1 Introduction

The examination process is a large part of education in general. Great amounts of time go into organizational problems and overhead. Although digitizing exams would . A step towards electronic examination would make the process more flexible, scalable and resource-efficient. Meanwhile, leading to a more accurate depiction of a students' competence.

Digitizing exams is no new idea. Although, many concepts [cite papers that talk about byod or multi media room exams] and implementations focus on conducting e-exams in the same environment as *paper-based exam*. Resulting in exams that are either conduced on the universities hardware [@KlausurenAndDerUniMainz]. Alternative the exam is a so called **BYOD** (bring your own device) exam, where the exam taken student owned devices [@GLM2015_RobertPeregoodoff].

With the the current COVID-19 situation decentralizing exam would be of both economical and Technically the

It is important to notice, that *decentralized e-exams* differ from *paper-based exams* and even from *centralized e-exams* in some key points. Foremost, the examiner has less control over the environment the exam is taken under. This raises questions about exam integrity and fairness. These questions must be addressed through careful conceptualization of questions and intelligent software design.

Of course some *e-exam* are already conducted today. Large proportions are making use of a **proctoring system**. In such a system a supervisor can access the examinees' device, can monitor all their activity and will watch them through their webcam. This proctoring process is costly. It hardly scales and still easily can be fooled.

Further, test-taking applications are found in many *LMS' (Learn Management Systems)* such as [@Ilias], [@Moodle] or [@Blackboard]. Unfortunately, most often these applications focus on student self-assessment. They also majorly vary in quality and utility. As they are integrated in a complete LMS, changing to the *best* implementation is in many cases not an option.

Last, as exam data is highly confidential, there is a strong argument to be made against closed source solutions. It is crucial to know exactly how the used application works and how data is handled.

Adding, open source projects are less prone to major security issues as the development can leverage the crowd sourcing capabilities that an open source system provides.

2 Design oriented research

[@Hevner] proposes guidelines an approach to conduct design-oriented research

1. **Design as an Artifact:** The result of this research project is an IT artifact. It provides an implementation of a specific electronic exam. This artifact aims to be used in the real world examination process.
2. **Problem Relevance:** With the rise of the Corona crisis E-Education suddenly has become a necessity. While there is a wide variety of options when it comes to distributing educational content, there is a shortage of cheap and well-thought-out systems for electronic exams. Even after the Corona crunch, conducting electronic exams has the power to make the assessment process less time and cost intense.
3. **Design Evaluation:** The effectiveness of the artifact was evaluated with particular focus on cheating behaviour. First tests were conducted in a low stakes environment that only some degree can emulate a real test taking environment. Further tests and improvements can be conducted and made in real world exams.
4. **Research Contribution:** There are few systems designed for providing high academic validity (i.d. prevent cheating in exams). Many systems that do exist, make use of *Proctoring* (the student is continuously watched over video stream) environments which are expensive and still easily can be fooled. This research artifact aims to minimize academic dishonesty through design decisions.
5. **Research Rigor:** This thesis builds upon research in the fields of education. Taking into consideration what other Universities have already incorporated into their examination process and what empirical studies have shown to be valuable and efficient.
6. **Design as a Search Process:**
7. **Research Communication:** This thesis focuses on illustrating design considerations, that were made in order to develop an artifact, that most closely fit the needs of educators. Therefore,

this thesis is of interest to those want to build upon the considerations that were proposed. The actual technical intricacies should only briefly be touched upon, to give some insight into the technical base of the given artifact.

3 Requirements for E-Exams

As discussed, we find e-exams to be advantageous in a variety of ways. Still, it must be ensured that e-exams meet the same standards that are asked for in paper-based exams. For this cause we have to define requirements as a basis for any examination. [@Handke2012] provides such requirements. In this thesis we ask how to design a sound e-examination software. Thus, we focus on topics that are directly influenced by this software. We do not consider issues concerning the content of the exam. Further, we divided the given requirements into three broad categories:

The fist requirement defines the desired outcome of an exam:

- **General Validity.** Exams should aim to provide an accurate depiction of an examinees competence level.

