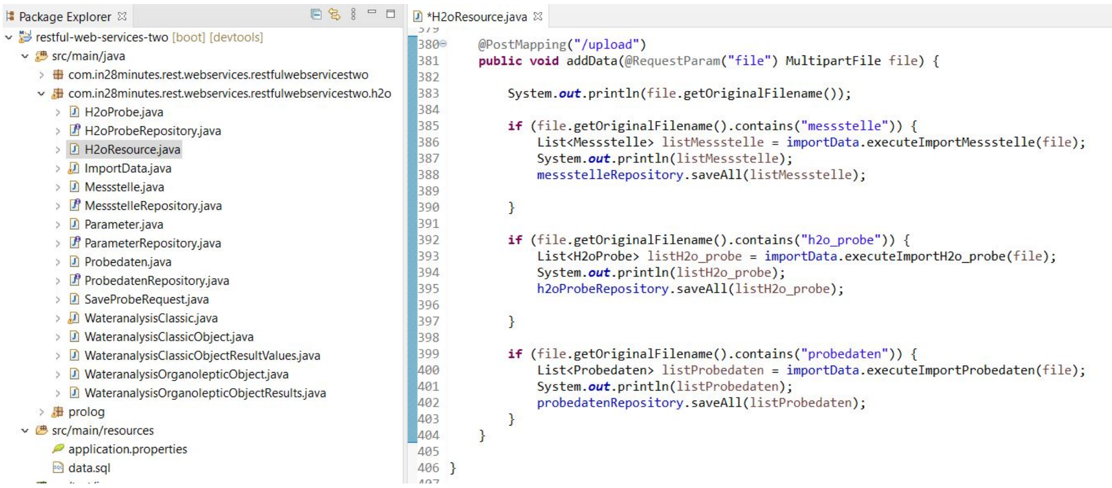


Figure 5.13.: Import data back-end call

5.7.2. Implementation examples: Statistical method

This sub-chapter describes the components of the statistical method and how they work in detail. The statistical calculations for the water analysis are located in the class "WateranalysisClassic.java" in the package "com.in28minutes.rest.webservices.restfulwebservicestwo.h2o" and contain of a bundle of statistical methods which are essential for performing the plausibility check for the on-spot parameters.

5.7.2.1. Plausibility check - Physico-chemical parameters

Calculating the test barriers for every on-spot parameter is the aim of the plausibility check. The test barriers have a lower limit and an upper limit, when the new measured value of a parameter is entered by the user, the test barriers were calculated at run-time. The system recognises if the new value exceeds one of the limits. In case the values surpass a barrier a deviation for this parameter exists. The formula to calculate the lower test barrier for a parameter is (average value - test setting value * standard deviation) and for the upper test barrier (average value + test setting value * standard deviation). Figure 5.13 shows the methods which are needed to calculate the test barriers.

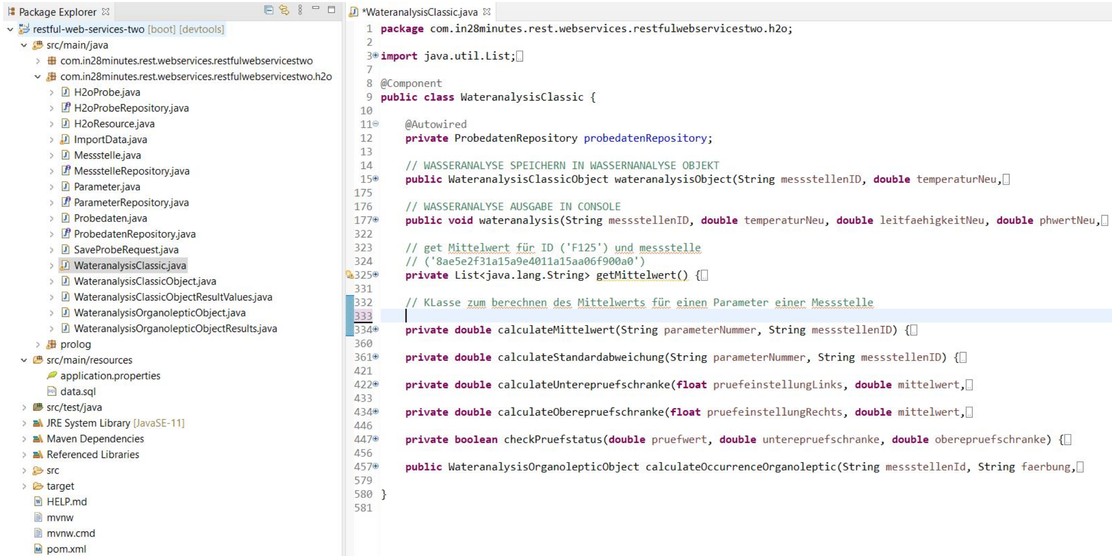


Figure 5.14.: Plausibility check

5.7.2.2. Frequency analysis - Organoleptic parameters

As already mentioned, for the organoleptic parameters it is not possible to apply the plausibility check. Therefore, the frequency analysis comes into play. The frequency analysis is supplied with subjective parameter values like smell or colour (and others) estimated by the user. The system then performs database queries to check whether these water characteristics have already occurred and how often these characteristics have occurred. Finally, the results of this analyses are provided to the user. (see figure 5.15)

*WateranalysisClassic.java

```java
    public WateranalysisOrganolepticObject calculateOccurrenceOrganoleptic(String messstellenId, String faerbung,
            String truebung, String geruch, String oelfilm, String schaumbildung) {
        // neues organoleptisches object erstellen
        WateranalysisOrganolepticObject organolepticValues = new WateranalysisOrganolepticObject();

        organolepticValues.faerbung = new WateranalysisOrganolepticObjectResults();
        organolepticValues.geruch = new WateranalysisOrganolepticObjectResults();
        organolepticValues.oelfilm = new WateranalysisOrganolepticObjectResults();
        organolepticValues.schaumbildung = new WateranalysisOrganolepticObjectResults();
        organolepticValues.truebung = new WateranalysisOrganolepticObjectResults();

        // set values
        organolepticValues.faerbung.setParameterValue(faerbung);
        organolepticValues.geruch.setParameterValue(geruch);
        organolepticValues.oelfilm.setParameterValue(oelfilm);
        organolepticValues.schaumbildung.setParameterValue(schaumbildung);
        organolepticValues.truebung.setParameterValue(truebung);

        // alle h2o proben holen für die bestimmte messstelle
// check ob Färbung für diese messstelle bereits vorgekommen ist
        List<String> listFaerbung = probedatenRepository.getByMessstelleIDAndParameternummerOrganoleptic("F115",
                messstellenId);

        // liste durchlaufen und in String speichern
        String codesFaerbung = "\n";
        codesFaerbung = codesFaerbung + "\n";
        for (int i = 0; i < listFaerbung.size(); i++) {
            codesFaerbung = codesFaerbung + listFaerbung.get(i) + "-mal" + "\n";

            System.out.println(codesFaerbung);
        }
        String codesFaerbungClean = codesFaerbung.replace(",", " = ");
        organolepticValues.faerbung.setVorgefalleneCodes(codesFaerbungClean);
        // CHeck ob der code bereits aufgetreten ist
        Integer faerbungAuftreten = probedatenRepository
                .getByMessstelleIDAndParameternummerAndStringvalueOrganoleptic("F115", messstellenId, faerbung);
        if (faerbungAuftreten > 0)
            organolepticValues.faerbung.setBereitsVorgefallen(true);
```

Figure 5.15.: Frequency analysis

5.7.2.3. Water analysis back-end call

When a new sample is created and all data has been entered by the user, the user can execute the water analysis before saving the new data. Calling this back-end endpoint executes both water analysis (plausibility check and frequency analysis) and the check of the Prolog rules. The results are saved in a big result object ("WateranalysisClassicObject.java") which carries all results of the water analysis, and this object is returned to the front-end. (see figure 5.16)

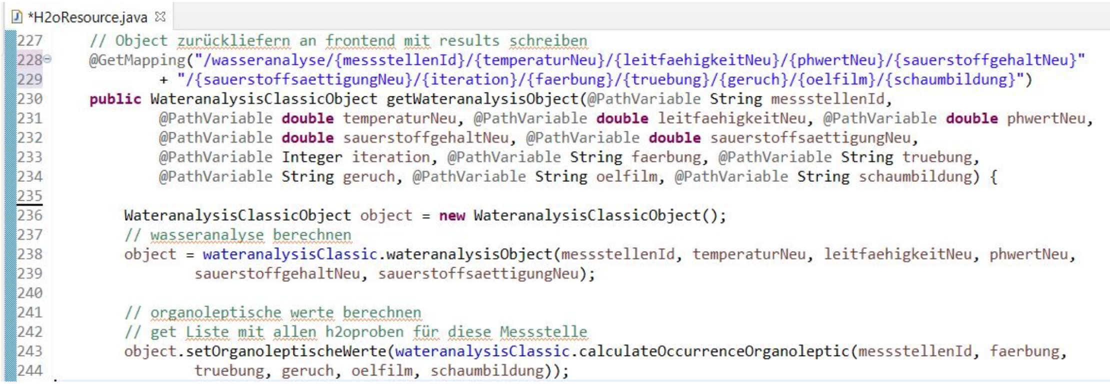


Figure 5.16.: Water analysis back-end call

5.7.3. Implementation examples: Prolog

This sub-chapter describes the components of the rule-based system Prolog and how they work in detail. In order that Prolog can work, the bidirectional API "jpl.jar" was imported as external library. The class "Prologjpl.java" located in the "Prolog package" access these JPL methods for running all Prolog operations. Figure 5.17 illustrates the class with its imports and methods.

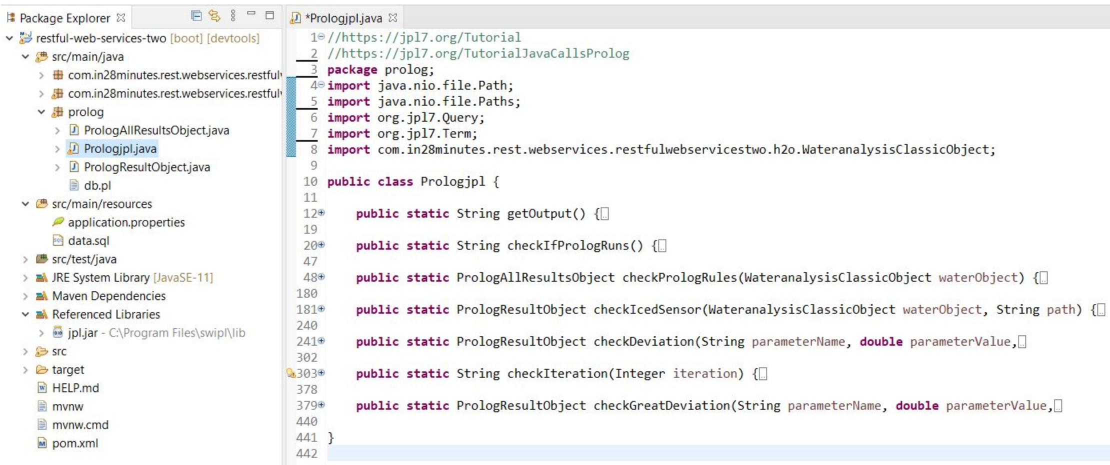


Figure 5.17.: Class: Prologjpl.java

5.7.3.1. Prolog database

The Prolog database is the core of the rule-based system. It includes all rules and facts defined for this specific topic. One rule check works in that way, that the new measured parameters values are saved as facts into the Prolog database. After that all rules can be checked, if a rule is triggered, the recommendation is returned to the user. Afterwards, the Prolog database is reset, because keeping old facts in the database would come to false determinations. Figure 5.18 presents the defined Prolog rules.

*db.pl

```prolog
:- dynamic parameter/4.
:- dynamic iteration/1.

prolog_runs(yes).

parameter(dummy, 20 , 30, 1).
iteration(0).

check_if_iced(PName, Messageice) :-
    parameter(PName, Wert),
    Wert < 0,
    Messageice = 'The system has detected a deviation, the value of the temperature parameter is in the minus range.
    The sensor of the device is iced and cannot be used for a correct measurement!
    Please defrost your sensor or change the device and repeat the measurement!'.

check_if_deviation(PName, Message) :-
    parameter(PName, Wert, Obergrenze, Untergrenze),
    (Wert =< Untergrenze; Wert >= Obergrenze),
    Message = 'The system has detected a deviation for parameter:'.

check_if_first_iteration(Iteration, Iterationmessage) :-
    iteration(Iteration),
    Iteration = 1,
    Iterationmessage = 'The system has detected a deviation,
    please check if the sampling was performed correctly and if necessary repeat the sampling!'.

check_if_second_iteration(Iteration, Iterationmessagetwo) :-
    iteration(Iteration),
    Iteration = 2,
    Iterationmessagetwo = 'The system has detected a deviation again,
    when possible please repeat the sampling at a different location! (e.g. upstream)'.

check_if_third_iteration(Iteration, Iterationmessagethree) :-
    iteration(Iteration),
    Iteration >= 3,
    Iterationmessagethree = 'The system has detected a deviation!
    If you are absolutely sure that no errors occurred during the sampling process and that you are at the right place,
    please observe the place for further peculiarities and record your observation in your report!'.

check_if_great_deviation(PName, Messagegreat) :-
    parameter(PName, Wert, Deviationover, Deviationunder),
    (Wert < Deviationunder; Wert >  Deviationover),
    Messagegreat = 'The system has detected a very high deviation, for one or more parameter values. The barrier values were surpassed significantly!
    Apparently you are using a defective device. Please check your sensor or change the measuring device. After that please repeat the measurement!'.
```

Figure 5.18.: Prolog database

5.7.3.2. Prolog execution

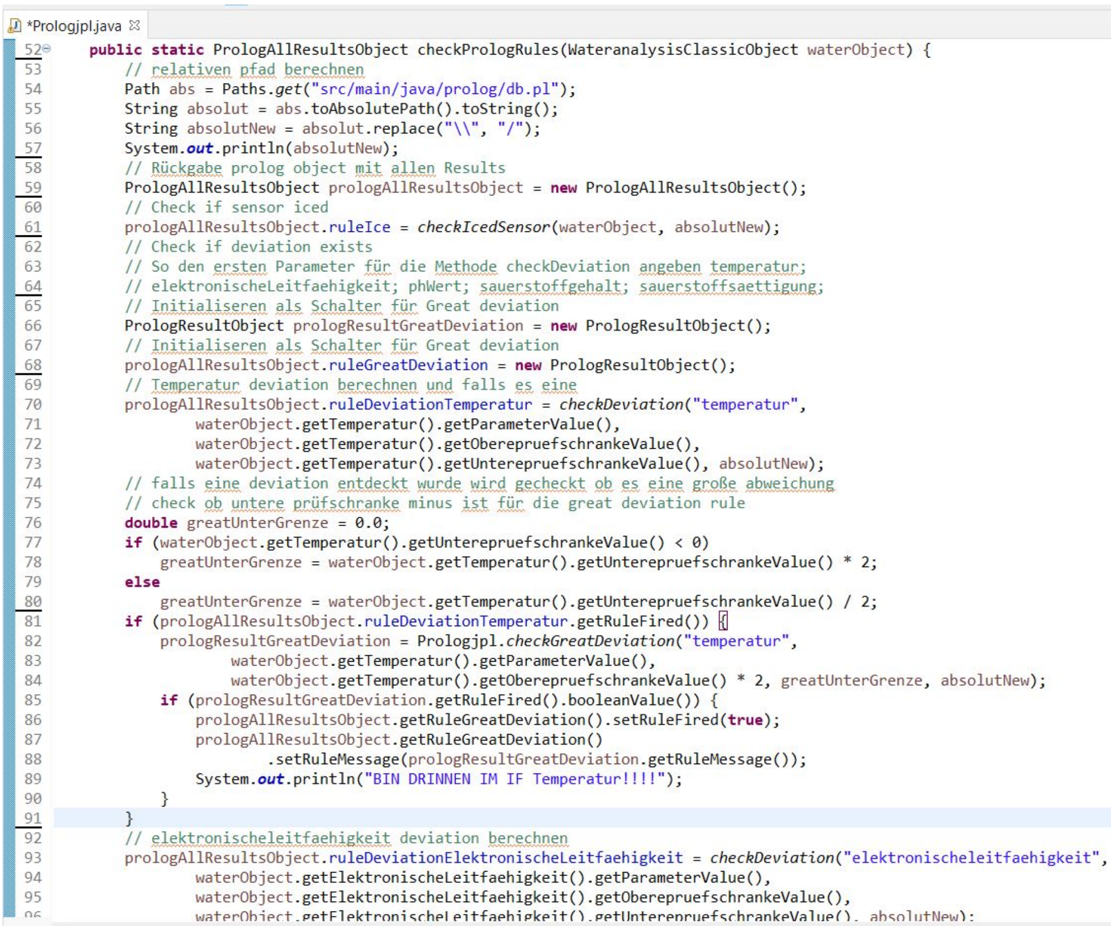
As mentioned before, the water analysis back-end call executes all statistical checks and additionally also the Prolog checks. Figure 5.19 (at the bottom) demonstrates the back-end call and the execution of the Prolog method "checkPrologRules". After running this method all Prolog rules were checked and the recommendations returned to the front-end, where they were presented to the user.

*H2oResource.java

```java
    // Object zurückliefern an frontend mit results schreiben
    @GetMapping("/wasseranalyse/{messstellenId}/{temperaturNeu}/{leitfaehigkeitNeu}/{phwertNeu}/{sauerstoffgehaltNeu}"
            + "/{sauerstoffsaettigungNeu}/{iteration}/{faerbung}/{truebung}/{geruch}/{oelfilm}/{schaumbildung}")
    public WateranalysisClassicObject getWateranalysisObject(@PathVariable String messstellenId,
            @PathVariable double temperaturNeu, @PathVariable double leitfaehigkeitNeu, @PathVariable double phwertNeu,
            @PathVariable double sauerstoffgehaltNeu, @PathVariable double sauerstoffsaettigungNeu,
            @PathVariable Integer iteration, @PathVariable String faerbung, @PathVariable String truebung,
            @PathVariable String geruch, @PathVariable String oelfilm, @PathVariable String schaumbildung) {

        WateranalysisClassicObject object = new WateranalysisClassicObject();
        // wasseranalyse berechnen
        object = wateranalysisClassic.wateranalysisObject(messstellenId, temperaturNeu, leitfaehigkeitNeu, phwertNeu,
                sauerstoffgehaltNeu, sauerstoffsaettigungNeu);

        // organoleptische werte berechnen
        // get Liste mit allen h2oproben für diese Messstelle
        object.setOrganoleptischeWerte(wateranalysisClassic.calculateOccurrenceOrganoleptic(messstellenId, faerbung,
                truebung, geruch, oelfilm, schaumbildung));

        PrologAllResultsObject prologResults = Prologjpl.checkPrologRules(object);

        String empfehlungVorhanden = "";
        String empfehlungNichtVorhanden = "The system could not detect any deviation! All parameter values are valid and can be saved!";
        boolean normalDeviationTriggered = false;
        boolean iceDeviationTriggered = false;
        boolean greatDeviationTriggered = false;

        // check ob iceRule gefeuert wurde dann prolog Boolean auf true schalten, weil
        // es eine Empfehlung gibt
        if (prologResults.getRuleIce().getRuleFired()) {
            object.setPrologResults(true);
            iceDeviationTriggered = true;
            // object.setPrologResultsMessage(prologResults.getRuleIce().getRuleMessage());
            empfehlungVorhanden = empfehlungVorhanden + prologResults.getRuleIce().getRuleMessage() + "\n";
        }

        if (prologResults.getRuleGreatDeviation().getRuleFired()) {
            object.setPrologResults(true);
            greatDeviationTriggered = true;
            empfehlungVorhanden = empfehlungVorhanden + prologResults.getRuleGreatDeviation().getRuleMessage() + "\n";
        }
```

Figure 5.19.: Prolog execution

The back-end call executes the Prolog method "checkPrologRules" of the Prolog class "Prologjpl.java". This method checks all Prolog rules one behind the other, by executing the associated Prolog method for every rule. The fired rules were saved in a result object and returned to back-end call. (see figure 5.20)



```java
*Prologjpl.java
    public static PrologAllResultsObject checkPrologRules(WateranalysisClassicObject waterObject) {
        // relativen pfad berechnen
        Path abs = Paths.get("src/main/java/prolog/db.pl");
        String absolut = abs.toAbsolutePath().toString();
        String absolutNew = absolut.replace("\\", "/");
        System.out.println(absolutNew);
        // Rückgabe prolog object mit allen Results
        PrologAllResultsObject prologAllResultsObject = new PrologAllResultsObject();
        // Check if sensor iced
        prologAllResultsObject.ruleIce = checkIcedSensor(waterObject, absolutNew);
        // Check if deviation exists
        // So den ersten Parameter für die Methode checkDeviation angeben temperatur;
        // elektronischeLeitfaehigkeit; phWert; sauerstoffgehalt; sauerstoffsaettigung;
        // Initialiseren als Schalter für Great deviation
        PrologResultObject prologResultGreatDeviation = new PrologResultObject();
        // Initialiseren als Schalter für Great deviation
        prologAllResultsObject.ruleGreatDeviation = new PrologResultObject();
        // Temperatur deviation berechnen und falls es eine
        prologAllResultsObject.ruleDeviationTemperatur = checkDeviation("temperatur",
                waterObject.getTemperatur().getParameterValue(),
                waterObject.getTemperatur().getOberepruefschrankeValue(),
                waterObject.getTemperatur().getUnterepruefschrankeValue(), absolutNew);
        // falls eine deviation entdeckt wurde wird gecheckt ob es eine große abweichung
        // check ob untere prüfschranke minus ist für die great deviation rule
        double greatUnterGrenze = 0.0;
        if (waterObject.getTemperatur().getUnterepruefschrankeValue() < 0)
            greatUnterGrenze = waterObject.getTemperatur().getUnterepruefschrankeValue() * 2;
        else
            greatUnterGrenze = waterObject.getTemperatur().getUnterepruefschrankeValue() / 2;
        if (prologAllResultsObject.ruleDeviationTemperatur.getRuleFired()) {
            prologResultGreatDeviation = Prologjpl.checkGreatDeviation("temperatur",
                    waterObject.getTemperatur().getParameterValue(),
                    waterObject.getTemperatur().getOberepruefschrankeValue() * 2, greatUnterGrenze, absolutNew);
            if (prologResultGreatDeviation.getRuleFired().booleanValue()) {
                prologAllResultsObject.getRuleGreatDeviation().setRuleFired(true);
                prologAllResultsObject.getRuleGreatDeviation()
                        .setRuleMessage(prologResultGreatDeviation.getRuleMessage());
                System.out.println("BIN DRINNEN IM IF Temperatur!!!!");
            }
        }
        // elektronischeleitfaehigkeit deviation berechnen
        prologAllResultsObject.ruleDeviationElektronischeLeitfaehigkeit = checkDeviation("elektronischeleitfaehigkeit",
                waterObject.getElektronischeLeitfaehigkeit().getParameterValue(),
                waterObject.getElektronischeLeitfaehigkeit().getOberepruefschrankeValue(),
                waterObject.getElektronischeLeitfaehigkeit().getUnterepruefschrankeValue(), absolutNew);
```

Figure 5.20.: Prolog method

5.7.4. Implementation examples: Front-end - React JS

This sub-chapter describes components and classes of the front-end in detail. The figure 5.21 demonstrates the package structure of the front-end application in the IDE Visual Studio Code. The source code of the application is in the package "src". This package contains further sub-packages "api" and "components". In package "api" the communication with the back-end is handled and in package "h2omobile" all components of the application are located. In addition, there are also several ".js" and ".css" classes, which were generated during creation of the js application. (The application was generated with the execution of the npm command "npx create-react-app h20mobile-app".) The most important classes are "index.js" where the application is loaded, and "App.js" where the "h2o component" is loaded. The style of the application is defined in "App.css" and "index.css". Furthermore, "bootstrap.css" was added as an external library, Bootstrap is the framework which is used for styling the front-end components.