Further, we find requirements that mainly influence interactions of examinees and examiners with the examination system:

- **Protection against contestation.** No formal, or technical deficiencies should occur, that would question the validity of the exam.
- **Equal Treatment.** Individual examinees must be treated equally.
- **Protection against cheating.** The exams outcome must be protected against manipulation by examinees.
- **Transparency.** The examination process and results must be understandable and verifiable.

Lastly, we find requirements that mainly influence the technical implementation of how the examination system handles data:

- **Protection of Data.** Data of examinees is personal data, as such it must be protected from misuse.

- **Integrity.** Exam data must maintain consistency, accuracy and trustworthiness throughout its entire lifetime.
- **Attributability.** A taken exam must uniquely map to a single examinee and vice versa.

These categories set a general framework of how to design any examination system. In case of this thesis, e-exams are of interest. They use specific design principles to match the previous requirements. In the following we will discuss these design principles.

3.1 General Validity

Generally, examinations should support the purpose of universities to produce highly capable individuals [@ETH]. The measurement of success in that aspect is largely based on the students performance in exams. Subsequently, students are highly incentivized to focus their studies on a specific exam format and its question types. This interdependency between knowledge acquisition and the examination shows the importance of exam design. Further, it poses the question of what and how to test. We find different question types are particularly well suited for testing specific aspects of learning. These questions types can be defined as follows [@ETH]:

- **(Semi) Closed questions**, which mainly revolve around the demonstration of *factual knowledge*. Solutions are not disputable, there are only right and wrong answers. Typical answer formats include multiple-choice and simple text input. *E.g. "What does* BYOD *stand for?"*
- **Competence questions**, which are suited to test for a certain *practical skill*. Solutions are given in form of an implementation of the specific task at hand. *E.g. "Using the provided software, implement an e-exam about e-learning."*
- **Essay-type questions**, which are suited for assessing *transfer knowledge* and *understanding*. Solutions are given by free text input. *E.g. "Explain why subjects in computer engineering are especially well suited for e-exams."*

Further, different degrees of allowed aid for a question can be identified: In open book exams, students are allowed to solve the question at hand using any resource. These open book exams rely mostly on both competence and essay-type questions. It could be argued that these types of questions resemble a real world scenario in which access to information is rarely limited. Meanwhile, in such

open book exam situations, closed question are rendered insignificant as simple factual knowledge is easily accessible. In order to ask closed questions it is therefore necessary to restrict access to any aid.

Classic paper-based exams do not provide a feasible way of combining degrees of allowed aid. Therefore, some question groups tend to be neglected. This constraints possibilities to create an accurate depiction of an examinees actual competence. With e-exams on the other hand, we can implement such a varying degree of usable aid, creating a *partial* open book exam. This can be achieved by letting students generally use any resource they need in order to answer the question. Additionally, we introduce per question time constraints. These time constraints can be adjusted according to the question and question type. Leaving closed questions with a strict time constraint and creating an *either-you-know-it-or-you-don't* situation, where the student has no time to look up any solution. Essay-type questions just as competence questions can employ more generous time frames, giving the examinees freedom to make use of their tools.

Ultimately, examination software in general has no direct impact of what exact questions the examiner asks. The content of a question obviously predefines how well this question can predict an examinees' capabilities. Still, the use of a partial open book mode allows for a diverse set of question. This mode allows to test factual, transfer and practical knowledge to an equally valid degree.

For the requirement of *general validity* we thus find two main design principles. The first principle is the usage of multiple question types. The second, is the enforcement of per question time constraints.

3.2 Protection against contestation

Generally, contestation of entire paper-based exams is not a common problem. This is mainly due to the controlled environment paper-based exams are taken in. Adding, the medium that is used to test examinees (i.e. paper) is fail-safe. E-exams, especially decentralized ones, introduce the possibility of failure of the exam medium. They rely on software, on the operation of a electronic device and of course on internet connection. Failure of the exam medium can lead to students contesting against the validity of the exam.

The reliability of the exam medium is most dependent on the e-exam software. As with any software,

high reliability can only be achieved by rigorous testing and continuous improvements.

Another important point is device operability. Decentralized e-exams are taken on the examinees device. It thus largely lies within in the responsibility of the device owner to assure that it is working as intended. Here must be mentioned, that modern devices generally show low failure rates. As students in any way need a reliable device to participate in their studies, device operability is not a major problem. Still, the examination tool can prevent unnecessary device failure by strongly advising examinees to keep their devices updated, plugged into power and to not use unreliable devices.

The last major point in which the exam medium can fail is connection loss that leads to time deficiencies for students. In normal operation exam answers should continuously be sent to a server to minimize the risk of data loss. In case of connection issues student must be able their exams without problems. Data must then later be send to the server. In case of both a device crash and internet failure, the exam should persist on the local storage of the device. The device then can be rebooted and the exam can be continued.

For the requirement of *protection against contestation* we thus find three main design principles. The first principle is the continuos improvement and sound testing of exam software. The second is advising examinees to assure their device is working as intended. The last is taking connection issues under account and being able to handle them.

3.3 Equal Treatment

Equal treatment of examinees should be ensured throughout the entire examination process, reaching from taking the exam to its correction.

Possible inequality arises in some key areas. In *BYOD* exams student devices are largely heterogeneous– they run different operating systems and consist of different hardware. This fact should not lead to different exam-taking experiences. The choice of hardware should be largely irrelevant. Consequently, it makes little sense to develop proprietary software for each operating system. Modern web technologies provide a common language among different systems. Web applications do not lack speed or functionality and can be adopted cross-platform. The software is hosted at a central entity where it

can be maintained and improved. The software artifact is then delivered via a modern browser. The examination software thus should rely on internet technologies.

The process of correcting exams is also an area where possible inequalities can be found. Especially the correction of exams entirely by hand is immensely time consuming. Resulting in fatigue and thus sometimes in answers checking mistakes. Besides accidental mistakes, [@James 1927] has found negative bias towards students with bad handwriting. He found students with bad handwriting get categorically worse grades than students with better handwriting.

By using e-exams these inequalities can be eliminated. First, some question types, such as multiple-choice questions can be checked automatically. Resulting in an immediate improvement over correcting these questions by hand. This leads to a lower correction load and thus to fewer correction mistakes. Second, as exam answers are available in digital text, reading and checking answers is easier. Answers must not be deciphered, correction of exams can be done faster. Meanwhile, e-exams can also eliminate biases against certain students.

For the requirement of *equal treatment* we thus find two main design principles. First, the software should be device agnostic. Second, the system must leverage automation possibilities.

3.4 Protection against cheating

When thinking about any assessment the consideration and handling of academic dishonesty (e.g. cheating in an exam) is one of the most important parts. Moving from paper to e-exams poses the question what parts must be adjusted to accommodate for changed circumstances and environments.

In his paper [@McGabe] poses seven fields of possible cheating in exams which he then evaluates by occurrence and perceived severeness. Six of which are relevant for this thesis' purposeˆ(The seventh would be *"Using false excuse to delay test taking"*). The fields can be described as follows:

Student cooperation:

- **Knowing the questions.** Learning about the exam content from someone who has already taken it.

- **Cooperation with outsiders.** Receiving disallowed help from someone outside the examination context.
- **Cooperation with fellow examinees.** Copying from another student during an exam with their knowledge or working together to solve questions.

Use of disallowed aid:

- **Exploit environmental circumstances.** Copying from another student during an exam without their knowledge.
- **Use of unauthorized notes.** Bringing prepared cheat notes to use in the exam.
- **Use of electronic, unauthorized aid.** Using search engines or the lecture material to solve questions.

Before thinking about how to obviate these cheating scenarios an important statement must be made: Cheating cannot completely be eliminated. There are always means for students to engage in cheating. Although e-exams cannot change that we can find measures to prevent cheating.

Knowing a question. The creation of questions is a time-consuming process. An examiners strategy thus may be to keep questions as secret as possible and reuse them throughout multiple exams. This is a rather ineffective strategy as platforms such as [@studydrive] often provide comprehensive exam protocols. These protocols are contributed by examinees who have already taken a given exam. The digital nature of e-exams inhibits the use of the above approach. Students are able to capture questions and distribute them even faster and more accurately. Thus, e-exams must choose a different approach. Instead of having few questions and keeping them secret, e-exams have to leverage large question pools. As question pools grow larger it becomes unfeasible for students to *know* every available question.