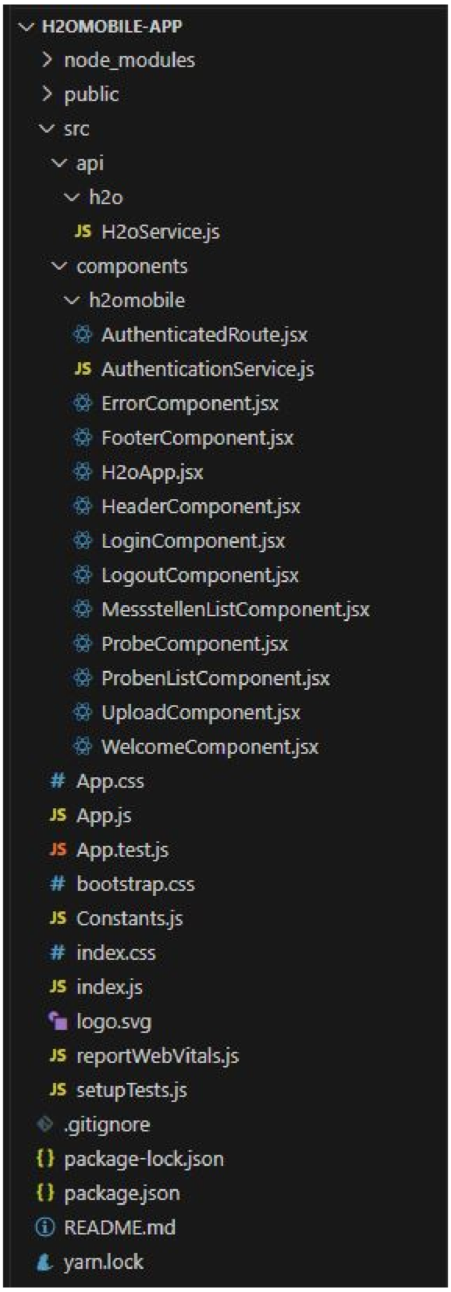


Figure 5.21.: Front-end structure

5.7.4.1. Api

This sub-chapter describes the package "API".

5.7.4.1.1. H2oService.js "H2oService.js" is located in the package "api" and is the class which handles the communication with the back-end. The back-end endpoints were called through this methods and communication is performed through the REST API. Figure 5.22 shows the back-end calls of the class.

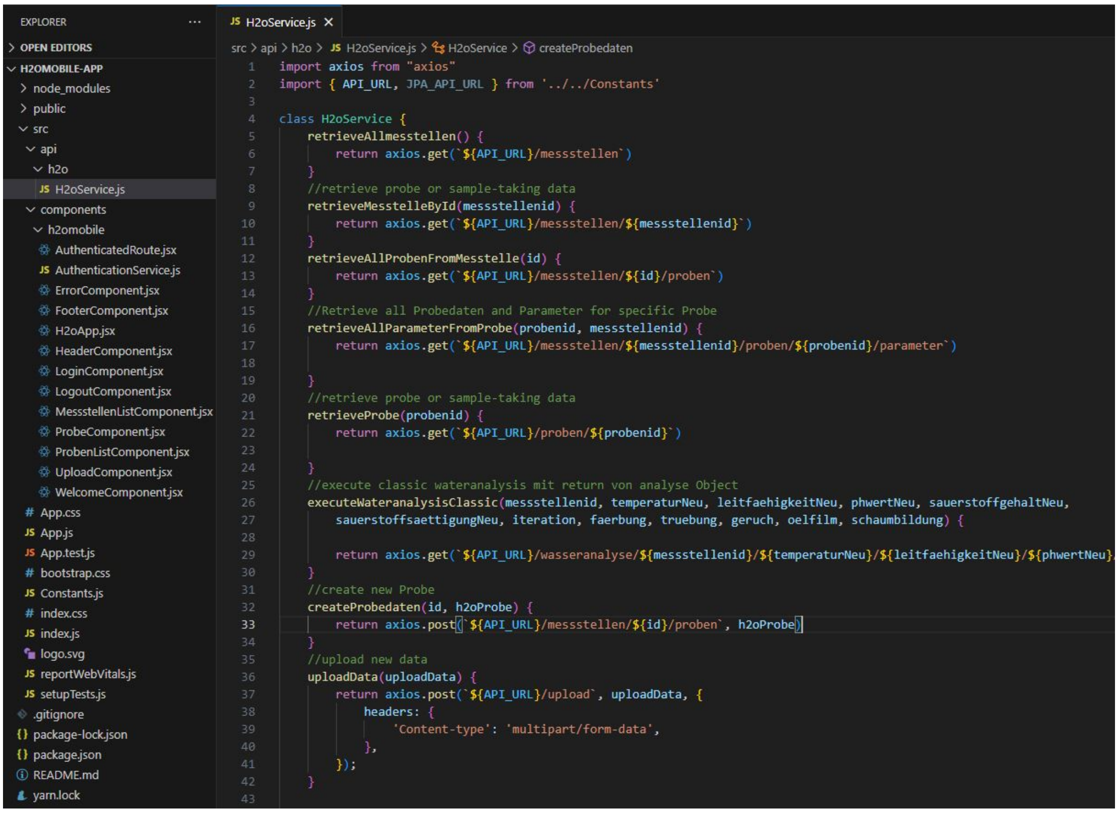
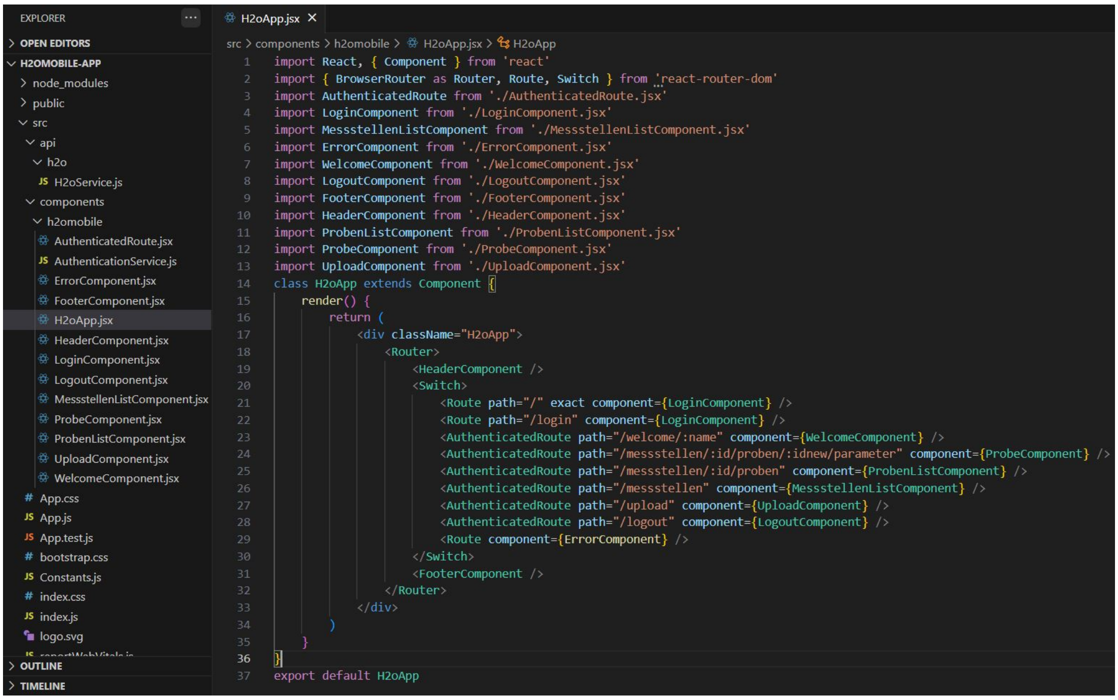


Figure 5.22.: H2oService.js

5.7.4.2. Components

This sub-chapter describes the most important components of the application.

5.7.4.2.1. H2oApp.jsx The component "H2oApp.jsx" is the heart of the application and includes the routing of the whole application. Figure 5.23 demonstrates the routes of the program. When a specific link is invoked, the routing switch opens the requested component and displays the page in the browser, for e.g., when calling the application link the component "LoginComponent" is called first. (see figure 5.24) The figure 5.25 demonstrates how the login interface looks like when compiled and loaded.



```
import React, { Component } from 'react'
import { BrowserRouter as Router, Route, Switch } from 'react-router-dom'
import AuthenticatedRoute from './AuthenticatedRoute.jsx'
import LoginComponent from './LoginComponent.jsx'
import MessstellenListComponent from './MessstellenListComponent.jsx'
import ErrorComponent from './ErrorComponent.jsx'
import WelcomeComponent from './WelcomeComponent.jsx'
import LogoutComponent from './LogoutComponent.jsx'
import FooterComponent from './FooterComponent.jsx'
import HeaderComponent from './HeaderComponent.jsx'
import ProbenListComponent from './ProbenListComponent.jsx'
import ProbeComponent from './ProbeComponent.jsx'
import UploadComponent from './UploadComponent.jsx'
class H2oApp extends Component {
    render() {
        return (
            <div className="H2oApp">
                <Router>
                    <HeaderComponent />
                    <Switch>
                        <Route path="/" exact component={LoginComponent} />
                        <Route path="/login" component={LoginComponent} />
                        <AuthenticatedRoute path="/welcome/:name" component={WelcomeComponent} />
                        <AuthenticatedRoute path="/messstellen/:id/proben/:idnew/parameter" component={ProbeComponent} />
                        <AuthenticatedRoute path="/messstellen/:id/proben" component={ProbenListComponent} />
                        <AuthenticatedRoute path="/messstellen" component={MessstellenListComponent} />
                        <AuthenticatedRoute path="/upload" component={UploadComponent} />
                        <AuthenticatedRoute path="/logout" component={LogoutComponent} />
                        <Route component={ErrorComponent} />
                    </Switch>
                    <FooterComponent />
                </Router>
            </div>
        )
    }
}
export default H2oApp
```

Figure 5.23.: H2oApp.jsx

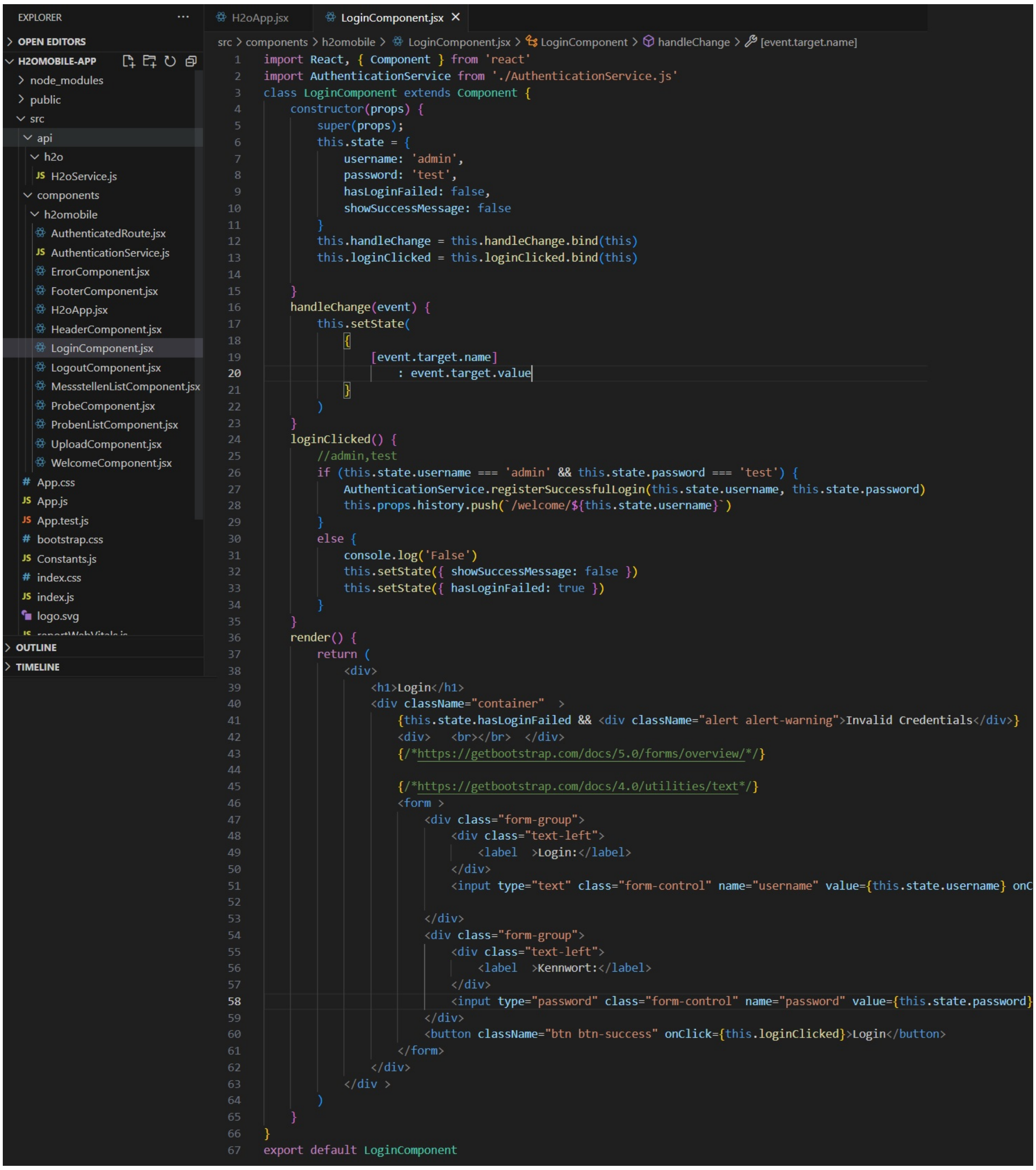


Figure 5.24.: LoginComponent.jsx

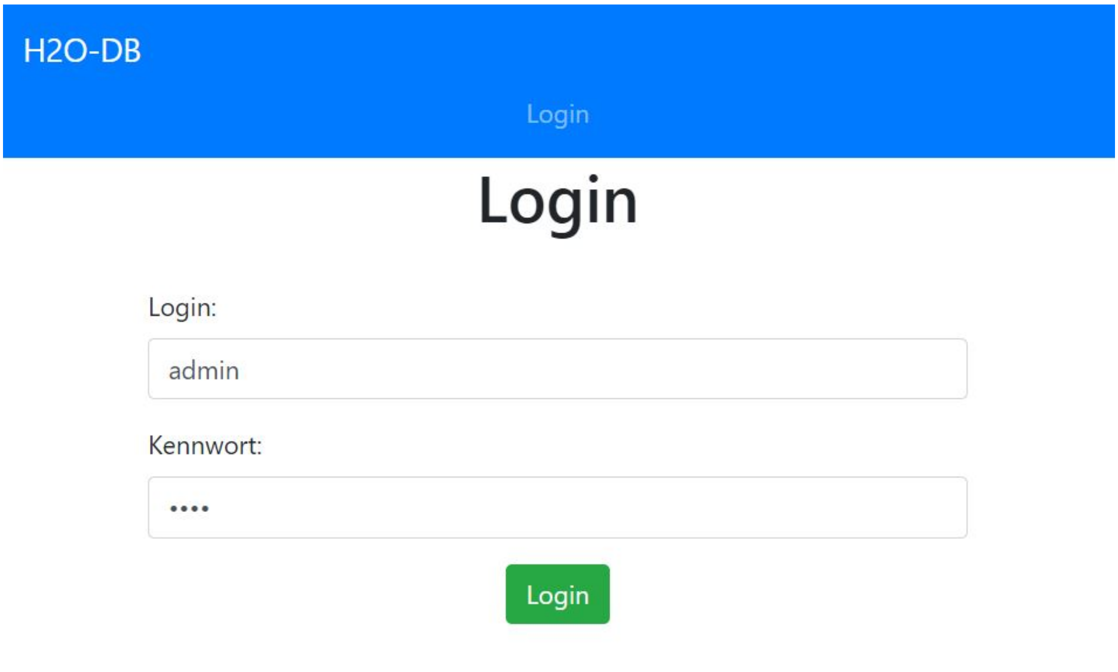


Figure 5.25.: User interface: Login

5.7.4.2.2. MessstellenListComponent.jsx Another example for an important component class is "MessstellenListComponent". This component provides a list with all measuring points and in order to add a measuring sample, the user must choose a measuring point. (see figure 5.26 and figure 5.27) Furthermore, this components contains also the back-end call "H2oService.retrieveAllmesstellen()" where a GET request is send via REST API to the back end. (see row 24 in figure 5.26 and row 5 in figure 5.22)

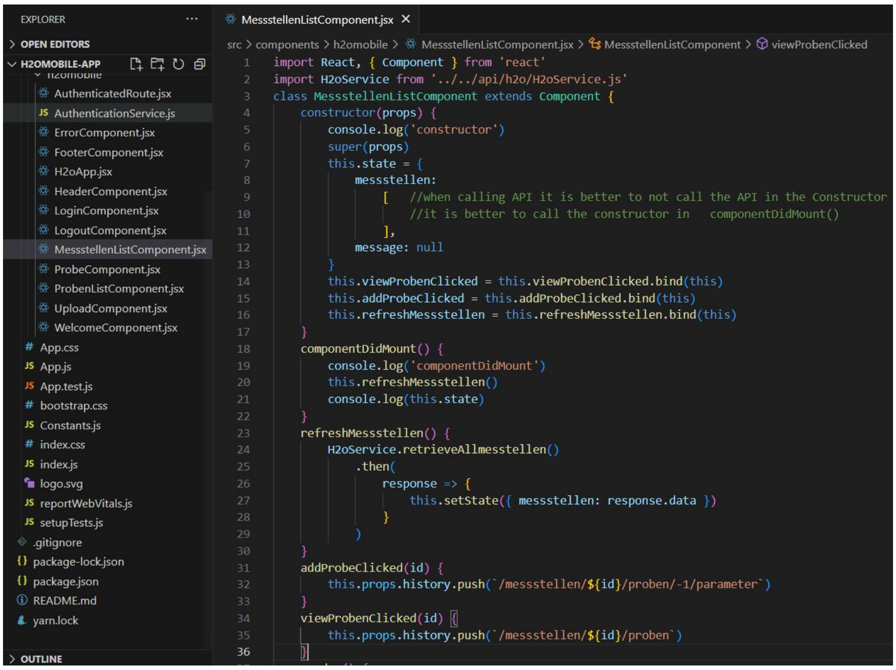


Figure 5.26.: MessstellenListComponent.jsx

Figure 5.27.: User interface: Measuring point list

5.7.4.2.3. ProbenListComponent.jsx The component "ProbenListComponent" is very similar to "MessstellenListComponent" as it includes also a GET request which retrieves all measuring samples of the chosen measuring point. (see figure 5.28)

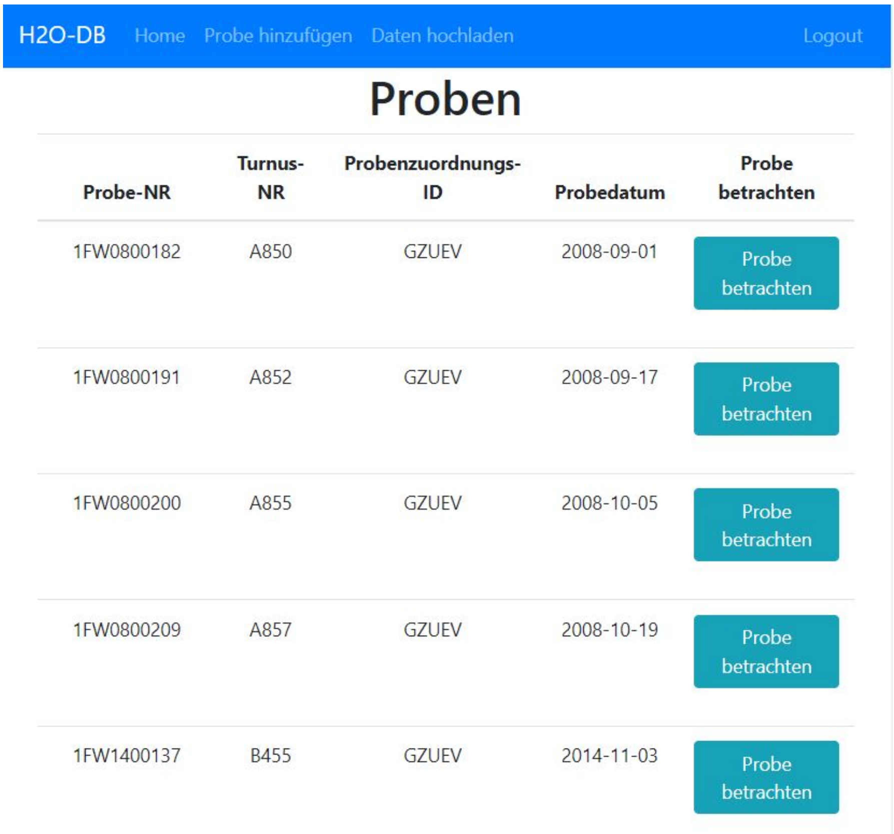


Figure 5.28.: User interface: Measuring sample list

5.7.4.2.4. ProbeComponent.jsx "ProbeComponent" is the most important and the most complex component, as it provides the main functionality of the prototype. The component has the most back-end calls and is the page with the most data operations. (see figure 5.29) The user interface enables two scenarios, one in which an old sample can be viewed, so therefore GET requests are invoked and the second scenario where a new sample can be added. After providing new sample data, two options are available. The user can start the water analysis or save the new data, both invoke back-end calls. (see figure 5.30) "createProbedaten" is a POST back-end call which saves the new data. (see row 32 in figure 5.22) "executeWateranalysisClassic" is a GET request which first sends measured data to the back-end and afterwards gets the water analysis results. (see row 26 in figure 5.22) Figure 5.31 shows how this calls are invoked in the component class.

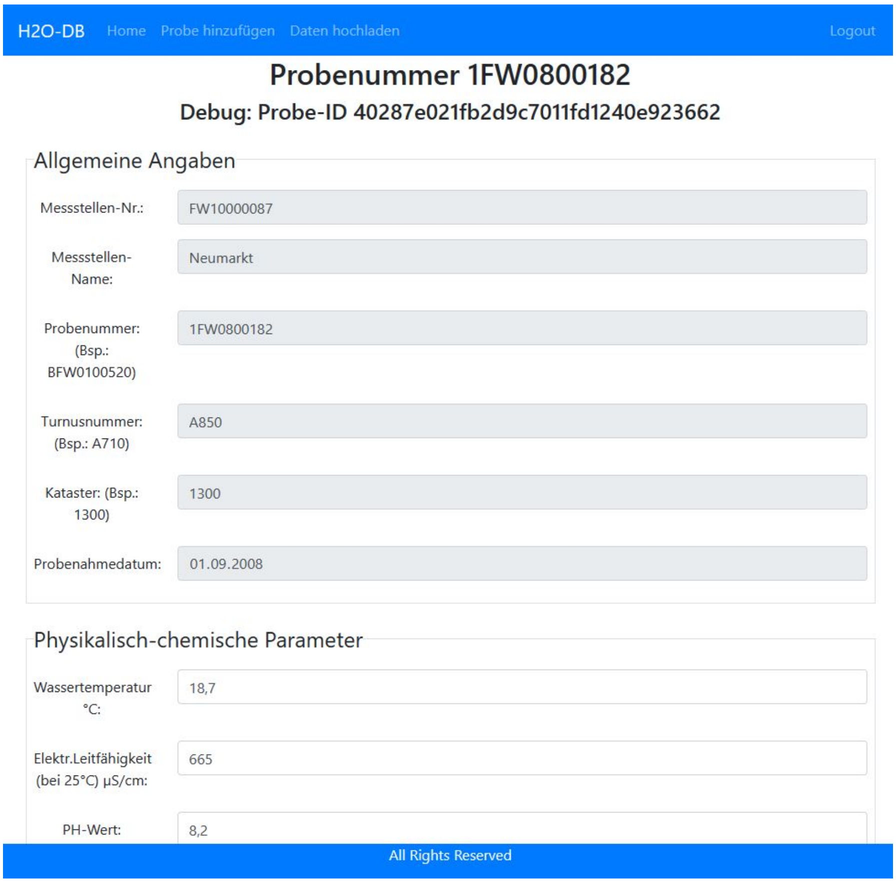


Figure 5.29.: User interface: Add new measuring sample

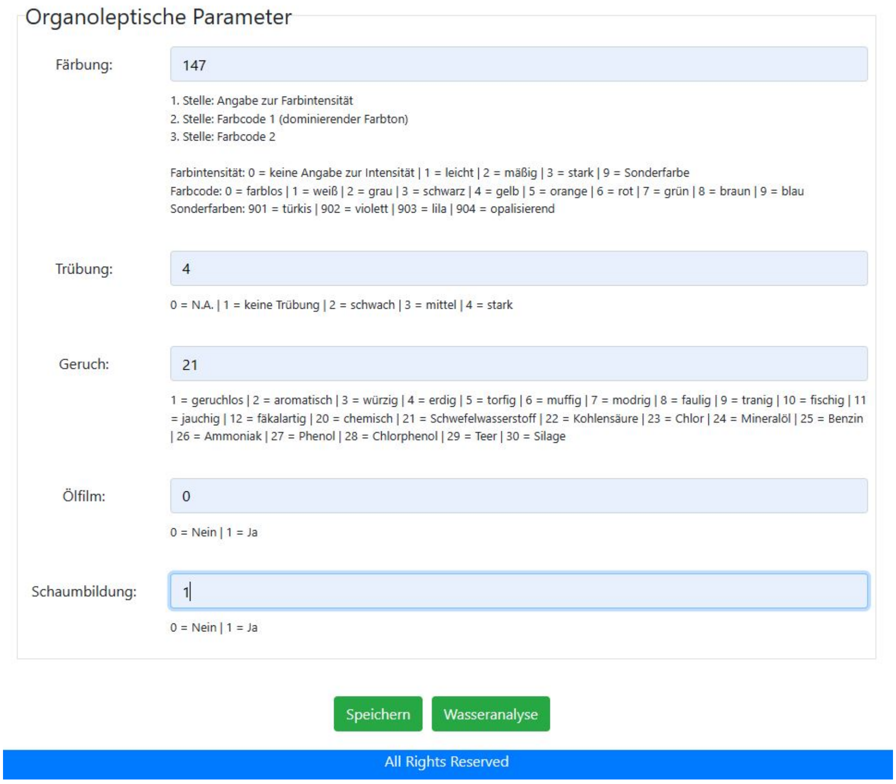


Figure 5.30.: User interface: Save or analyse new data

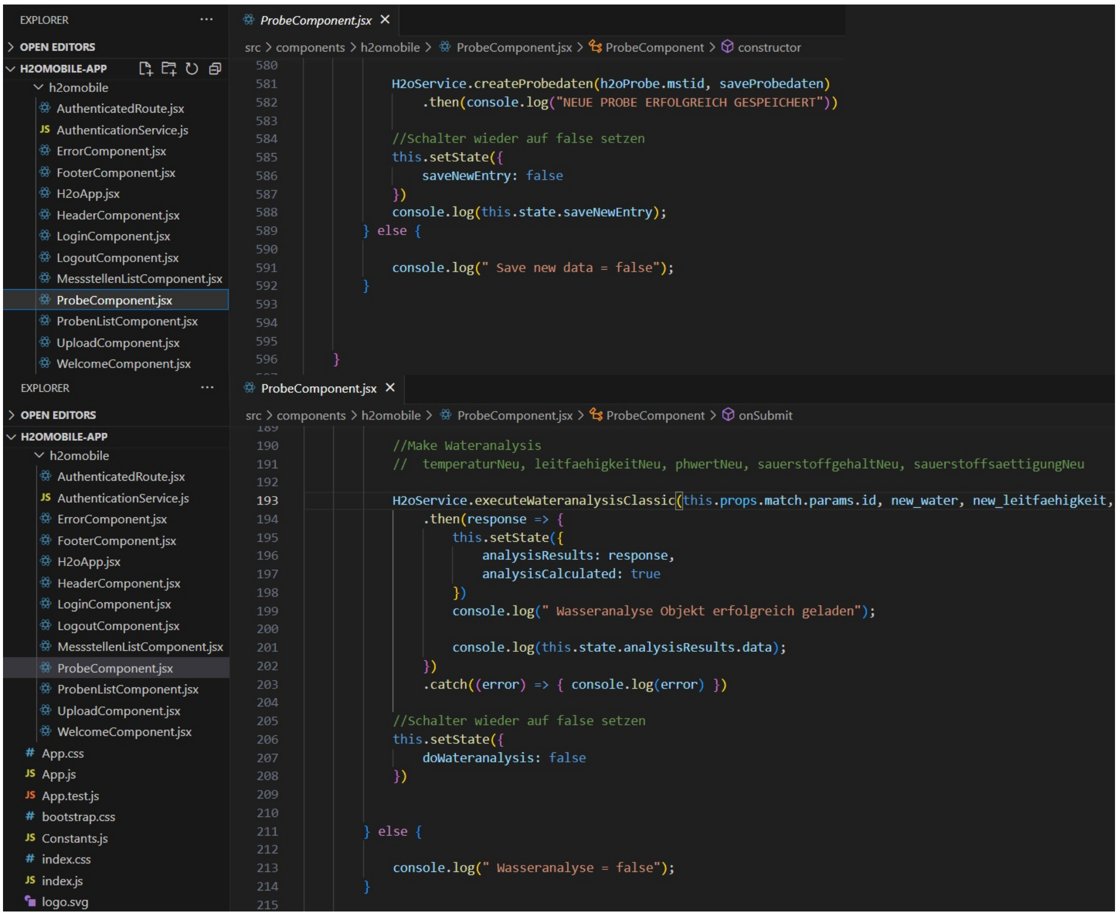


Figure 5.31.: Back-end calls in ProbeComponent.jsx

Finally, the results of the analysis are presented to the user on the screen. Figure 5.32 shows results of the plausibility check for the physico-chemical parameters and figure 5.33 results of the frequency analysis for the organoleptic parameters and recommendations for the user.

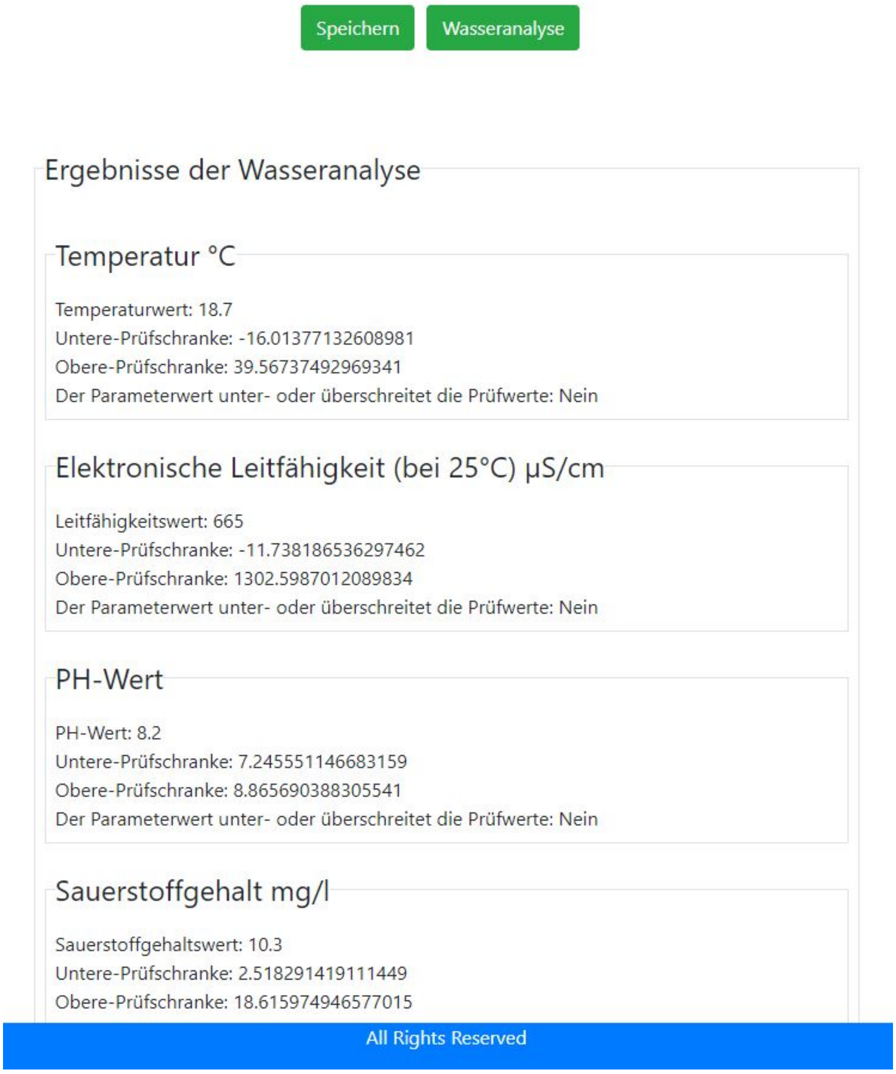


Figure 5.32.: User interface: Results for physico-chemical parameters

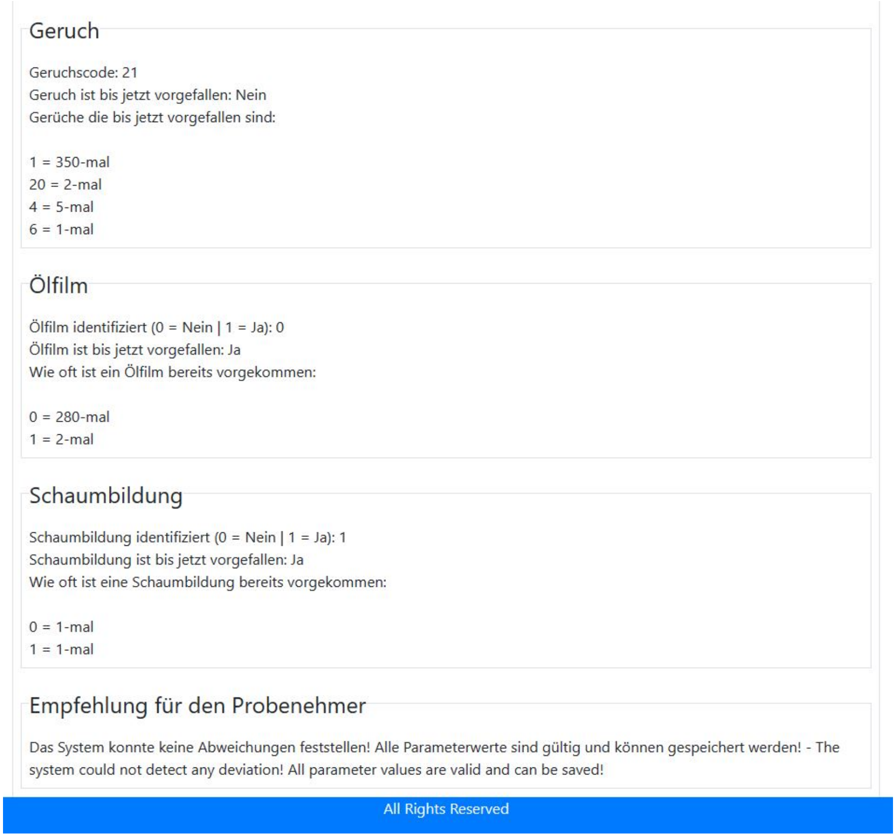


Figure 5.33.: User interface: Results for organoleptic parameters and the user recommendation

6. Test and validation

This chapter describes the testing and validation of the prototype, which is performed through a scenario-based approach. The test scenarios were executed to validate the functionality of the prototype.

6.1. Test description

The tests are divided into three types:

1. Data preparation and data upload test
2. Water analysis test
3. Mobile version test

6.1.1. Data preparation and data upload test

These tests validate basic functionality of the prototype and demonstrate that the application works properly. A flawless operation of the basic functionality is the foundation for further functional tests.

1. Basic test 1: Save new sample data
2. Basic test 2: Data upload

6.1.1.1. Basic test 1: Save new sample data

This scenario validates the correct saving of new data. The user adds new measurement data to a measurement point of his choice. Finally, the validation of this scenario is performed by checking if the new data was saved properly and can be viewed.

6.1.1.2. Basic test 2: Data upload

The data upload enables the user to upload data during run time of the application per excel files. The system can thus be updated with any data at any time. The function currently supports the upload of the tables "messstelle", "h2o_probe" and "probedaten". These tables include the data of measuring points, measuring samples and parameter values. The test scenario contains uploading the tables and checking if the new data can be viewed without errors.

6.1.2. Water analysis

The water analysis tests deal with scenarios which validate the main functionality of the prototype. The focus of these scenarios is validating the plausibility check, the frequency analysis and the recommendations of the rule-based system. Every scenario starts with the same user intention, the user adds a new measuring sample to a specific measuring point. First the user selects a measuring point of his choice. After selecting the measuring point the user creates a new sample. The next step is entering measuring values for the on-the-spot parameters and executing the water analysis. This is the point where the use-cases begin to differ from each other. The results of the water analysis can either contain deviations, which lead to a reason, or either not contain deviations which lead to the routine outcome. Within these reasons several variants can occur. Following scenarios are currently covered by the prototype:

1. Routine outcome: No deviation detected

   Routine procedure: No deviation detected - Saving new sample

2. Exception: Deviation detected

   Reason 1: Wrong sampling, wrong location, or possible contamination

   Reason 2: Faulty equipment - Iced sensor

   Reason 3: Faulty equipment - Defective device

6.1.3. Mobile version test

As the prototype was defined platform independent, a mobile test must be performed which validates that this requirement has been met. The test scenario will verify the main functionality on an Android and an iOS device.

1. Mobile test 1: Android device

2. Mobile test 2: iOS device

6.2. Test scenarios

6.2.1. Test scenarios: Data preparation and data upload

6.2.1.1. Basic test 1: Save new sample data

Test description: This scenario enables the user to add new data to a specific measuring point. The user picks a measuring point and creates a new measuring sample data for this point. After filling up the metadata of the sample and the parameter values of the measuring, the user executes the water analysis. The results of the analysis were shown on the screen and the user can save the sample. After saving, the user can open the saved sample and view the previously added data.

User story: As a user, I want to add new sample data for a specific measurement point. After filling up metadata and measured parameter values, I want to execute the water analysis and view the results of the analysis. After viewing I want to save the new sample and check if the data was inserted properly into the system.

Test steps:

1. The user logs in into the system with username "admin" and password "test".
2. The user opens open the menu "Probe hinzufügen".
3. The user can pick a measuring point by pressing the button "Probe hinzufügen".
4. The system opens a new window were metadata of the new sample and parameter values can be added.
5. The user fills up the input fields and executes the water analysis.
6. The system performs the water analysis in the back end and displays the results of the analysis on the screen.
7. The user views the results and afterwards saves the new measuring sample.
8. The user can go back to the menu "Probe hinzufügen", search for the measuring point, open the measuring samples and view the previously added data.

Expected result: The user can create a new measuring sample for a measuring point of choice and execute the water analysis. After checking the results, the user can save the new sample. Finally, the user can also check if the new sample was saved, by opening the sample from the measuring sample.

Acceptance criteria:

1. The user can create a new measuring sample for a measuring point of choice.
2. The user can fill up the input fields.
3. The user can save the new measuring sample.
4. The user can open the previously saved sample.

6.2.1.2. Basic test 2: Data upload

Test description: This scenario enables the user to upload data with an excel file, in one page via a data upload function (Attention: The file upload must be carried out in the correct order, first "messstelle", second "h2o_probe" and third "probedaten")

User story: As a user, I want to add new data with an excel file. After uploading excel files for measuring points, measuring samples and parameter data, I want to view the previously added data and add a new sample for this measuring point.

Test steps:

1. The user logs in into the system with username "admin" and password "test".
2. The user opens the menu "Daten hochladen".
3. The user pushes the button "Datei auswählen".
4. The system opens a "data upload window".
5. The user chooses in the file system the excel file for the table "messstelle".
6. The user presses the button "Upload".
7. The system inserts the data into the database.
8. The user repeats the previous steps for the tables "h2o_probe" and "probedaten".
9. The user can open the menu "Probe hinzufügen".
10. The user can see the measuring point previously added.
11. The user can open the measuring point (Button "Probe betrachten") and view all measuring samples of this point.
12. The user can open a specific measuring sample and view all parameter values.
13. The user can go back to the menu "Probe hinzufügen" and push the button "Probe hinzufügen" for this measuring point.
14. The menu for adding a new sample opens.
15. The user can fill out the fields, execute the wateranalysis, view the results and save the new data.
16. The user can go back to the samples of this measuring point and view the previously added sample.

Expected result: After successfully uploading all three excel files for the tables "messstelle", "h2o_probe" and "probedaten". The user can view data of all three tables, execute the water analysis for a new measuring sample of the new measuring point and finally save the new sample.

Acceptance criteria:

1. The user can upload all three excel files for the tables "messstelle", "h2o_probe" and "probedaten".

2. The user can view data of the new measuring point, the measuring samples, and the parameter values.

3. The user can add a new sample for the new measuring point and execute the water analysis.

4. The user can save the new measuring sample.

6.2.2. Test scenarios: Water analysis

6.2.2.1. Routine outcome: No deviation detected

Routine description: This scenario simulates the routine procedure of the sampling process where correct data is inserted into the system. The plausibility check confirms that the inserted values are within the test barriers and no deviation was detected. The system gives a recommendation that no deviation was found, and the data can be added to the system. Finally, the user can save the new sample.

6.2.2.1.1. Routine procedure: No deviation detected - Saving new sample

User story: As a user, I want to add a new sample by saving new valid and correct measuring data. As no deviation is detected and the measuring values are within the test barriers, I can save the sample.

Test steps:

1. The user logs in into the system with username "admin" and password "test".

2. The user opens the menu "Probenahme".

3. The user selects measuring point and presses the button "Probe hinzufügen".

4. The system creates a new sample where measuring data and values can be inserted into the data.

5. The user inserts plausible and correct parameter values into the system (Plausible data depends on the chosen measuring point, every measuring point has specific values. Similar values from old samples of this specific measuring point provide a good template.)

6. The user pushes the button "Wasseranalyse".

7. The system performs the water analysis in the background and provides the results on the screen. It is displayed that the parameter values are not passing over the test barriers and the data therefore is correct and valid.

8. The user scrolls down and checks the water analysis results and sees the recommendation "The system could not detect any deviation! All parameter values are valid and can be saved!"
9. The user can save the sample and go back to the main menu.

Expected result: After executing the plausibility check, the results must be shown on the screen. Every parameter value must be within the test barriers and the recommendation must state, that the values are correct and can be saved. The user can save the new sample and go back to the main menu.

Acceptance criteria:

1. The user can select a measuring point and create a new sample.
2. The user can insert measuring data and executes the water analysis.
3. The system performs the water analysis.
4. The system displays the water analysis results (Plausibility check and frequency analysis).
5. The parameter values are within the test barriers.
6. The system displays the recommendation, which states that no deviation was found and that the sample can be saved.
7. The user can save the new sample.

6.2.2.2. Exception: Deviation detected

Exceptions description: This number of scenarios simulate situations where the result contains a deviation for one or more parameters. This can occur through faulty sampling, sampling on the wrong location, faulty equipment but also because of a possible contamination. In the case of faulty equipment various variants can occur (e.g., iced sensors or defective device). In every of these scenarios recommendations suitable for this specific use-case must be provided.

6.2.2.2.1. Reason 1: Deviation detected - Wrong sampling, wrong location, or possible contamination

Reason description: This scenario simulates the situation where the result contains a deviation. The deviation can occur, because of a real deviation or because of faulty sampling. Faulty sampling are mistakes which can be made during the sample taking process or because of performing a sampling at the wrong location. In case of deviation the user must be supported and guided with recommendations for his further investigations so that he can identify if the deviation is caused by errors done during sampling or a real deviation exists. If, after fixing the problems and repeating the measurement, the deviation still exists, then it must be assumed that the deviation is correct.