Cooperation with other examinees. For closed questions this cooperation can be prevented by using tight time restrictions. As already stated above, these questions fall in the category *either-you-know-the-answer-or-you-don't*. There is no need for a lengthy reflection period. With these short time frames, there simply is not time for cooperation with others. For more open question types, time limitations are not as tight. At the same time, answers require more in-depth considerations. To ensure that students write down their own ideas and cannot share their thoughts, the input possibilities must

be limited. Copying and pasting should be disabled to prevent sharing of answers between students. To further inhibit cheating, the order of questions should be be randomized and the navigation between questions should be inhibited.

Cooperation with outsiders. As decentralized e-exams are not conducted in a controlled environment, cooperation with outsiders becomes a severe problem. Examinees could try to take the exam in the presence of an expert. Some try to solve this problem by using proctored e-exams. These exams use live surveillance through webcam and microphone that is evaluated by a person watching in real time. This approach hardly scales as for every 4-5 students a supervising proctor is needed. Programs like [@ETC_Toefl] can make use of such a system, as their high test fees leave room for additional expenses.

Although, live surveillance of students is not a valid option, the psychological effects of being monitored can be leveraged. A measure might be to make use of integrated webcams and microphones of the devices at hand. This video and sound data can be reviewed if needed. More importantly, it creates a mental barrier. If examinees really commit to engage in academic fraud they will most certainly find a way to do so. The goal is simply to prevent those from cheating that would only cheat if they would feel no threat of being caught. The sole existence of any measures makes students behave more honest. This can be compared to video surveillance that makes crime less common at public places [@Welsh2004].

Exploit environmental circumstances. (e.g. Peeking at answers) Again randomization can solve this problem. As questions appear in a different order for each student, even multiple-choice questions cannot simply be copied.

Use of unauthorized cheat notes or electronic aid. Following the argument made about partial open book exams we find that besides time constraints no additional measures must be enforced. Cheat notes are redundant if there is no time to use them.

We find E-exam software to be able to enforce measures against cheating. Still, as specific software is in use, the degree of cheating must constantly be assessed. Further software bugs must be fixed, while security flaws must be identified and resolved.

For the requirement of *protection against cheating* we find three main design principles. First, the creation and management of large question pools must be possible. Second, it must be possible to enforce per question time constraints. Further, the question order must be randomizable. Lastly, a way to create the feeling of being surveilled must exist.

3.5 Transparency

The examination process should be transparent for examinees. Students must be able to understand their mistakes and shortcomings. This implies that the exam software provides ways to give feedback. Further, as examiners are not free of mistakes, corrections can sometimes be faulty. Well implemented transparency allows students to review the examiners correction and contest against individual corrections. Important to mention is, that every student should get the chance to review their exam. The digital nature of e-exams makes this degree of transparency easy to realize. Sharing a corrected digital copy of an exam, allows examinees review their answers and understand their gaps in knowledge. Contestation against certain questions could also be processed within the exam software.

For the requirement of *transparency* we thus find two main design principles. First, the examiner must be able to give feedback to question. Second, this feedback must be available by the student.

3.6 Attributability, Protection of Data and Integrity

Exam data is highly sensitive and demands high levels of information security. As with any information system, basic information security principles apply. The following points prove to be of special importance.

Exam data must be uniquely traceable to examinees. This can be easily realized by having examinees log into an user account before they can perform any action. Examinees either get an unique identifier in-software or an unique identifier that is provided by the testing authority. Any of their actions is then linked to their user id.

To assure solid data protection, strong user rights management must be enacted. This guarantees that only authorized groups can view or correct exams. In this way data is largely protected from missuses.

This measure ties into the integrity of exam data. As access is restricted exam data cannot be changed. To provide even more security, answered questions can be sent to a central server instance as soon as students continue to the next question. Further, frequent database backups of the exam data should be standard procedure.