User story: As a user, I want to add a new sample by inserting measuring data. In a case of a normal deviation, I want the system to support me with my further investigations, by giving recommendations about the next actions I should take. If, after repeated measurements, it turns out that the deviation still exists, it must be assumed that the deviation is correct, and the sample must be saved.

Test steps:

1. The user logs in into the system with username "admin" and password "test".
2. The user opens the menu "Probenahme".
3. The user selects measuring point and presses the button "Probe hinzufügen".
4. The system creates a new sample where measuring data and values can be inserted.
5. The user inserts implausible parameter data, which will cause a deviation (The input data should cause a slight deviation e.g.: pH-value > 10, temperature > 40 etc.).
6. The user pushes the button "Wasseranalyse".
7. The system performs the water analysis in the background and provides the results on the screen: Results of the plausibility check, frequency analysis and recommendation.
8. The user checks the results and sees that a deviation exists and that a recommendation is displayed: "The system has detected a deviation, please check if the sampling was performed correctly and if applicable repeat the sampling!".
9. The user enters similar implausible parameter data and repeats the water analysis.
10. The system repeats the analysis in the background and provides the results on the screen.
11. The user checks the results and sees that a deviation still exists and that a new recommendation (second) is displayed: "The system has detected a deviation again, when possible, please repeat the sampling at a different location! (e.g., upstream)".
12. The user enters again similar implausible parameter data and repeats the water analysis.
13. The system provides the results on the screen again.

14. The user checks the results and sees that deviation exists and that a new recommendation (third) is displayed: "The system has detected a deviation! If you are absolutely sure that no errors occurred during the sampling process and that you are at the right place, please observe the place for further peculiarities and record your observation in your report!"

15. The user can finish his investigation and save the new sample.

Expected result: After executing the water analysis for the first time, the results and the first recommendation must be shown on the screen. Minimum one parameter value must has exceeded slightly the test barriers, so that the rule-based system, gives the first recommendation (see step 8). After entering similar implausible parameter data, the second recommendation must be shown on the screen (see step 11). After entering similar implausible parameter data, the third recommendation must be shown on the screen. (see step 14) The user can finish his investigation and save the new sample.

Acceptance criteria:

1. The user can select a measuring point and create a new sample.

2. The user can insert slightly implausible measuring data three times in a row.

3. The system performs the water analysis three times in a row and displays the correct recommendations.

4. The user can finish his investigation and save the new sample.

6.2.2.2.2. Reason 2: Deviation detected - Faulty equipment - Iced sensor

Reason description: This scenario simulates the situation where the measured parameter temperature contains a negative value. This deviation shows that the sensor of the device is iced and can't be used for a correct measurement. The user must defrost the sensor or change the device.

User story: As a user, I want to add a new sample by inserting measuring data. In a case of deviation in a minus range for the temperature parameter, I want the system to support me with my further investigations, by giving recommendations about the next actions I should take.

Test steps:

1. The user logs in into the system with username "admin" and password "test".

2. The user opens the menu "Probenahme".

3. The user selects measuring point and presses the button "Probe hinzufügen".

4. The system creates a new sample where measuring data and values can be inserted.
5. The user inserts an implausible parameter value for temperature (e.g., a minus value like -10).
6. The user pushes the button "Wasseranalyse".
7. The system performs the water analysis and detects a deviation for the temperature value. The recommendation is displayed on the screen.
8. The user checks the result and sees that a deviation exists and that a recommendation is displayed: "The system has detected a deviation, the value of the temperature parameter is in the minus range. The sensor of the device is iced and can't be used for a correct measurement! Please defrost your sensor or change the device and repeat the measurement!".
9. The user must repeat the measurement (Routine procedure).

Expected result: After executing the water analyis, the results and the recommendation must be shown on the screen. The result for the temperature parameter should display that the parameter has undershot the lower barrier and that a deviation is present. Finally, the rule-based system should recommend in this case, that the deviation is caused because of faulty equipment and that the sensor of the device is iced and must be defrosted.

Acceptance criteria:

1. The user can select a measuring point and create a new sample.
2. The user can insert a negative value for the temperature parameter and start the water-analysis.
3. The system performs the water analysis and shows the results on the screen.
4. The system recommends, that the deviation is caused because of faulty equipment and that the sensor of the device is iced and must be defrosted.

6.2.2.2.3. Reason 3: Deviation detected - Faulty equipment - Defective device

Reason description: This scenario simulates the situation where the result contains a very high deviation because of an incorrect measurement of a faulty device. This can occur when the measuring device or the sensors are generally defective.

User story: As a user, I want to add a new sample by inserting measuring data. In a case of very high deviation, I want the system to support me with my further investigations, by giving recommendations about the next actions I should take.

Test steps:

1. The user logs in into the system with username "admin" and password "test".
2. The user opens the menu "Probenahme".
3. The user selects measuring point and presses the button "Probe hinzufügen".
4. The system creates a new sample where measuring data and values can be inserted.
5. The user inserts a very high implausible parameter value for a parameter (e.g., a temperature value over 100 or a pH-value over 20).
6. The user pushes the button "Wasseranalyse".
7. The system performs the water analysis and detects a very high deviation. The recommendation is displayed on the screen.
8. The user checks the results and sees that a deviation exists and that a recommendation is displayed: "The system has detected a very high deviation, for one or more parameter values. The barrier values were surpassed significantly! Apparently, you are using a defective device. Please check your sensor or change the measuring device. After that please repeat the measurement!"
9. The user must repeat the measurement (Routine procedure).

Expected result: After executing the water analysis, the recommendation must be shown on the screen. The system should recommend in this case, that the deviation is caused because of defective device and that the user should check the sensor of the device or change the device.

Acceptance criteria:

1. The user can select a measuring point and create a new sample.
2. The user can insert a very high value for the parameters and start the water analysis.
3. The system performs the water analysis and shows the recommendation on the screen.
4. The system recommends, that the deviation is very high, caused because of faulty equipment and that the sensor or the device is defective and must be checked or changed.

6.2.3. Test scenarios: Mobile version

Test description: The aim of these tests is to verify the main functionality of the prototype on an Android and an iOS device. "Basic test 1: Save new sample data" (chapter 6.2.1.1.) and "Basic test 2: Data upload" (chapter 6.1.1.2.) are the scenarios used for the tests, and they must be executed on mobile devices.

6.2.3.1. Mobile test 1: Save new sample data - Android device

Since this mobile scenario for Android corresponds completely to the scenario "Basic test 1: Save new sample data" (chapter 6.2.1.1.) and "Basic test 2: Data upload" (chapter 6.1.1.2.), the test description is omitted and only the acceptance criteria are illustrated in this chapter. For performing the tests, a different execution order of the scenarios is used. The test will start with "Basic test 2: Data upload" (chapter 6.1.1.2.) an end with "Basic test 1: Save new sample data" (chapter 6.2.1.1.). Executing the tests in this order can cover the main functionality in one extensive system test.

Acceptance criteria:

1. The user can call the application on his Android device.
2. The user can upload all three excel files for the tables "messstelle", "h2o_probe" and "probedaten".
3. The user can view data of the new measuring point, the samples, and the parameter values.
4. The user can create a new sample for the previously added measuring point.
5. The user can fill up the input fields.
6. The user can save the new sample.
7. The user can open the previously saved sample.
8. The mobile version of the application is presented without visualization errors. All elements are readable and all fields which should be editable can be edited.
9. The functionality of the prototype can be executed faultless. (Performing the water analysis, reading the results and saving the sample)

6.2.3.2. Mobile test 2: Save new sample data - iOS device

Since this mobile scenario for iOS corresponds completely to the scenario "Basic test 1: Save new sample data" (chapter 6.2.1.1.) and "Basic test 2: Data upload" (chapter 6.1.1.2.), the test description is omitted and only the acceptance criteria are illustrated in this chapter. For performing the tests, a different execution order of the scenarios is used. The test will start with "Basic test 2: Data upload" (chapter 6.1.1.2.) an end with "Basic test 1: Save new sample data" (chapter 6.2.1.1.). Executing the tests in this order can cover the main functionality in one extensive system test.

Acceptance criteria:

1. The user can call the application on his iOS device.

2. The user can upload all three excel files for the tables "messstelle", "h2o_probe" and "probedaten".

3. The user can view data of the new measuring point, the samples, and the parameter values.

4. The user can create a new sample for the previously added measuring point.

5. The user can fill up the input fields.

6. The user can save the new sample.

7. The user can open the previously saved sample.

8. The mobile version of the application is presented without visualization errors. All elements are readable and all fields which should be editable can be edited.

9. The functionality of the prototype can be executed faultless. (Performing the water analysis, reading the results, and saving the sample)

6.3. Test execution

This sub-chapter describes the execution of the test scenarios and presents the test results.

6.3.1. Summary: Test execution

This summary gives an overview of passed test scenarios.

<u>Execution status:</u>

- Data preparation and data upload
  - **Passed** - Basic test 1: Save new sample data
  - **Passed** - Basic test 2: Data upload
- Water analysis
  - **Passed** - Routine procedure: No deviation detected - Saving new sample
  - **Passed** - Reason 1: Wrong sampling, wrong location, or possible contamination
  - **Passed** - Reason 2: Faulty equipment - Iced sensor
  - **Passed** - Reason 3: Faulty equipment - Defective device
- Mobile version test
  - **Passed** - Mobile test 1: Android device
  - **Passed** - Mobile test 2: iOS device

6.3.1.1. Test execution - Basic test 1: Save new sample data

Test execution: The test was passed because all the requirements specified below were satisfied.

Acceptance criteria:

1. The user can create a new sample for a measuring point of choice.
2. The user can fill up the input fields.
3. The user can save the new sample.
4. The user can open the previously saved sample.

The next paragraph illustrates results of the execution with important screenshots.

6.3.1.1.1. Test execution results: The figure 6.1 shows the "Adding new sample" page. The input fields are editable. Figure 6.2 demonstrates the execution of the water analysis and the results of the analysis. Figure 6.3 illustrates the newly added sample in the sample list, and in the second picture also reading the data of the sample.

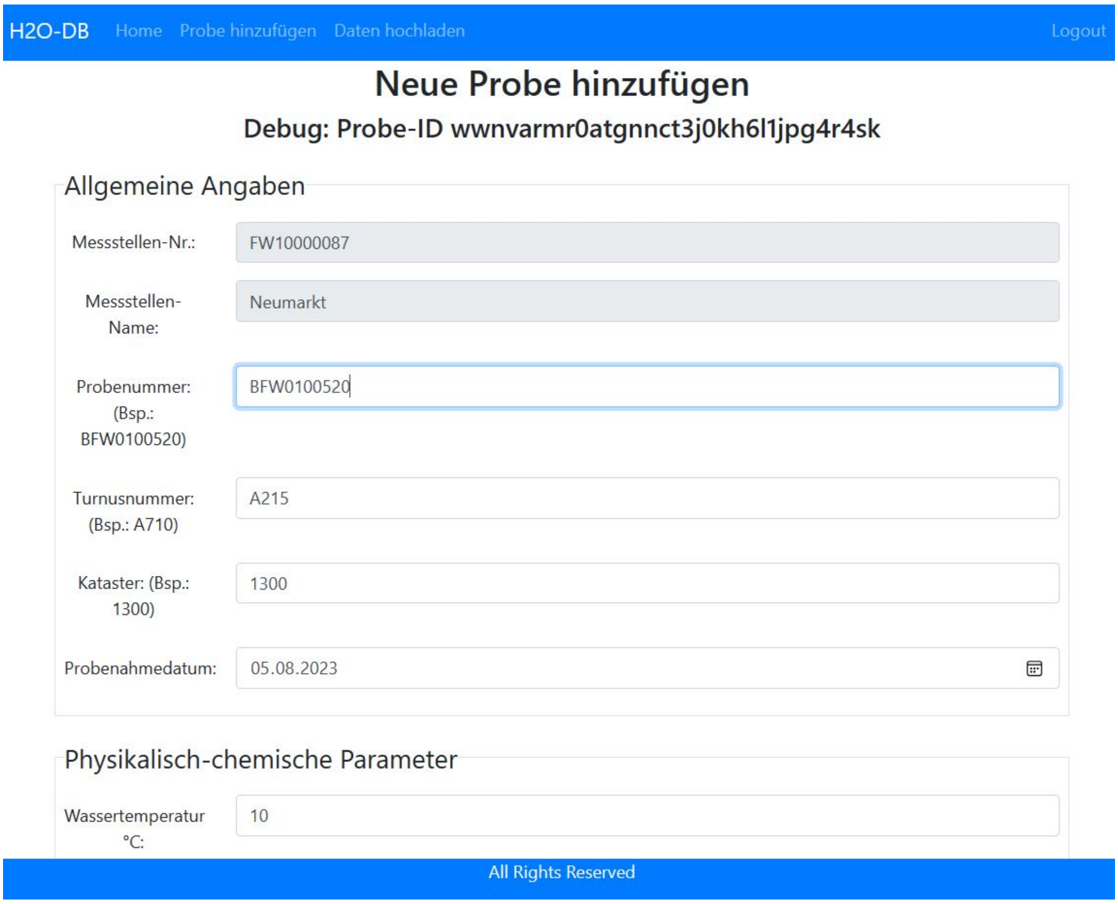


Figure 6.1.: Basic test 1: Creating new sample and data input

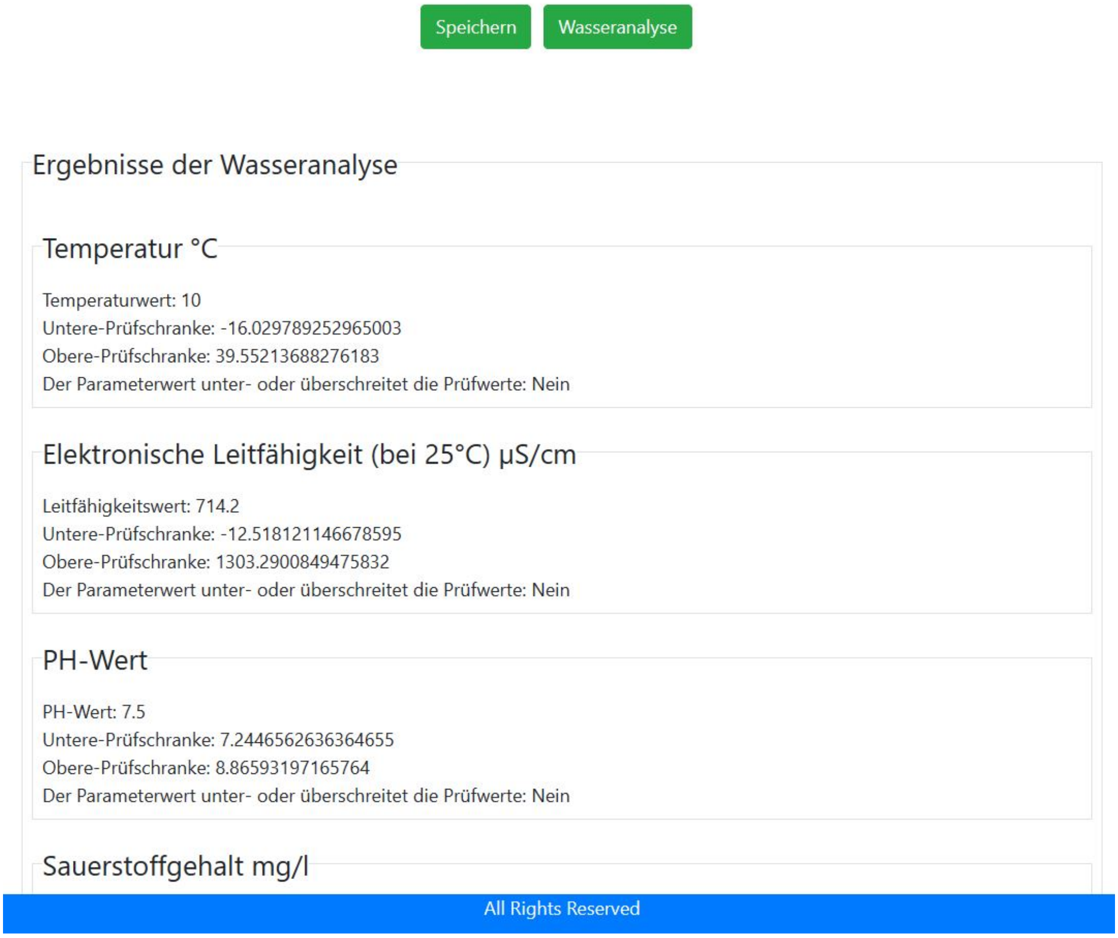


Figure 6.2.: Basic test 1: View results of water analysis

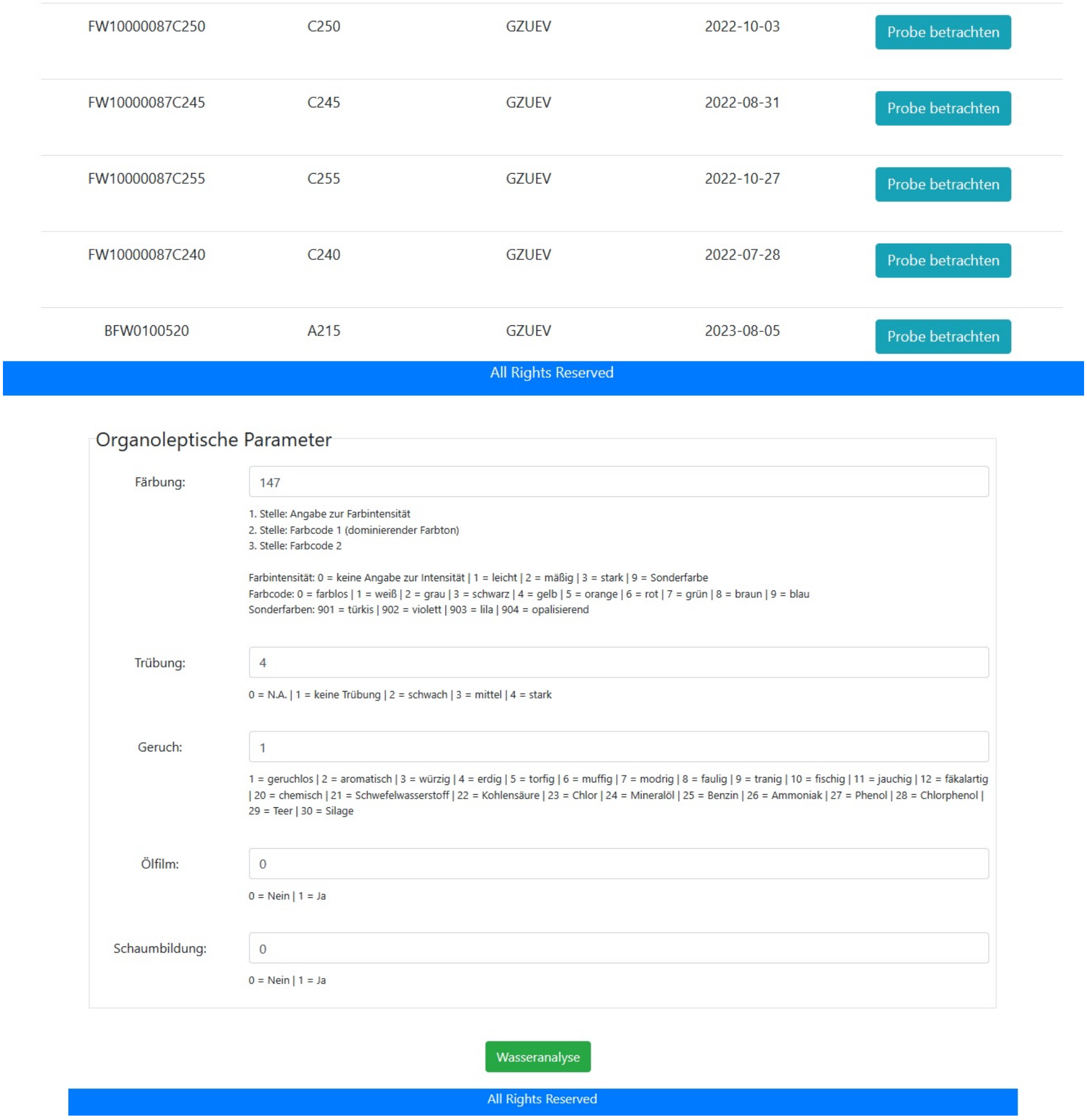


Figure 6.3.: Basic test 1: View new sample

6.3.1.2. Test execution - Basic test 2: Data upload

Test execution: The test was passed because all the requirements specified below were satisfied.

Acceptance criteria:

1. The user can upload all three excel files for the tables "messstelle", "h2o_probe" and "probedaten".
2. The user can view data of the new measuring point, the samples, and the parameter values.
3. The user can add a new sample for the new measuring point and execute the water analysis.
4. The user can save the new sample.

The next paragraph illustrates results of the execution with important screenshots.

6.3.1.2.1. Test execution results: The figure 6.4 shows the "Data upload" mask, where all three excel files were uploaded. The next three figures (6.5, 6.6 and 6.7) illustrate that the data of the new measuring point can be viewed. Figures 6.8 and 6.9 demonstrate that creating, analysing, and saving a new sample for the new measuring point is possible.

Figure 6.4.: Basic test 2: Data upload mask

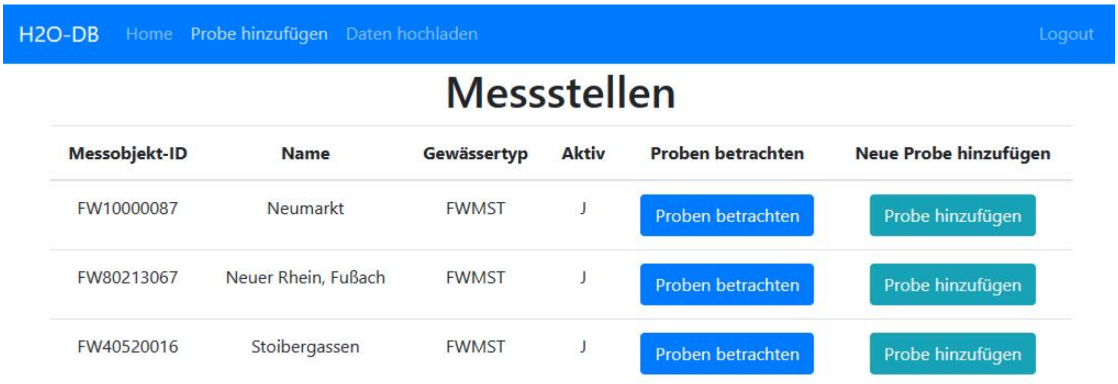


Figure 6.5.: Basic test 2: View new measuring point

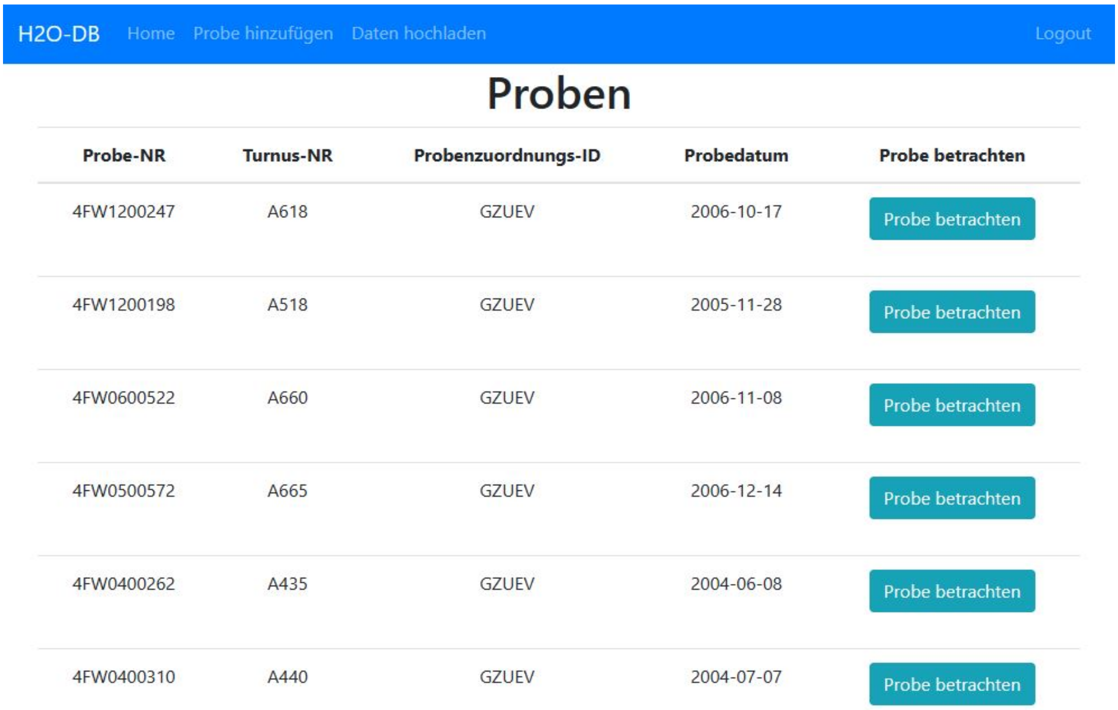


Figure 6.6.: Basic test 2: View measuring sample list

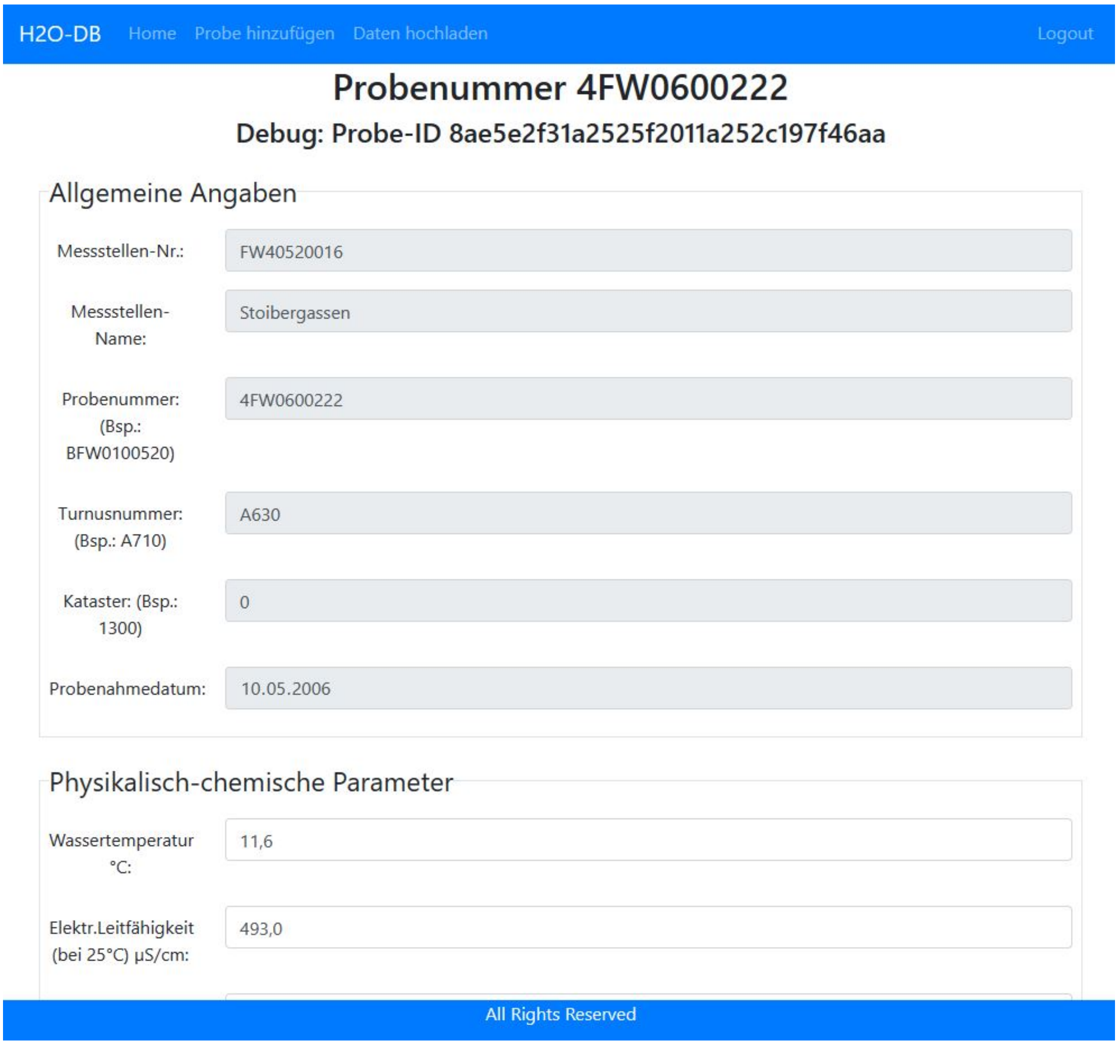


Figure 6.7.: Basic test 2: View data of sample

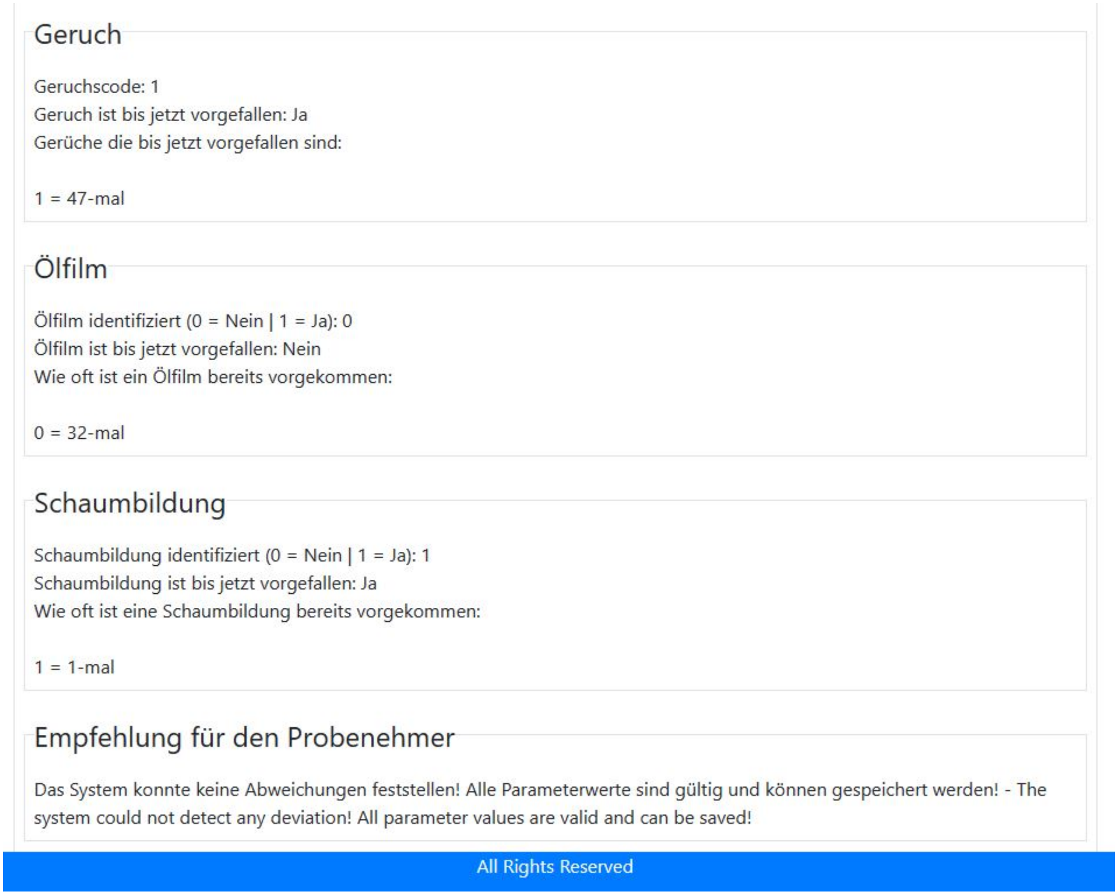


Figure 6.8.: Basic test 2: Create new sample and execute water analysis

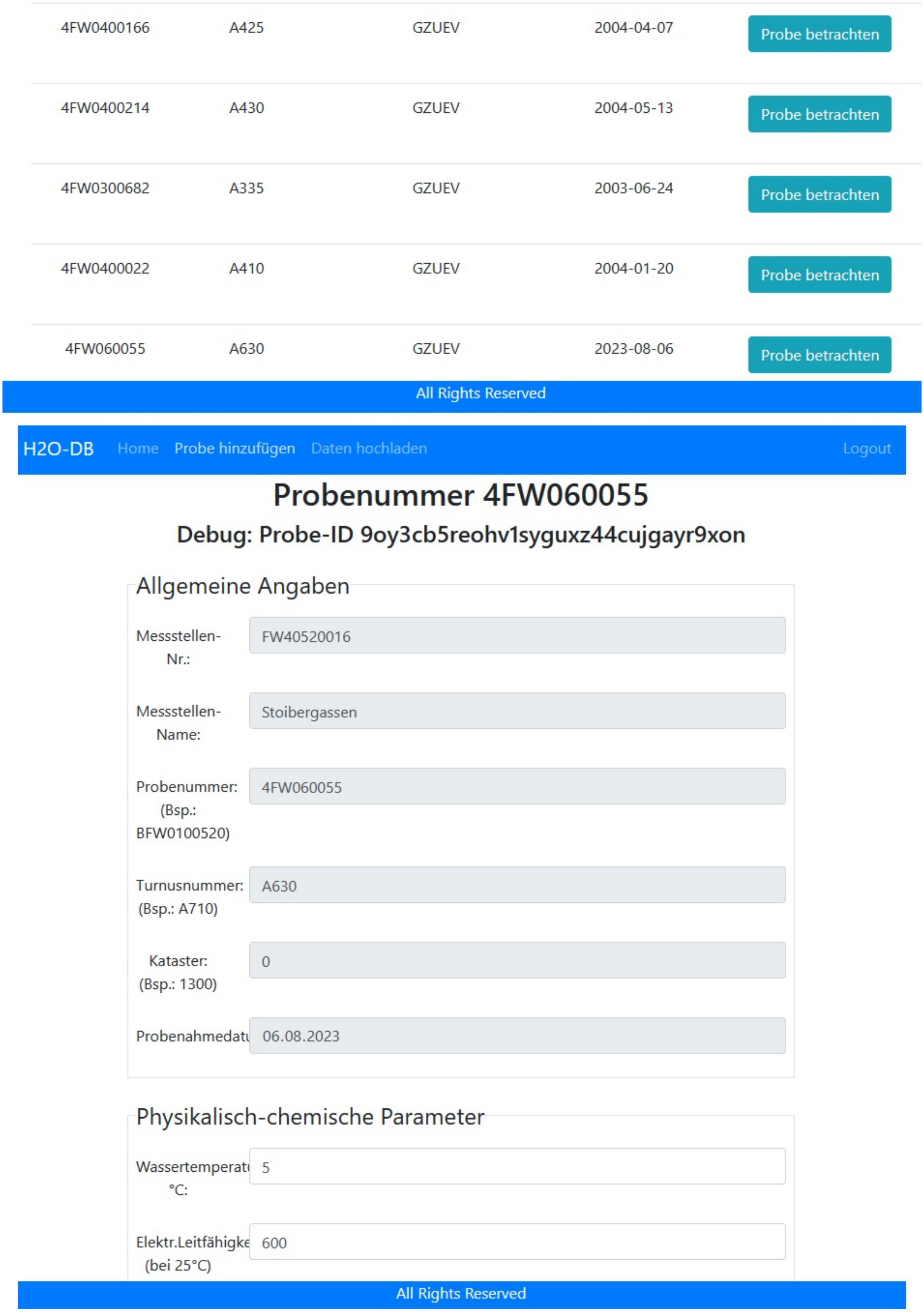


Figure 6.9.: Basic test 2: View new added data of sample

6.3.1.3. Test execution - Water analysis test 1: Routine procedure - No deviation detected

Test execution: The test was passed because all the requirements specified below were satisfied.

Acceptance criteria:

1. The user can select a measuring point and create a new sample.
2. The user can insert measuring data and executes the water analysis.
3. The system performs the water analysis.
4. The system displays the water analysis results (Plausibility check and frequency analysis).
5. The parameter values are within the test barriers.
6. The system displays the recommendation, which states that no deviation was found, and that the sample can be saved.
7. The user can save the new sample.

The next paragraph illustrates results of the execution with important screenshots.

6.3.1.3.1. Test execution results: Figures 6.10 and 6.11 shows that creating a new sample is possible and the user can input values for the metadata fields, the physico-chemical parameter values, and the organoleptic parameter values. Figures 6.12 and 6.13 demonstrate the results of the water analysis, the plausibility check, frequency analysis and that the recommendation is displayed. The parameter values are within the test barriers, the frequency analysis provides a statistic about the occurring of the parameter values and the recommendation states that no deviation was found and that the sample can be saved. Finally, after saving the new sample the sample is listed in the sample list of the specific measuring point. (see figure 6.14)

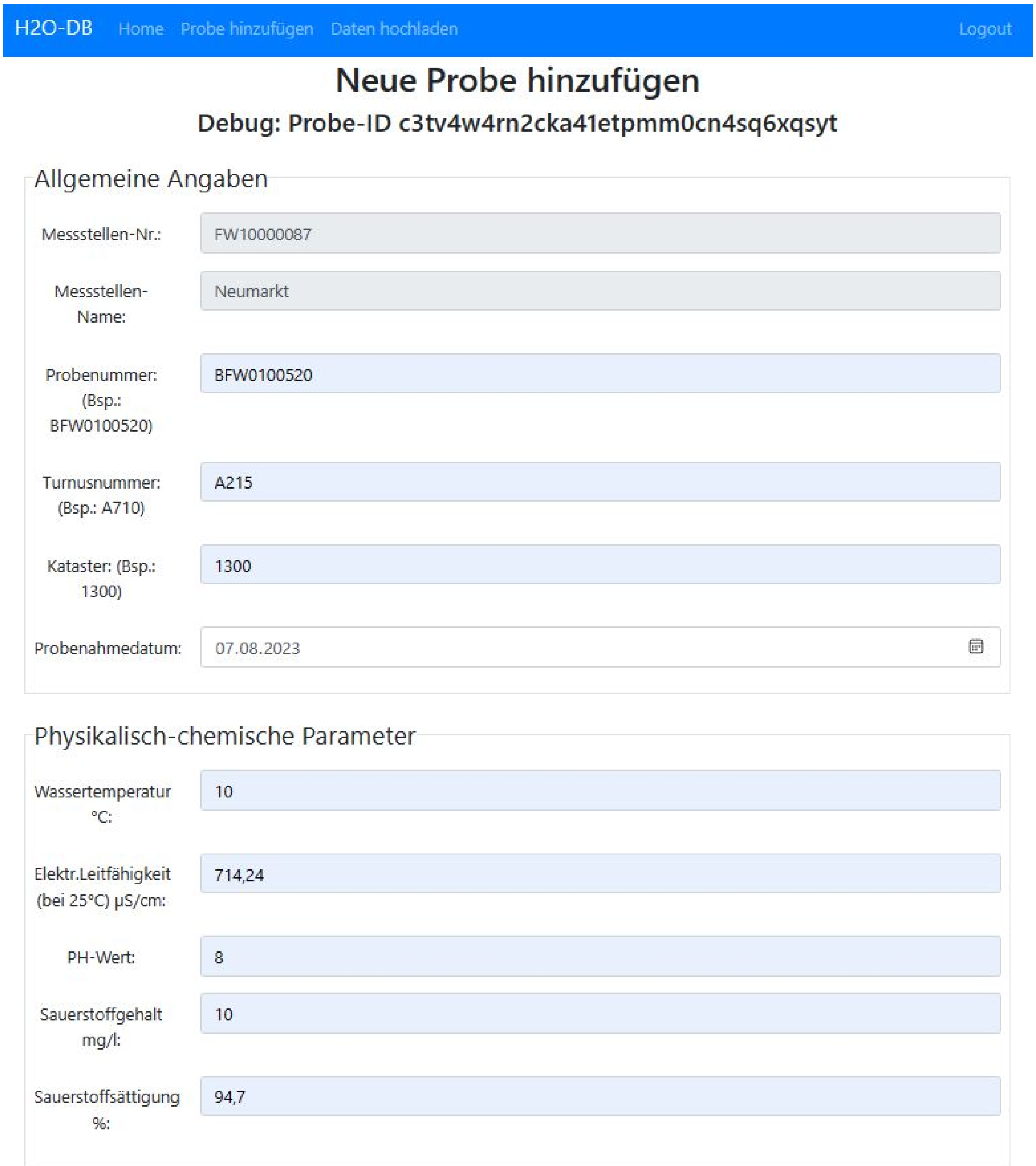


Figure 6.10.: Water analysis test 1: Add new sample - Input metadata and physico-chemical parameters

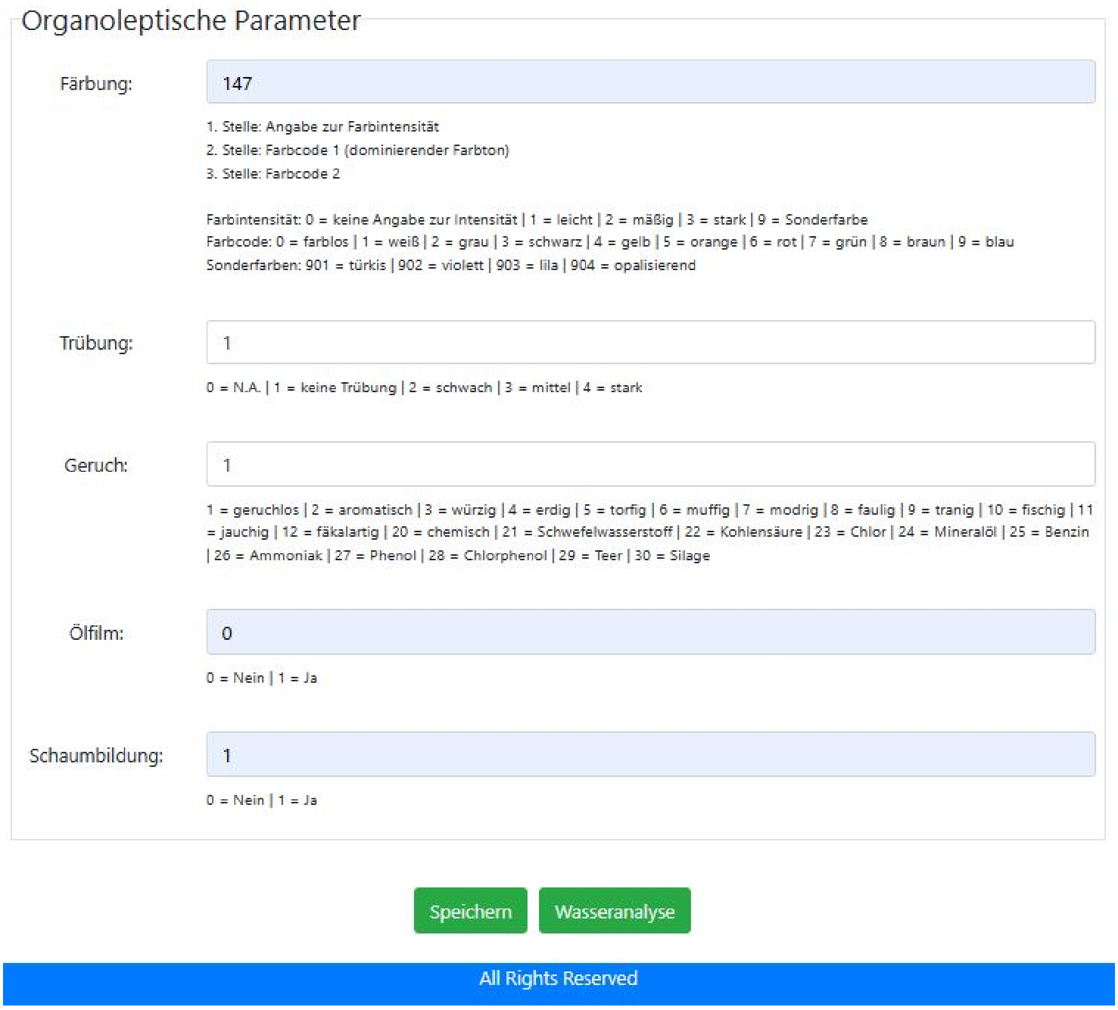


Figure 6.11.: Water analysis test 1: Add new sample - Input organoleptic parameters