Another consideration to take into account is the availability of the source code. Processes should be completely transparent and comprehensible. Exam authorities should be able to host exams . This can be achieved by providing the exam software in open source format. Further open source programs can leverage crowd participation to render software bug-free and to eliminate existing security flaws.

For the requirement of *protection against contestation* we thus find three main design principles. First, sound user rights management must be enforced. Second, source code should be well maintained and open source.

4 Software

In the previous section I laid out the requirements an artefact must match in order to count as appropriate exam software. In this thesis we will evaluate five popular software alternatives:

- [@Ilias]
- [@Moodle]
- [@OpenOlat]
- [@Blackboard]
- [@LPlus]

These five systems consist of popular e-learning applications that are in use by large parts of german higher education. *Illias*, *Moodle* and *Open Olat* are open source and free to use. Both *Blackboard* and *LPlus* are closed source and are paid products. Besides LPlus, all of the products are full LMS (learn management systems). A learn management system provides management of the complete e-learning process in a single application. LMS allow for distribution of teaching material, exchange between students and provide a platform for educators to get in touch with their students. The systems discussed here also provide e-assessment features. Universities in german higher often have such LMS in operation. Using the integrated assessment tools becomes a seamless experience for these universities. While this sounds great in theory, the integrated assessment capabilities still need to be evaluated. We will measure the quality of all five tools based on their degree of fulfillment of the requirements and design principles we laid out earlier.

As a baseline all these tool are market ready products. All of them have user management with respective permissions and are actively maintained, laying a foundation for the *requirement of Protection of data, attributability and integrity.* Further, all of them rely on a browser to provide their services, thus being device agnostic, leveraging automation where it is possible and thus matching *the equal treatment requirement* requirement.

4.1 General validity

All products allow a variatey of question types. Matching the first design principle. To completely match the *general validity requirement* exams need to be able to enforce per question time constrains. Here only [@OpenOlat] and [@LPlus] can provide such a feature. The other systems allow for time constraints to be set on the whole exam, time limits on a per question basis is not possible.

4.2 Protection against contestation

One of the biggest shortcomings of the softwares at hand is their way of handling connection errors. All of them rely on a stable internet connection. Loosing connection forces the student to wait until the connection is reestablished. This is especially problematic with questions that rely on time restrictions. As with local storage of answers, also timestamps can be saved. This assures the timely answering of questions. Students then could directly move on to answer the next question. Due to the lack of any offline capabilities, a students answers could be lost. Not providing a student with an adequate way of solving an exam this could lead to contestation.

4.3 Transparency

In terms of transparency LPlus, OpenOlat and Moodle all perform well. They provide a way to give feedback to specific questions. Students can review both their exam results and the given feedback. Still, direct contestation of corrections is not possible in app. Thus, this process must fallback on other means of communication. Ilias and BlackBoard are very limited in giving any feedback. The only feedback a student can get is the amount of points he achieved.

4.4 Protection against cheating

As mentioned above only LPlus and OpenOlat provide a way of creating partial open book exams. Time restrictions are a main tool for preventing several ways of cheating. Exams created with Ilias, Blackboard and Moodle thus can only rely on open questions or they risk cheating of students.

None of the systems provide a way for visual or auditory supervision. Students are only identified by the password and username. Such authentication can easily be shared and provides no sound

authentication. Lack of authentication thus opens up possibilities of students getting help from outsiders.

Another topic where the systems fail is the restriction of input. Copying and pasting is not disabled. Further, logging of keystrokes or similar measures are not enforceable.

Further, exams must be customizable in such a way, that students cannot jump between questions, cannot re-answer questions and can only see one question at a the time. This customization is found in Moodle, LPlus and OpenOlat. Both Ilias and Blackboard are limited in that regard.

Lastly, all of the tools allow for randomization of the question order in an exam. Adding, every tool provides a way of creating and maintaining a question pool. Questions can also be imported and exported, theoretically allowing for sharing between examiners. To allow for truly large question pools. In reality sharing at a platform magnitude must be envisioned. Non of the products can provide such a platform.

4.5 Protection of Data, Attributability and Integrity

With respect to *Protection of Data, Attributability and Integrity* all solutions provide user management with different roles and permissions. Adding all of the products are still actively developed and thus contiguously improved. The wide usage of the products at hand proves their capabilities to meet any legal data protection requirements. Further, no major security flaws in the software are known of.