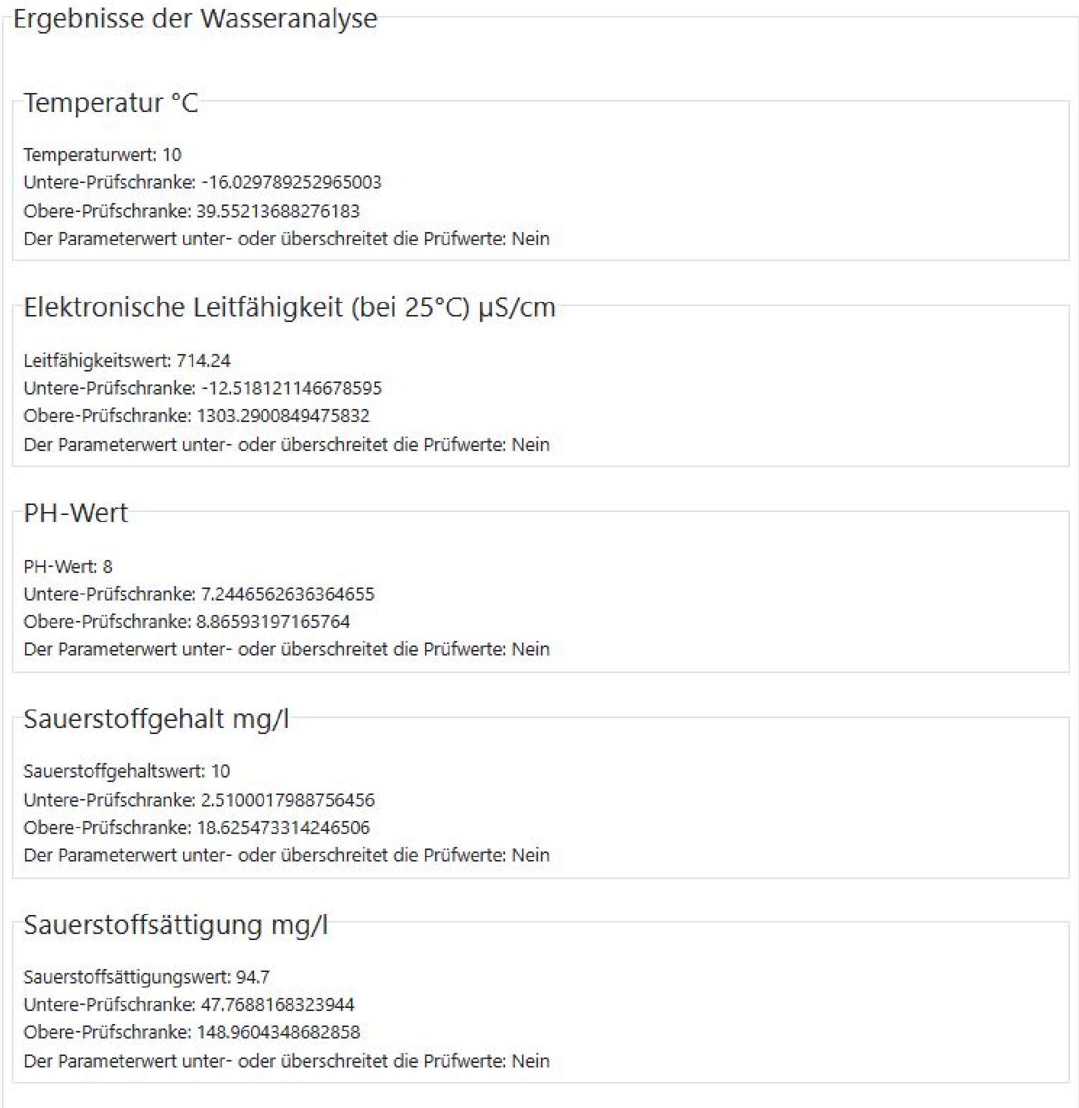


Figure 6.12.: Water analysis test 1: Results water analysis - Plausibility check

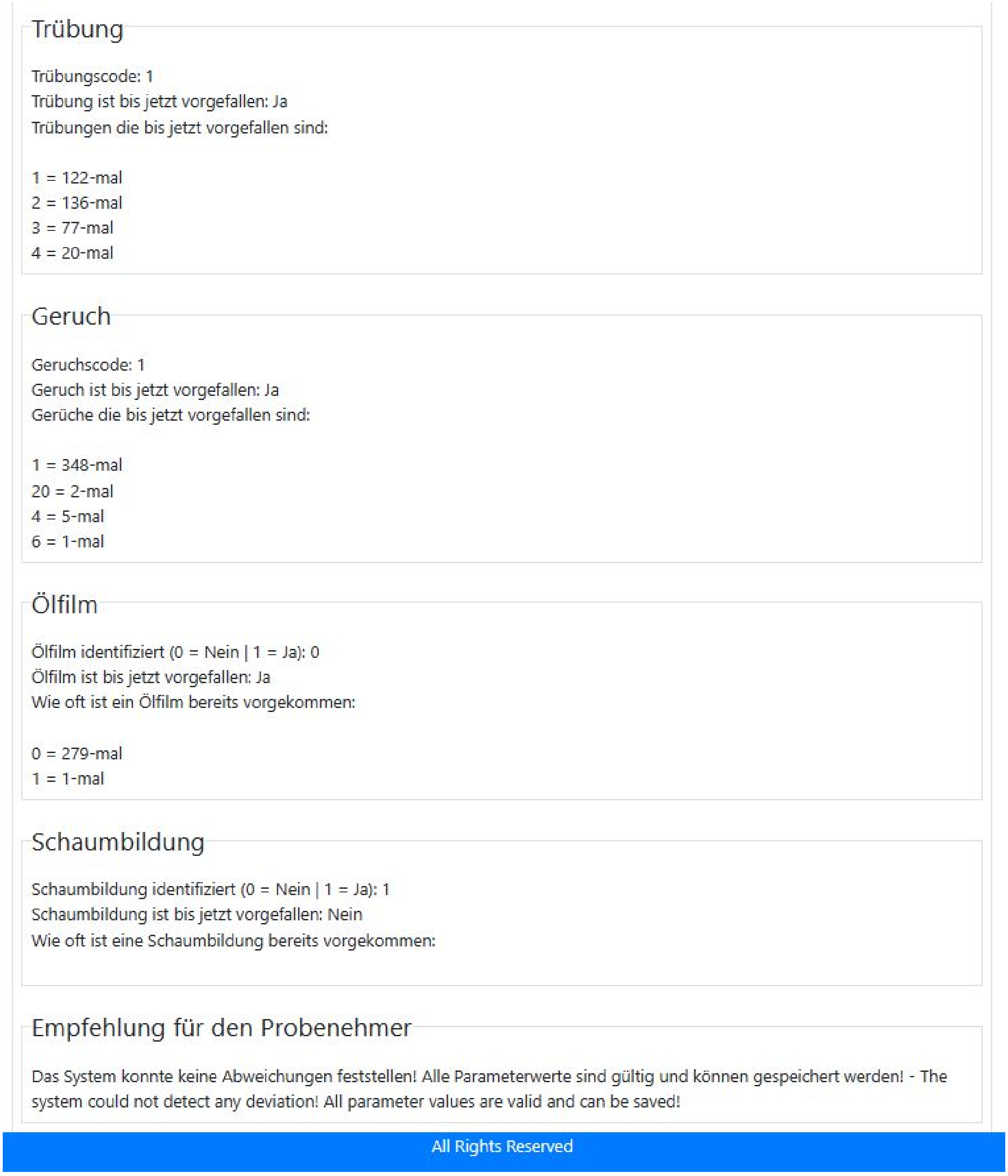


Figure 6.13.: Water analysis test 1: Results water analysis - Frequency analysis and recommendation

Figure 6.14.: Water analysis test 1: New sample in sample list

6.3.1.4. Test execution - Water analysis test 2: Reason 1 - Wrong sampling, wrong location, or possible contamination

Test execution: The test was passed because all the requirements specified below were satisfied.

Acceptance criteria:

1. The user can select a measuring point and create a new sample.
2. The user can insert slightly implausible measuring data three times in a row.
3. The system performs the water analysis three times in a row and displays the correct recommendations.
4. The user can finish his investigation and save the new sample.

The next paragraph illustrates results of the execution with important screenshots. As the main functionality (reading and saving data) was already validated, the screenshots will focus on the technical requirements of the water analysis, more precisely the recommendation results of the rule-based system.

6.3.1.4.1. Test execution results: Figures 6.15, 6.16 and 6.17 demonstrate the three recommendations of the rule-based system for this specific scenario, after the third recommendation the user can save the new sample.

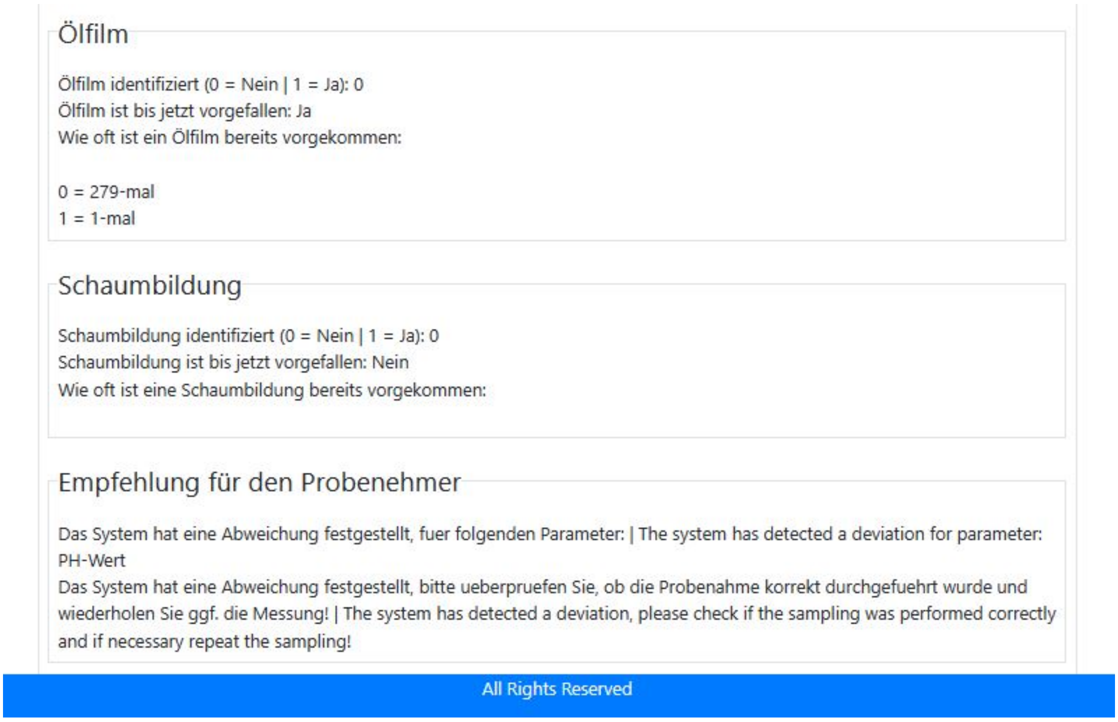


Figure 6.15.: Water analysis test 2: Reason 1 - Wrong sampling, wrong location, or possible contamination - Recommendation 1

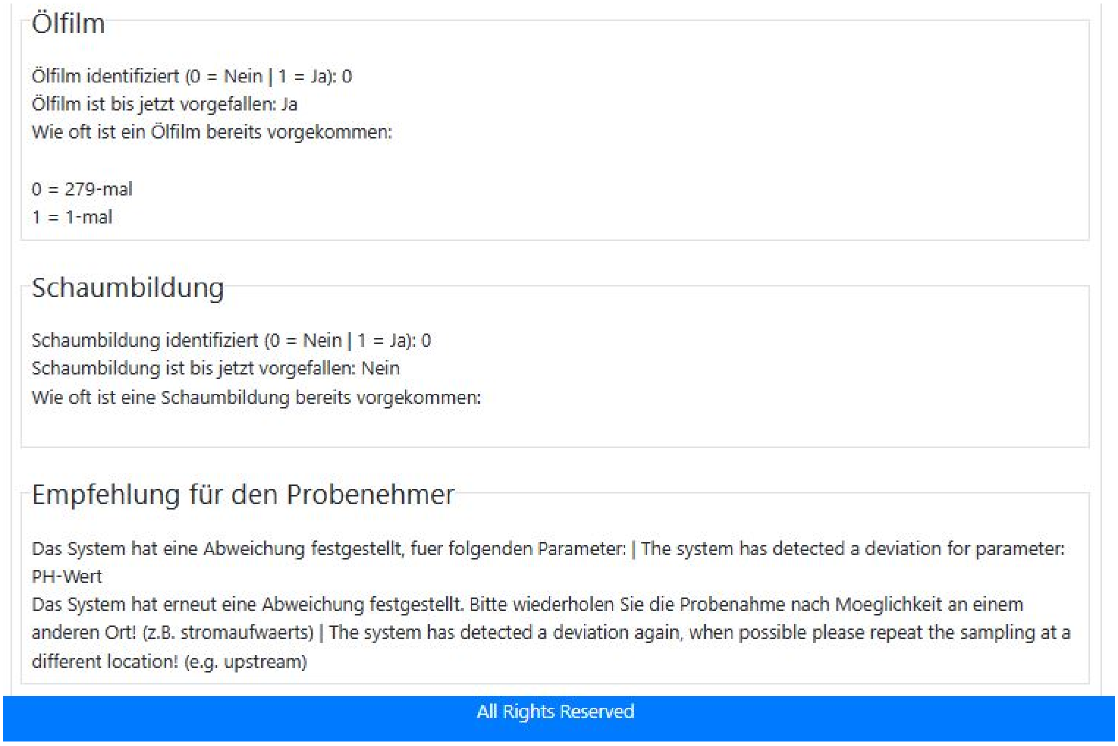


Figure 6.16.: Water analysis test 2: Reason 1 - Wrong sampling, wrong location, or possible contamination - Recommendation 2

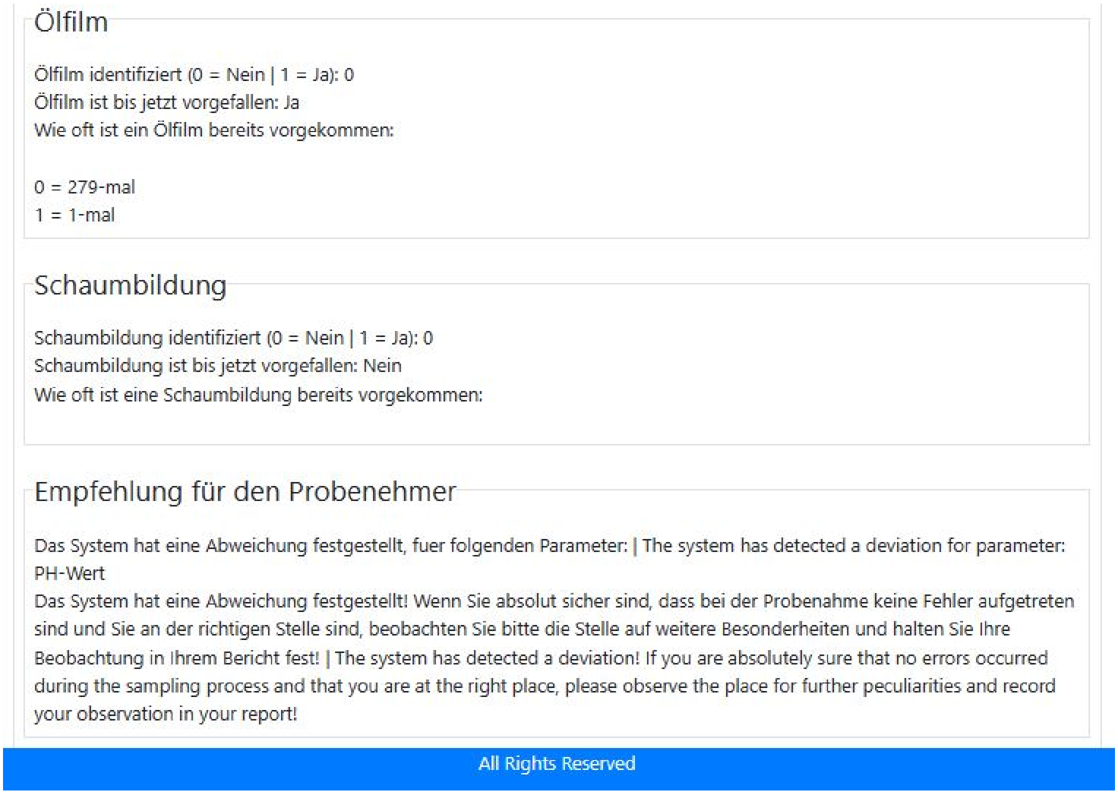


Figure 6.17.: Water analysis test 2: Reason 1 - Wrong sampling, wrong location, or possible contamination - Recommendation 3

6.3.1.5. Test execution - Water analysis test 3: Reason 2 - Deviation detected - Faulty equipment - Iced sensor

Test execution: The test was passed because all the requirements specified below were satisfied.

Acceptance criteria:

1. The user can select a measuring point and create a new sample.
2. The user can insert a negative value for the temperature parameter and start the water analysis.
3. The system performs the water analysis and shows the results on the screen.
4. The system recommends, that the deviation is caused because of faulty equipment and that the sensor of the device is iced and must be defrosted.

The next paragraph illustrates results of the execution with important screenshots. As the main functionality (reading and saving data) was already validated, the screenshots will focus on the technical requirements of the water analysis, more precisely the recommendation results of the rule-based system.

6.3.1.5.1. Test execution results: Figure 6.18 demonstrates the recommendations of the rule-based system for this specific scenario. In this case that the deviation is caused because of faulty equipment and that the sensor of the device is iced and must be defrosted.

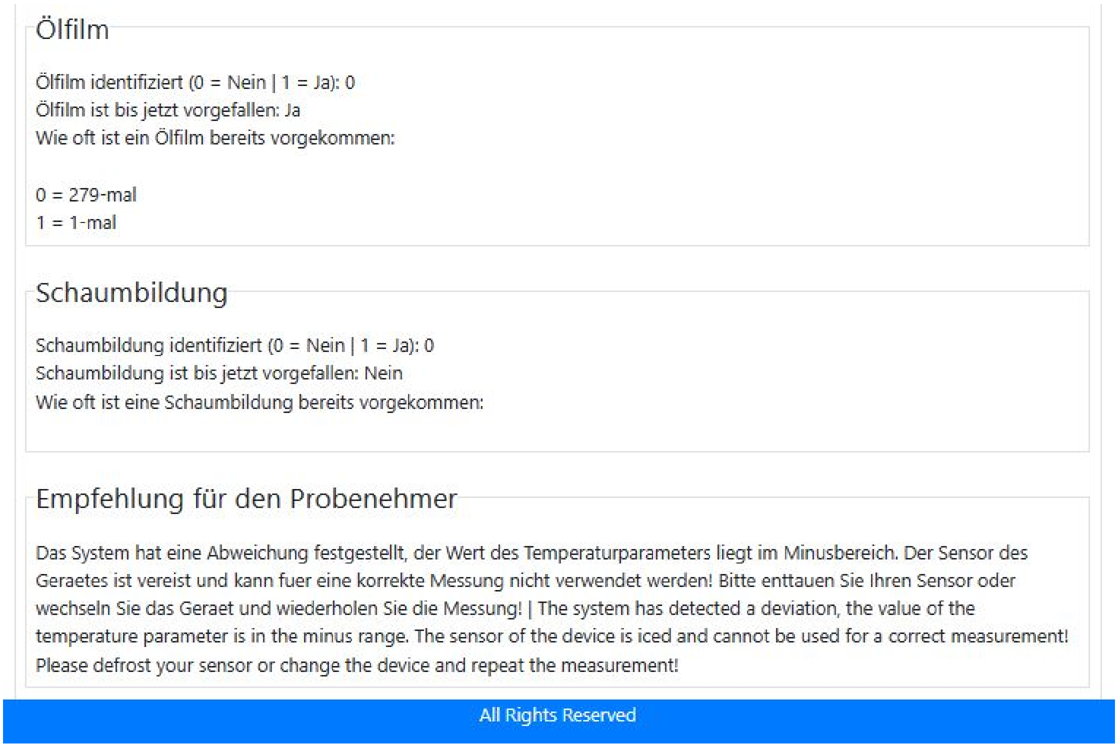


Figure 6.18.: Water analysis test 3: Reason 2 - Deviation detected - Faulty equipment - Iced sensor - Recommendation

6.3.1.6. Test execution - Water analysis test 3: Reason 3 - Faulty equipment - Defective device

Test execution: The test was passed because all the requirements specified below were satisfied.

Acceptance criteria:

1. The user can select a measuring point and create a new sample.
2. The user can insert a very high value for the parameters and start the water analysis.
3. The system performs the water analysis and shows the recommendation on the screen.
4. The system recommends, that the deviation is very high, caused because of faulty equipment and that the sensor or the device is defective and must be checked or changed.

The next paragraph illustrates results of the execution with important screenshots. As the main functionality (reading and saving data) was already validated, the screenshots will focus on the technical requirements of the water analysis, more precisely the recommendation results of the rule-based system.

6.3.1.6.1. Test execution results: Figure 6.19 demonstrates the recommendations of the rule-based system for this specific scenario. In this case that the high deviation is caused because of faulty equipment and that the sensor, or the device must be checked or changed.

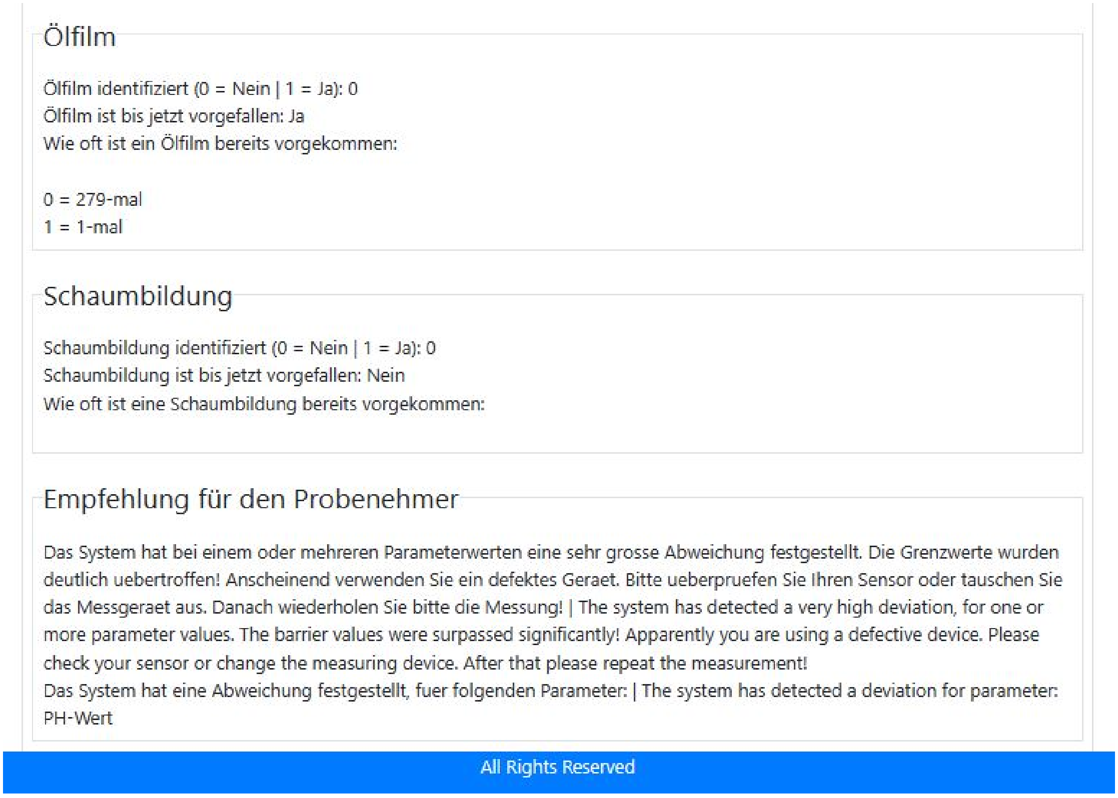


Figure 6.19.: Water analysis test 3: Reason 3 - Faulty equipment - Defective device - Recommendation

6.3.1.7. Test execution - Mobile test 1: Android device

Test execution: The test was passed because all the requirements specified below were satisfied.