4.6 Further considerations

As already mentioned above most of the systems currently in use are LMS. As we have shown above, they do not fulfil all the requirements needed for an exam. Problematically, due to their heavy integration with every part of an universities systems, LMS are not easily swapped out. Instead, it may be advantageous to use a standalone solution that uniquely focuses on providing e-exams.

4.7 Conclusion

[Grafik]

To conclude, non of the mentioned systems provide a examination system that meets all proposed requirements. The advantages e-exams provide in comparison to paper based exams make the development of a such an application compelling. As the Covid19 crisis has put even more pressure on the examination process the need for an artefact that meets all requirements has become even more urgent.

5 Designing the software artefact

In the following an overview of how a software artefact could implement all these design principle from a technical point of view will be made. Further, as a proof of concept two design principles are implemented in a working software prototype. This prototype helps determine software architecture as well as used technologies. First, general thoughts about data structure will be made, followed by an short introduction into used technologies. All these statements will be made with the exam requirements in mind.

As mentioned earlier open source code can allow for a more secure and transparent software. Consequentially the source code of this artefact can be found on github under the MIT open code licenceˆ([@https://github.com/jasperanders/XM], [@https://github.com/jasperanders/XM-Api])

5.1 Data structure and architecture

The artefact is classic client-server application. With a frontend (client), responsible for interaction with both examinees and examiners and a backend (server), responsible for data handling. The backend server runs on a [@Node.js] server, while the frontend is delivered via any modern browser, thus fulfilling the *equal treatment requirement*.

The client and server side communicate over a simple REST API. The advantage of a client-server application lays in the separation of the data and the end-user. Users do not have direct access to data but any read or write action must be made over and API. Here permissions can be checked and possible misuse inhibited. Second, as the user has no direct influence on the servers actions the sever can act as a source of truth. Data that was once committed to the server is now immutable. For example, the foraging of exam answers after submission becomes impossible. The API can also provide different kinds of endpoints to interact with for different kinds of users. Examiners for example can use one API endpoint to create exam questions, whereas students will get an error code if they try to interact with it.

One of the most important design considerations is the data model that is used to store and access exam and user data. There are four central instances: First, there is the question instance. This is

probably the most important instance. Especially with the creation of question-pools in mind, it is clear that question instance must live independent of any exam. A question consists of a title, the question type, the question text, the questions points and the questions time limit. Further each question has a question body, the shape of the question-body depends on the question type. For example the multiple-choice question-body consists of a reference to the question it belongs to and a selection of possible answers. For free-text questions currently no additional body is need, still to be consistent in data structure and to allow for later additions, free-text questions also have a body.

These questions can then be assembled into exams. Each exam contains an exam name users that are allowed to take part in a given exam and an exam date. Most impotently, an exam contains an Array of question ids, which constitute the exam content.

The third central instance is the user. Users can either be examinees or students with each having different roles. Students for example can not create exams or questions. Examiners on the other hand can do so. If any new data instance is create, e.g. a new question, the author automatically is set to the logged in user, creating the given instance. Further, users posses an email and password, a name and a unique identifier. This id is provided by the application but could also be an identifier provided by the testing authority. With user handling and automatic assignment of authorship we can assure the *attributability requirement* and to a large degree the *protection of data* and the *integrity requirements*.

The last key instance is the answer. For each question an examinee answers an answer instance is created. This instance contains a timestamp at which the question is started, a timestamp at which the question was answered and the question-id the answer is referring to. Additionally the answer object provides a flag that marks it as an master answer. Master answers are the correct answers, or in case of free-text questions provide a guideline of what is to be considered correct. Analogue to the question, the answer also bears an answer-body. This form of this body again is dependent on the question type. Multiple-choice question-body contain the selected answers, whereas a free-text question-body contains the given free-text answer. Additionally, any answer body contains a reference to the answer and to the fitting question via their respective ids.