<u>Acceptance criteria:</u>

1. The user can call the application on his Android device.
2. The user can upload all three excel files for the tables "messstelle", "h2o_probe" and "probedaten".
3. The user can view data of the new measuring point, the samples, and the parameter values.
4. The user can create a new sample for the previously added measuring point.
5. The user can fill up the input fields.
6. The user can save the new sample.
7. The user can open the previously saved sample.
8. The mobile version of the application is presented without visualization errors. All elements are readable and all fields which should be editable can be edited.
9. The functionality of the prototype can be executed faultless. (Performing the water analysis, reading the results, and saving the sample)

The next paragraph illustrates results of the execution with important screenshots.

6.3.1.7.1. Test execution results: Figures from 6.20 - 6.28 show how the mobile version of the application looks on an Android device.

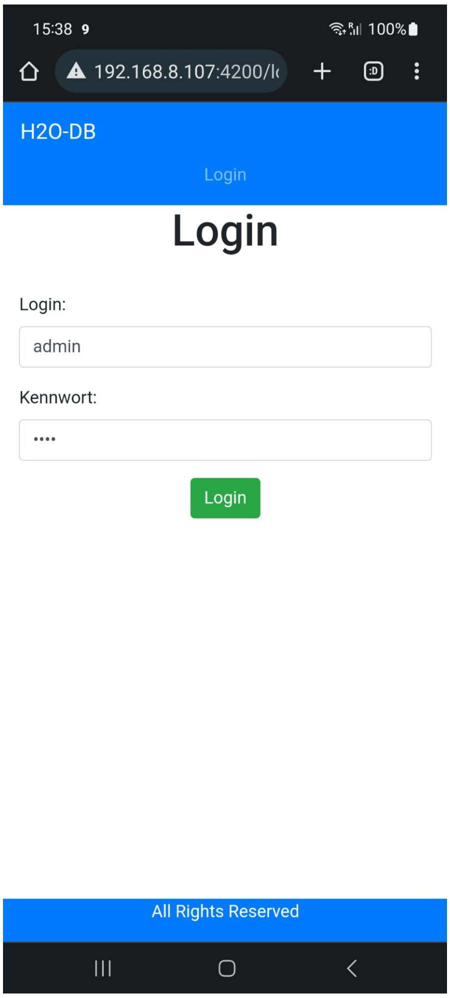


Figure 6.20.: Mobile test 1: Android device - Login

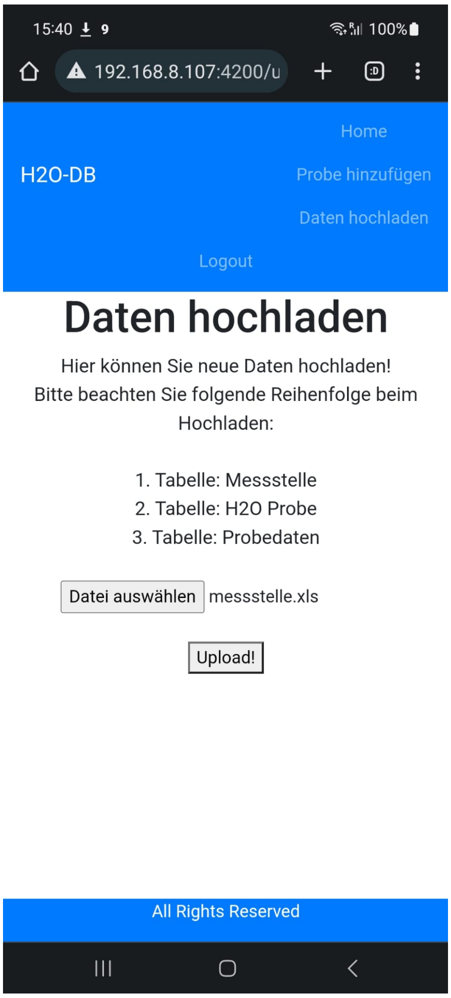


Figure 6.21.: Mobile test 1: Android device - Upload

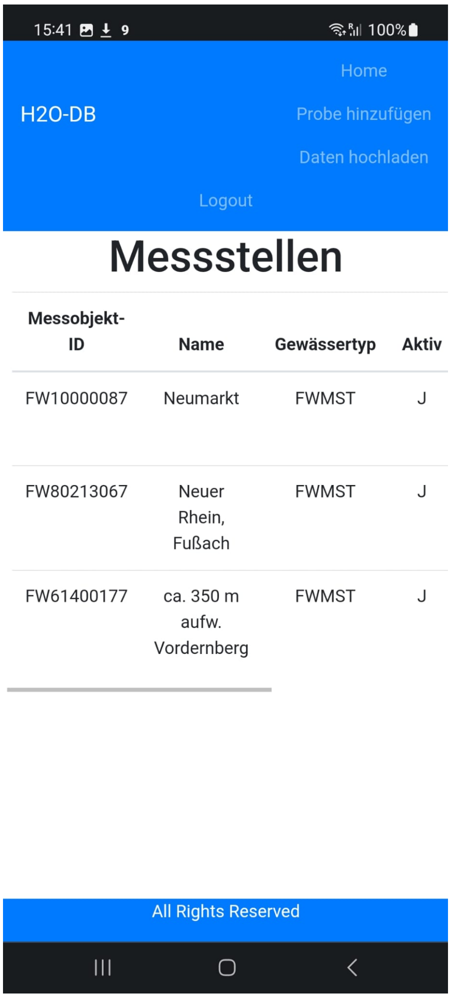


Figure 6.22.: Mobile test 1: Android device - Read new measuring point

Figure 6.23.: Mobile test 1: Android device - Read sample list

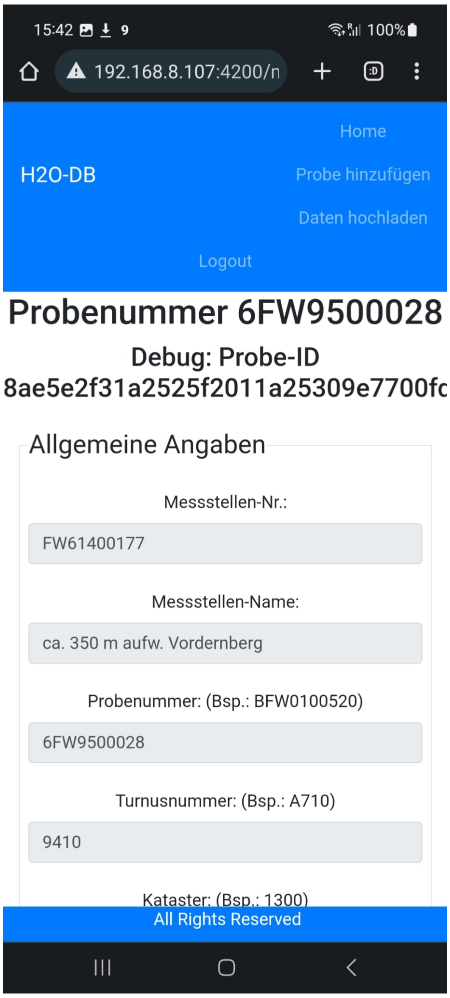


Figure 6.24.: Mobile test 1: Android device - Read old sample

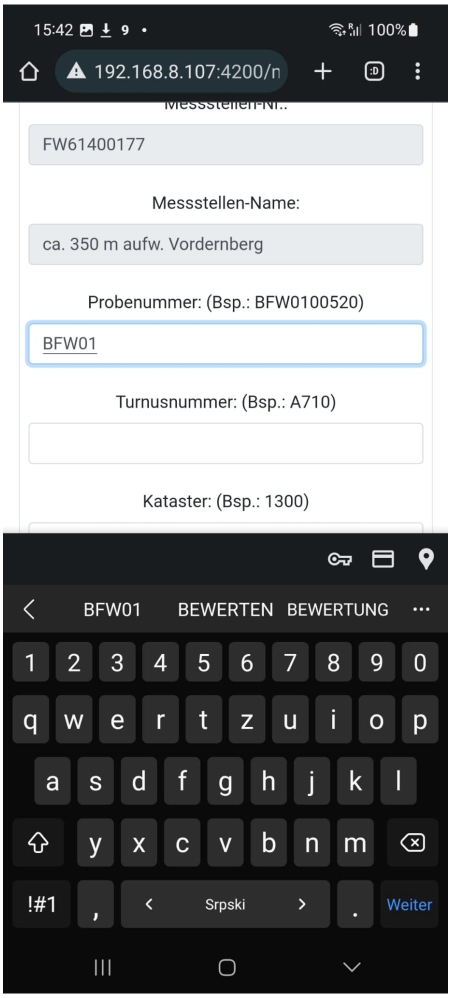


Figure 6.25.: Mobile test 1: Android device - Add new sample

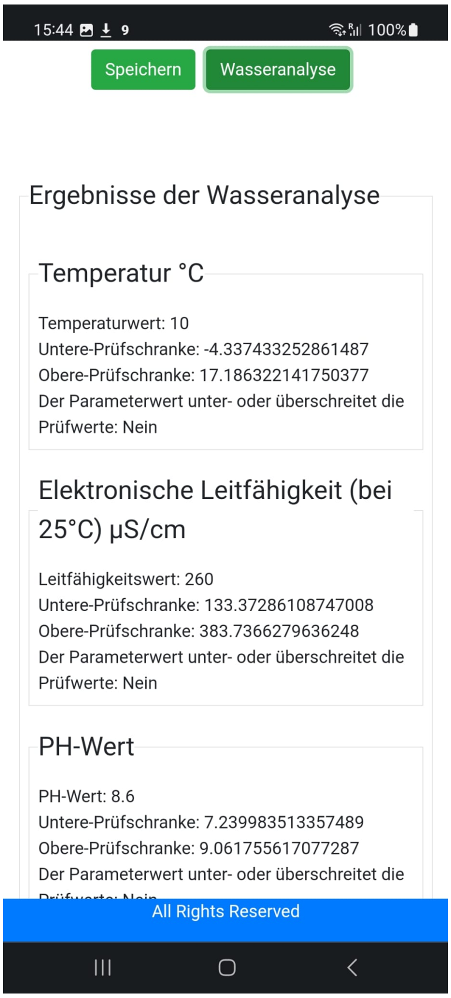


Figure 6.26.: Mobile test 1: Android device - Execute water analysis

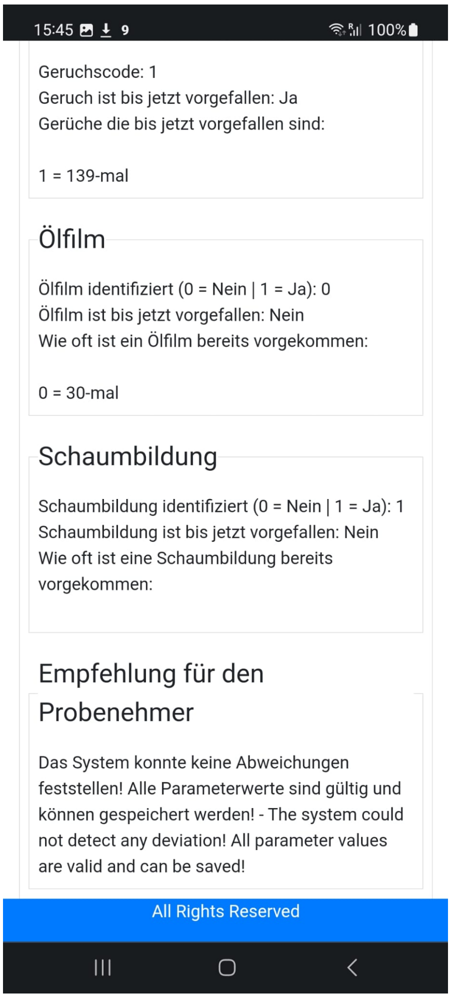


Figure 6.27.: Mobile test 1: Android device - View recommendation

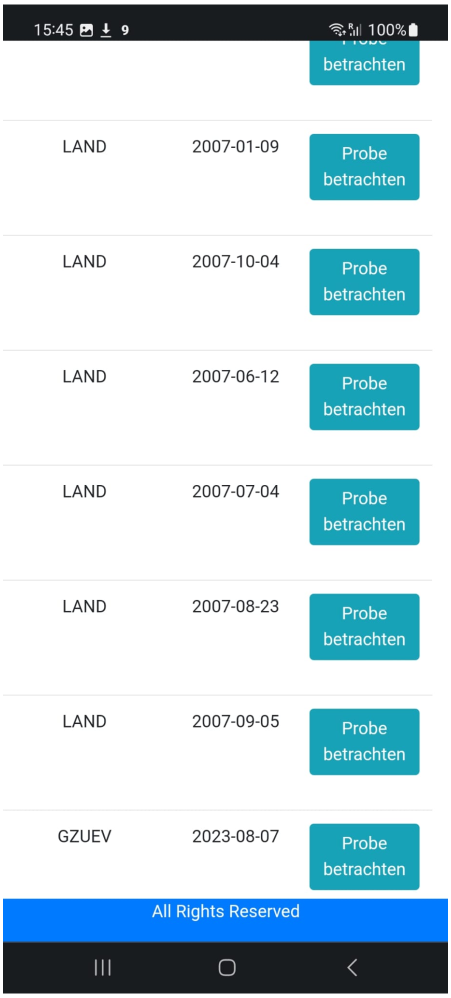


Figure 6.28.: Mobile test 1: Android device - View saved measuring sample in list

6.3.1.8. Test execution - Mobile test 2: iOS device

Test execution: The test was passed because all the requirements specified below were satisfied.

<u>Acceptance criteria:</u>

1. The user can call the application on his iOS device.
2. The user can upload all three excel files for the tables "messstelle", "h2o_probe" and "probedaten".
3. The user can view data of the new measuring point, the samples, and the parameter values.

4. The user can create a new sample for the previously added measuring point.
5. The user can fill up the input fields.
6. The user can save the new sample.
7. The user can open the previously saved sample.
8. The mobile version of the application is presented without visualization errors. All elements are readable and all fields which should be editable can be edited.
9. The functionality of the prototype can be executed faultless. (Performing the water analysis, reading the results, and saving the sample)

The next paragraph illustrates results of the execution with important screenshots.

6.3.1.8.1. Test execution results: Figures from 6.29 - 6.37 show how the mobile version of the application looks on an iOS device.

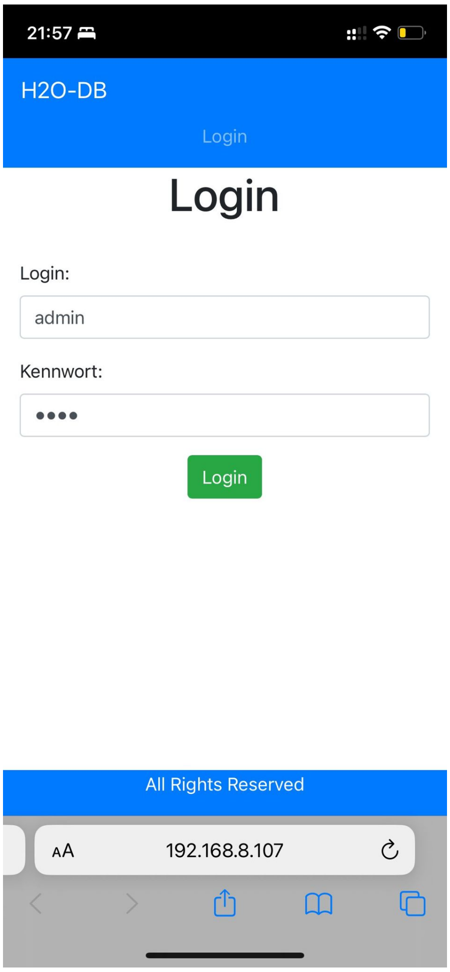


Figure 6.29.: Mobile test 2: iOS device - Login

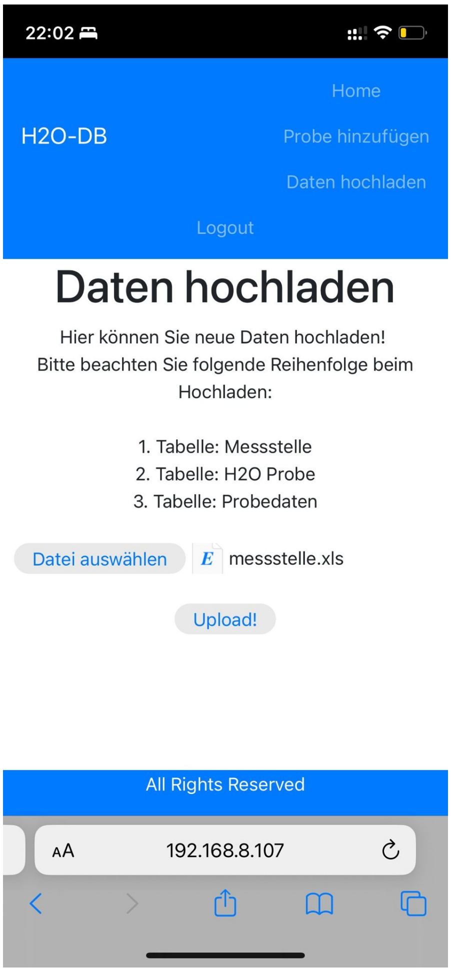


Figure 6.30.: Mobile test 2: iOS device - Upload

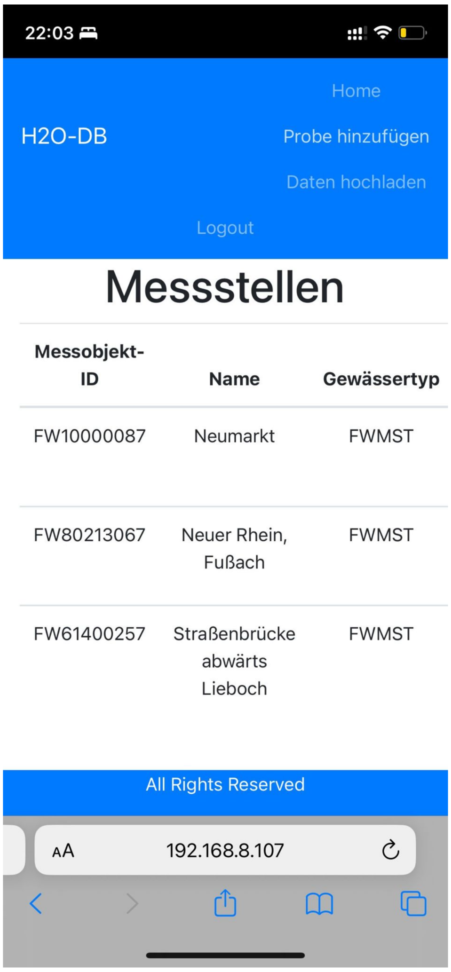


Figure 6.31.: Mobile test 2: iOS device - Read new measuring point

Figure 6.32.: Mobile test 2: iOS device - Read sample list

Figure 6.33.: Mobile test 2: iOS device - Read old sample

Figure 6.34.: Mobile test 2: iOS device - Add new sample

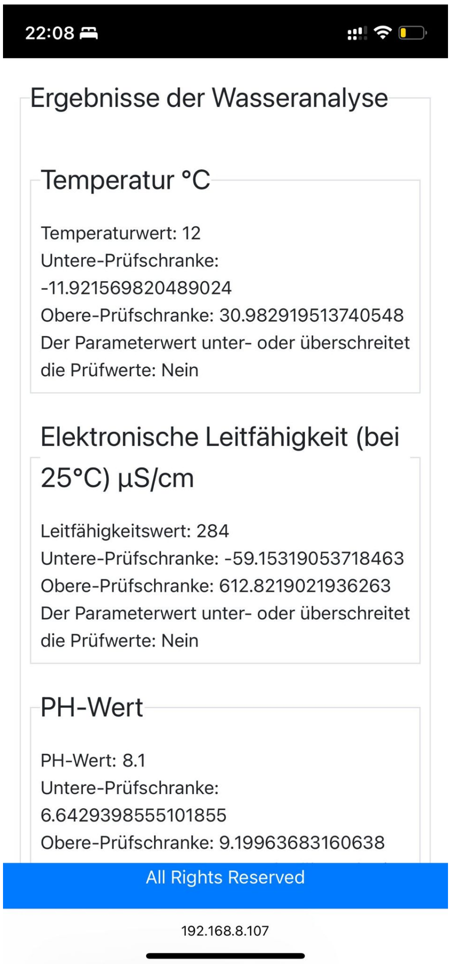


Figure 6.35.: Mobile test 2: iOS device - Execute water analysis

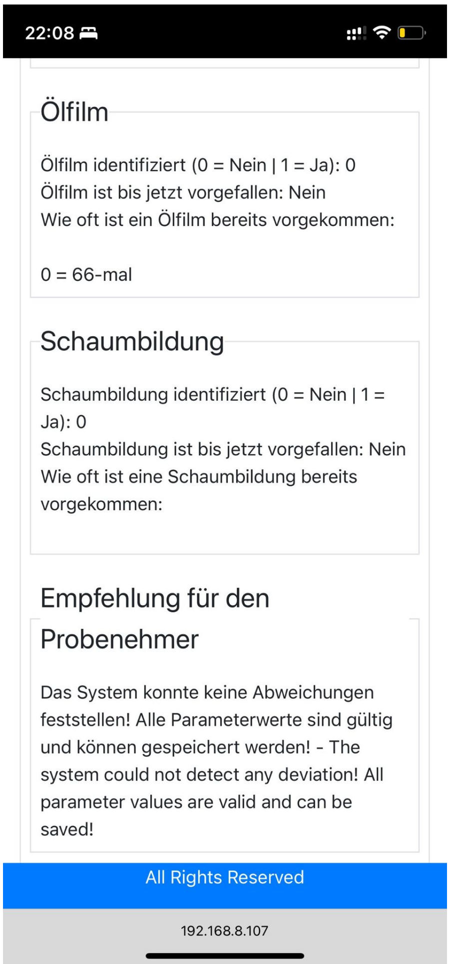


Figure 6.36.: Mobile test 2: iOS device - View recommendation

Figure 6.37.: Mobile test 2: iOS device - View saved measuring sample in list

6.4. Limitations

As stated in the previous chapter, the prototype works very well under the conditions that were defined in the master's thesis for this application. Some of these conditions were intentionally defined in this form, so that the prototype would not exceed its scope, and other conditions resulted from circumstances of the Federal Environment Agency. A more extensive functionality or certain services that were not implemented were not of major importance for the fulfilment of the research questions.