The above provides an overview of the data structure as found in the database. As this app inhibits offline capabilities large portions of the above data structure can be again found in the frontend application. Of course the data is reduced to the data user, e.g a student, is allowed to see but references among question and answer remain identical. To realize the above mentioned offline capabilities the exam data is at all times persisted in the local storage of the browser. Data remains in this local storage until deleted by the app or intentionally removed by the user. If there is an internet connection the data in local storage and the data in the local storage remain the same. Should internet connection fail, will the exam continue as normal. Answers will be saved to local storage and at a later point in time sent to the server.

Depending on the circumstances students are taking their exams, examiners can adjust the degree of offline capabilities. Students could be allowed to take their complete exam in offline mode and only at the last question should sent their answers in. Alternatively students could be allowed to only have connection failures of a single question. With the handling of offline capabilities we can assure the *protection against contestation requirement.*

To provide a way to meet the requirement of **general validity**, and to fulfill parts of the design principles to assure **protection against cheating**, the frontend application must enforce per question time constraints. While the backend can review the actual used time the user interface must assist the student in taking as much time as they are allowed. The artefact achieves this by showing the remaining time and submitting the currently provided answer. This answer is then sent to the server. Students are thus forced to comply to respective time constraints, leaving them no room to accidentally miss allowed times.

5.2 Tech Stack

Both the server and the client side are written in a code language called JavaScript. It is the most popular language on [@Github https://octoverse.github.com/#top-languages]. JavaScript allows programmers to realize the complete technology stack with one language, making it a compelling language to write an application in. Besides many modern and popular libraries for web development are written in JavaScript. Some of these libraries also find use in this artefact, the most crucial being

[@React] and [@Express].

[@React] is “a JavaScript library for building user interfaces”. It uses structures that are devidable in reusable components. React makes it easy to create complex application instead of only simple websites. It was orignally created by [@Facebook] and finds it use in the tech-stacks of [@Uber], [@Airbnb], [@Netflix] and many more. Further the frontend uses the JavaScript superset TypeScript. TypeScript allows to define and check for complex types whereas JavaScript in general is typless. These types allow for more secure data handling making the application overall less prone to bugs. Lastly a ui-library is used to create a visually more pleasing experience faster.

[@Express] is a common library to create backend services with. It is lightweight and allows for the creation of both simple and complex APIs. As many backend-applications rely on the same structure generators for the common fundamentals exist. For this artefact I used the [@JonasScholz] Rest API generator. The express server also handles data storage, for this purpose a database is connected. The artefact uses a noSQL database called [@MongoDB]. [@MongoDB] does not store data in tables but in JSON like documents. As the JSON format is inspired by JavaScripts objects, the data structure used in the frontend part of the application thus directly translates to the data structure that is used to store the given data.

5.3 Future implementations

As the artefact is only a proof of concept and no market ready software it lacks some key features. Exams for example can be taken but there is no way of evaluating them through a user interface. As the correction of exam is not implemented at the moment the *transparency requirement* remains unmatched at the current state of the artefact.

Further the the video and sound surveillance is not an implemented feature jet. Protection against cheating thus remains also partially unmatched. As can be seen above, the video supervision only addresses the aspect of cheating by the use of outsider help. Looking at the design principle list, this supervision of students is the only measure against cheating that is not enforced.

A design principle mentioned is the usage of large question pools. In theory is there a way of creating

such a question pool in the current software artefact, still there are some limitations. To really generate large question pools, the power of large groups must be utilitized. Universities have created questions for numerous years but as questions leak or get published, these questions remain worthless for a single institution. Sharing them on the other hand opens questions up to new uses. To really enable the common sharing of questions two things must be achieved. First a common question format must be found. As question layout and design should be handled by the application the question is used in this standard should only contain data and no markup. The data structure of the question type of this artefact could theoretically serve such a purpose. Still, the data structure must firmly be evaluated and surely extended. Second a platform is needed to share questions. Such a platform should also take care some kind of quality assurance. At the time of this writing such a platform does simply not exist.

Lastly a note about usability and complexity of the artefact must be made. As this artefact serves as a minimal viable product, many of the user interactions are not suited for large amounts of data. Regarding the further development of the app usability and performance must continuously be evaluated and incrementally improved.