6.4.1. Design limitations

6.4.1.1. Standalone application

The prototype was implemented as standalone application and runs currently on a local machine. The circumstances that the application is not "online" limits its possibilities, launching the application would make the system reliable and accessible for others. Finally, hosting the application by the Environment Agency would be the best solution.

6.4.1.2. State of data

As described in the previous chapter, the prototype is currently not connected to the systems of the Environment Agency. Therefore, it is not possible to keep the data of the prototype up to date in an automated way. The user must add data through the upload function of the prototype or manipulate the data source file of the H2 database. Furthermore, the number of available measuring points, samples and parameter data is also limited. In order that a user wants to analyse a specific measuring point, he must check if the measuring point, he wants to analyse is saved in the database or when it was last updated and if necessary, download it from the H2O database and finally upload the data manually. As you can see a connection to the H2O database or a integration into the systems of the Environment Agency is a very important point for the future and probably indispensable.

6.4.2. Prototype limitations

6.4.2.1. Covered number of water body types

The focus of the prototype was to cover normally distributed streaming waters (Rivers, streams etc.), but other types of water-bodies like groundwater and lakes are also very important test objects of the Environment Agency Austria. Ultimately, only the inclusion of all water body types with all their specific characteristics, would make the system complete.

6.4.2.2. Covered number of recommendations

The aim of the application was to validate the scenarios and through this to answer the research question, to cover many recommendations was not necessary. On the other side, "market maturity" would probably be easier to achieve with a larger number of recommendations.

6.4.2.3. Adding new recommendations requires source code customization

The system is currently designed, that adding rules for new recommendations require modifying and extending the existing source code of the application. One could simply add a new Prolog rule (recommendation) to the Prolog database, but in order for the new rule to be understood by the system, new code must be implemented in the back-end.

6.4.2.4. Statistic method

The prototype uses classic statistical methods like calculations of the standard deviation for the plausibility check. This plausibility check is currently used by the Federal Environment Agency and is completely sufficient for a proper check of the test barriers. However, it would be interesting if a more modern statistical methods can give even better estimates of the state of the water with the help of the on-site parameters.

6.5. Overcoming the current limitations

As touched on a bit in the last chapter, there are some limitations that the prototype currently has. This chapter gives some suggestions for improvements so that the prototype can reach a certain market maturity.

6.5.1. Integration into the environment of the Environment Agency Austria

The integration of the prototype into the environment of the Federal Environment Agency results in many advantages for the use and operation of the application. Significant limitations of the system can be eliminated in this way. First and foremost, a connection would result in the system being permanently accessible via the web. Furthermore, the connection to the H2O database would make it possible for the prototype to be permanently supplied with current data.

6.5.2. Adding all water body types

The inclusion of all water body types with all their specific characteristics, would improve the system significantly and is very important for the Environment Agency Austria. Other water body types such as groundwater and lakes are also being measured by the Agency and this data is indispensable. An extension of the plausibility check and the addition of new rules for these use-cases must be carried out so that the application is complete and can reach market maturity.

6.5.3. Adding new recommendations via a new feature

The current number of recommendations is limited and as already mentioned, an extension with new rules would be of great benefit to the user. Efforts to offer a function, which makes it possible to automatically add new rules to the system, would also be justified. Such a function would make the system even more powerful, since users could maintain and extend the application by themselves. Nevertheless, it must also be said here that such a function can only be developed with a great deal of effort.

6.5.4. Modern statistical methods

Researching whether more modern statistical methods could provide even better estimates of the state of the water using on-site parameters is an interesting area for further work.

Modern statistical methods such as predictions and complex analysis of old data could significantly increase the power of the application. Perhaps such an extension would also make it possible to find out more about the causes of the calculated deviations.

6.5.5. Extending the use of AI tools

The use of Prolog under the framework of this master's thesis has shown the advantage of logic programming. For further work, however, it would be interesting whether modern AI methods could solve this job even better, or whether even more knowledge about the on-site parameters and their measurements could be generated. Finally, it would also be fascinating if such a technology combined with more modern statistical methods could provide more conclusions about the deviation, its cause, and the general water status.

7. Discussion of results

In relation to the literature analysis carried out in chapter 2, the most important conclusion to be drawn is that the basic statistic methods identified in the literature form a solid basis for the development of a system like the one implemented in this thesis. While literature analysis has shown many more possibilities, the practical environment has turned out to be satisfied with the basic statistics functionality. The AI based component of the prototype implementation builds on a Prolog interpreter. This interpreter has shown its value when implementing reasoning capabilities. The literature analysis has pointed out the nature building blocks of the model and of the implementation. During the development of the Prototype, Prolog has played a much stronger role than initially expected. The rule-oriented parts of the system could be realised much easier than through classic database technology (such as SQL statements).

The plausibility check is a solid way for the sampler to find deviations in the measured parameters. In the case of a deviation which was not caused because of human error or a defective device, the frequency analysis allows the user to look at the historical data of the organoleptic properties. This is very helpful in this case, since the civil engineer can analyse the old data and get a feeling for the water object. This can help him in his further observation and lead to him being able to draw further conclusions about the origin of the deviation. Together with statistical data and rule-based recommendations of the system, the sample taker can complete his sampling report in a more qualitative and fact-based manner than ever before. This is what makes the prototype so valuable to the user.

The use of Prolog has shown that the central management of rules and the associated recommendations offer a great advantage. The rules are manageable in one place and can be added very easily, so the system can simply be expanded with new recommendations. However, what costly is, is the back-end work involved when adding a new recommendation, since adding the new rules requires changing the source code. When the water analysis is carried out, which is triggered by the sampler, the new parameter values are sent back to the back-end via the REST interface. The water analysis is then carried out in the back-end, which also includes checking the Prolog rules. When performing this check, the rules must be queried through methods of the Prolog interface. For new rules to work, these methods must be implemented, only then is a new rule fully integrated. Implementing a function that would be able to insert new rules into the system completely independently would be a very helpful tool. With such a function, the users could easily expand the rule-based system independently and through these simplifications the product would achieve a certain degree of market maturity.

Furthermore, it must be explored if modern statistical methods in combination with modern AI technology could solve this problem in a better way. The question is whether the combination of e.g., data science technologies in the database area with prediction methods or other AI-based technologies can give the user even more precise recommendations. More precise recommendations in the sense that the user is given more information about the cause of the deviation and the general condition of the water object.

From a technological point of view, the system offers good conditions for a modern and progressive web application. The technologies used harmonise well with each other and visually the application looks very user-friendly and practical for a technical application. Technical applications tend to be designed very complicated and difficult to use. The usability and user experience of the prototype could be improved even more as that was not the focus of this master's thesis.

Additionally, if the application is finally integrated into the environment of the Federal Environment Agency, the maintainability and use would improve significantly, and a market maturity could be achieved. The integration of the H2O database allows you to supply the system with new measuring points, samples, and parameter data. New data is essential for the calculations of the plausibility check, especially in times of climate change. Finally, when the application is hosted by the Agency, one could also integrate the offline capability, as this an important functionality for the sample taker when working on the field, as water bodies can have bad internet connections.

8. Outlook and conclusion

In this thesis, problems that can occur during the water sampling process were addressed. First, the sampler lacks information about the water body data during sampling and second also the risk of human error or defective equipment leading to incorrect data being stored. The most important findings from this work are that the use of the prototype can significantly improve a year-old process in terms of digitisation and securing data quality. Improving the sampling process by pre-checking on-site parameters is an important measure to prevent incorrect data collection. The system guides the user through his sampling and supports him when a deviation is detected with very helpful recommendations. The plausibility check of the physico-chemical parameters shows if the measured on-site parameters exceed the calculated test barriers. The frequency analysis for the organoleptic parameters helps the sample taker to get information about the historical conditions of the water. The recommendations allow the user to identify whether the origin of the deviation is due to human error, broken equipment, potential contamination, or other causes. With the results of this water analysis and historical data the civil engineer can continue his investigations and finish his report. Finally, it can be clearly seen that using the prototype can support the user by finding the deviation cause and minimising or preventing incorrect data collection.

As described in the papers "Aimq: a methodology for information quality assessment" and "Beyond accuracy: What data quality means to data consumers. Journal of management information systems" both written by Richard Y. Wang and Diane M. Strong (and others), data quality is a critical concern of organisations from Management Information System (MIS) [LSKW02, p. 133] and incorrect data can have social and economics impacts and must be prevented. [WS96, p. 6]. The prototype is precisely this attempt to improve data quality and prevent incorrect data entry. The use of the prototype helps the sampler to identify data deficits in the measurement values of the on-site parameters at an early stage, since deviations are presented to him directly at the water object. Together with the recommendations and the statistical results, the user can determine the causes of the deviation. If it turns out that the deviation was caused by human error or defective devices, the prototype was able to successfully prevent incorrect data storage and thus make a major contribution to data quality in the field of water monitoring.

From a methodological point of view, of course, the applied methods "Spiral model", "Software Process Validation" and "Knowledge representation" (see chapter 3) were the most useful methods for the development of the model and the prototype. However, from a design perspective, the principles of "Great Principles of computing" [DM15] (see chapter 1.4.1) and the stages of "Managing the Computer Resource: A Stage Hypothesis"

[Nol73] (see chapter 4.2) also were important for the model designing process. Above all, principles of computation, communication, recollection, evaluation, and design had an influence on designing, developing, and evaluating the support model and prototype. One principle of computation describes that a representation of a real process can always contain errors, the system was therefore designed and implemented to the best of my knowledge and belief, in order to avoid representation errors. A communication principle stated that a system's communication must ensure reliable data transmission and another technical principle (recollection) characterized that accessing and storing data must be done properly. An attempt was made to implement both principles by relying on modern technologies that work well with each other and thus ensure error-free communication and data storage. The last two principles have shaped the beginning and end of the development process, as for example a design principle has explained very well that complex problems must be split into smaller parts so that the model can be conceived, understood, programmed, and maintained easier. This principle was implemented by analysing the specific problem of the sample-taking process in detail, especially through the earlier chapters, like the "state of the art analysis" in chapter 2 and chapter 4 the modelling phase. The tools of the last principle "evaluation" are modelling, simulation, experiment, and statistical analysis of data. These tools were a fundamental part of the master's thesis and made a very important contribution to the validation of the system. The Great Principles of computing where always kept in mind when designing, implementing, and evaluating the model, so a reliable error-free problem-solving system could be developed. Furthermore, the descriptions of stage 3 (Control stage) and 4 (Integration stage) by Nolan also had an impact on the design phase of the model. Since these stages are comparable to the situation in the Environment Agency Austria, it was important to try to follow the described approaches when developing the model and prototype. Stage 3 describes the moment when entering into the result of crisis and therefore control tasks were executed, where inefficiencies in computer applications are usually the first target. In the last stage the elimination of the unnecessary tasks is performed and planning tasks for designing new solutions which effectively satisfies user needs, were executed. During the development of the support model and prototype, exactly these phases were gone through since there were also inefficiencies in the sampling process. The aim was to improve the process, by implementing a system which suits the needs of the user and by creating an error-free and useful IT solution of the future. Finally, from a development process perspective the combination of the Spiral model and the Software Process Validation was an effective implementation strategy to develop this prototype. The cycles of the Spiral model, each time leading to a functioning intermediate product, were able to ensure great stability in the development process. From a quality assurance point of view, the model comparison was a very powerful method to ensure the quality of the prototype. In every cycle the validation of the prototype was executed, by comparison of the process model with the implemented prototype version. When a discrepancy was detected, the prototype was corrected, and the comparison performed again until the requirements of the model were achieved. The integration of Knowledge Representation Method is of course the most important feature, because the knowledge-bases system makes it possible for the application to behave more than just like a statistical calculation. The Prolog Reasoner enables the system to provide these important recommendations to the user, without Prolog the system would just deliver statistical results which are based on calculations but would not really help the user to find the cause of the deviation.

From a technological point of view, the choice of technologies is based on analysis of the technologies currently available, but also on the specifications given by the Environment Agency Austria. The promising combination of these selected technologies ultimately led the system to become a modern and progressive web application. Especially by using JavaScript as a front-end technology, the requirement for the system to be platform independent could be implemented without any problems, since React js already handles mobile representation pretty well out of the box. Furthermore, other advantages like fast loading times through refreshing only necessary components and mobility through launching in the web browser, were also very convincing. Additionally, the communication over the REST API with the back-end also worked very well, no major problems have been encountered during development. In addition, the decision to use Java Spring Boot as the back-end technology comes as a requirement from the Environment Agency, but the Java framework is nevertheless a very good choice, because it takes a lot of the burden off the developer during development, be it automated code generation or tools offered by the framework. The back-end runs very stable and the H2 in-memory database could also be integrated very easily. Using an in-memory database turned out to be a good decision as there were many versions of the prototype during the development process and these could potentially have caused erroneous data conditions. Due to the automatic deletion and import of the data with each application restart, the database is always in a clean state and no data errors could have crept in. From water analysis perspective statistical methods and Prolog come to play. Since the statistical part just include basic statistical measures such as variance, standard deviation and test barriers, no additional frameworks or special libraries were required. The handling of the calculations with the historical data via the database also ran smoothly thanks to the Spring Boot methods. On the other hand, Prolog did need a connection to a library. The external library "jpl.jar" must be imported to be able to use the Prolog methods and to communicate with the Prolog database. In summary, using the Prolog methods and database was also easy once you understood how the Prolog rules and facts can be executed. The combination of statistical methods and a knowledge base system offers advantages, since you can combine mathematical calculations from historical data with logic programming, which ultimately result in useful recommendations.

Finally, future research should consider if the system would benefit from a more complex statistical methods or other AI tools. It would be interesting to know if, for example, predictions with linear regression or machine learning techniques such as deep learning (or other AI tools) can generate more knowledge about the water status and through this provide an even better basis for more precise recommendations.

Bibliography

[Arn92] Stephen E Arnold. Information manufacturing: the road to database quality. *Database*, 15(5):32–39, 1992.

[Ate17] Tal Ater. *Building progressive web apps: bringing the power of native to the browser*. " O'Reilly Media, Inc.", 2017.

[Ausa] Environment Agency Austria. Environmental expertise and it know-how. `https://www.umweltbundesamt.at/en/services/services-it`. (Accessed 02.10.2023).

[Ausb] Environment Agency Austria. Water quality accurately measured. `https://www.umweltbundesamt.at/en/news-reports/news-eaa/en-news-2019/news-en-190718`. (Accessed 02.10.2023).

[BL85] Ronald J Brachman and Hector J Levesque. Readings in knowledge representation. Technical report, AT and T Bell Labs., 1985.

[Boe86] Barry Boehm. A spiral model of software development and enhancement. *ACM SIGSOFT Software engineering notes*, 11(4):14–24, 1986.

[Bra88] Ronald J Brachman. The basics of knowledge representation and reasoning. *AT&T technical journal*, 67(1):7–24, 1988.

[Bun97] Alan Bundy. Artificial intelligence techniques. *Artificial Intelligence Techniques: A Comprehensive Catalogue*, pages 1–129, 1997.

[Den03] Peter J Denning. Great principles of computing. *Communications of the ACM*, 46(11):15–20, 2003.

[DM15] Peter J Denning and Craig H Martell. *Great principles of computing*. MIT Press, 2015.

[DMD06] Aldo De Moor and Harry S Delugach. Software process validation: Comparing process and practice models. In *EMMSAD*, pages 171–178, 2006.

[Dom17] Scott Domes. *Progressive Web Apps with React: Create lightning fast web apps with native power using React and Firebase*. Packt Publishing Ltd, 2017.

[ea19a] Nikolaos Doulamis et al. Waterspy: A high sensitivity, portable photonic device for pervasive water quality analysis. *Sensors*, 19(1):33, 2019.

[ea19b] Rotman Criollo et al. Akvagis: An open source tool for water quantity and quality management. *Computers & Geosciences*, 127:123–132, 2019.

[ea19c] Sidrah Hafeez et al. Comparison of machine learning algorithms for retrieval of water quality indicators in case-ii waters: a case study of hong kong. *Remote Sensing*, 11(6):617, 2019.

[ea19d] Umair Ahmed et al. Efficient water quality prediction using supervised machine learning. *Water*, 11(11):2210, 2019.

[FMoAMa] Regions Federal Ministry of Agriculture, Forestry and Water Management. Protection and monitoring of water bodies. `https://info.bml.gv.at/en/topics/water/water-quality-and-water-protection/protection-and-monitoring-of-water-bodies.html`. (Accessed 02.10.2023).

[FMoAMb] Regions Federal Ministry of Agriculture, Forestry and Water Management. Water and data (wisa). `https://info.bml.gv.at/en/topics/water/water-and-data-wisa.html`. (Accessed 02.10.2023).

[FMoAMc] Regions Federal Ministry of Agriculture, Forestry and Water Management. Water in austria. `https://info.bml.gv.at/en/topics/water/water-in-austria.html`. (Accessed 02.10.2023).

[FMoAMd] Regions Federal Ministry of Agriculture, Forestry and Water Management. Water quality and water protection. `https://info.bml.gv.at/en/topics/water/water-quality-and-water-protection.html`. (Accessed 02.10.2023).

[Foua] The Apache Software Foundation. What is maven? `https://maven.apache.org/what-is-maven.html`. (Accessed 05.07.2023).

[Foub] OpenJS Foundation's. About node.js. `https://nodejs.org/en/about`. (Accessed 05.07.2023).

[H2] H2. H2 database engine. `https://www.h2database.com/html/main.html`. (Accessed 05.07.2023).

[Hib] Hibernate. Hibernate orm. `https://hibernate.org/orm/`. (Accessed 05.07.2023).

[HLW18] Gerald Hochedlinger and Harald Loishandl-Weisz. H2o fachdatenbank – plausibilitätsprüfung - berechnungsmethodik. `https://wasser.umweltbundesamt.at/h2odb`, January 2018.

[HM19] Walter Huimann and Denis Masinovic. Web application solutions of the environment agency austria. Private Interview, October 2019.

[Isr83] David J. Israel. The role of logic in knowledge representation. *Computer*, 16(10):37–41, 1983.

[Kar] Ranga Karanam. Go java full stack with spring boot and react. `https://www.udemy.com/user/in28minutes/`. (Accessed 10.01.2021).

[Köl17] Walter Kölle. *Wasseranalysen - richtig beurteilt*. Wiley-VCH Verlag, 2017.

[KS03] Cynthia F Kurtz and David J Snowden. The new dynamics of strategy: Sense-making in a complex and complicated world. *IBM systems journal*, 42(3):462–483, 2003.

[LSKW02] Yang W Lee, Diane M Strong, Beverly K Kahn, and Richard Y Wang. Aimq: a methodology for information quality assessment. *Information & management*, 40(2):133–146, 2002.

[Mal16] Ivano Malavolta. Beyond native apps: web technologies to the rescue!(keynote). In *Proceedings of the 1st International Workshop on Mobile Development*, pages 1–2. ACM, 2016.

[Mar83] James Martin. *Managing the data base environment*. Prentice Hall PTR, 1983.

[MLL19] Fitore Muharemi, Doina Logofătu, and Florin Leon. Machine learning approaches for anomaly detection of water quality on a real-world data set. *Journal of Information and Telecommunication*, pages 1–14, 2019.

[Nol73] Richard L Nolan. Managing the computer resource: a stage hypothesis. *Communications of the ACM*, 16(7):399–405, 1973.

[npm] Inc. npm. About npm. `https://docs.npmjs.com/about-npm`. (Accessed 05.07.2023).

[Poh15] Rolf Pohling. *Chemische Reaktionen in der Wasseranalyse*. Springer-Verlag, 2015.

[QGI] QGIS. Qgis - the leading open source desktop gis. `https://www.qgis.org/en/site/about/index.html`. (Accessed on 26.11.2019).

[RH] Inc. Red Hat. What is a rest api? `https://www.redhat.com/en/topics/api/what-is-a-rest-api`. (Accessed 05.07.2023).

[Smi82] Brian Smith. Reflection and semantics in a procedural language. *Technical Report, TR-272, MIT Laboratory for Computer Science*, 1982.

[SP] SWI-Prolog. Jpl: A bidirectional prolog/java interface. `https://www.swi-prolog.org/packages/jpl/`. (Accessed 29.06.2023).

[SPA19] Argyrios Samourkasidis, Evangelia Papoutsoglou, and Ioannis N Athanasiadis. A template framework for environmental timeseries data acquisition. *Environmental Modelling & Software*, 117:237–249, 2019.

[Sta] Getting Started. Get started with bootstrap. `https://axios-http.com/docs/intro`. (Accessed 07.07.2023).

[Tea] Bootstrap Team. Get started with bootstrap. `https://getbootstrap.com/docs/5.3/getting-started/introduction/`. (Accessed 21.10.2019).

[TFMoAM15] Environment The Federal Ministry of Agriculture, Forestry and Water Management. Probenahmehandbuch chemie wasser, grundwasser, oberflächengewässer, sedimente. Juni 2015.

[VDA98] Wil MP Van Der Aalst. Three good reasons for using a petri-net-based workflow management system. *Information and process integration in enterprises: Rethinking documents*, pages 161–182, 1998.

[VMwa] Inc. VMware. Spring data jpa. `https://spring.io/projects/spring-data-jpa`. (Accessed 05.07.2023).

[VMwb] Inc. VMware. Spring framework. `https://spring.io/projects/spring-framework`. (Accessed 05.07.2023).

[Win95] Terry Winograd. From programming environments to environments for designing. *Communications of the ACM*, 38(6):65–74, 1995.

[WPP77] David HD Warren, Luis M Pereira, and Fernando Pereira. Prolog-the language and its implementation compared with lisp. *ACM SIGPLAN Notices*, 12(8):109–115, 1977.

[WS96] Richard Y Wang and Diane M Strong. Beyond accuracy: What data quality means to data consumers. *Journal of management information systems*, 12(4):5–33, 1996.

A. Appendix

For further information about the application and for source code insight, please check out the code on this GitHub repository: `https://github.com/igici98/h2omobile`.