Group4 GmbH

gutenshare network

Project Summary

Alicia Rüegg
Kaspar Wolfisberg
Louis Leon Müller
Aleksandar Spasojevic
Arik Sidney Guggenheim

Table of Contents

1 Project Idea 4

1.1 Current situation 4

1.2 Idea 4

1.3 Customer value 4

1.4 State of the art / competitor analysis 5

1.5 Main use cases 6

1.5.1 Main use case 1: Uploading a new document 6

1.5.2 Main use case 2: Search, download, rate and comment a document 6

1.6 Economic viability 7

2 Analysis 9

2.1 Domain model 9

2.2 Use cases 9

2.2.1 UC 1: Search for a Document (“fully dressed”) 10

2.2.2 UC 2: Upload document (“casual”) 11

2.2.3 UC 3: Commenting a document (“casual”) 12

2.2.4 UC 4: Rating documents (“casual”) 12

2.2.5 UC 5: Browsing (“casual”) 13

2.2.6 UC 6: User statistics (“brief”) 13

2.2.7 UC 7: Sign up (“brief”) 13

2.2.8 UC 8: Log in (“brief”) 13

2.2.9 UC 9: Log out (“brief”) 13

2.3 Use case diagrams 14

3 Design 14

3.1 Architecture 14

3.1.1 Frontend 15

3.1.2 Backend 15

3.1.3 Database 16

3.1.4 Infrastructure 16

3.1.5 Architectural implementation 17

3.2 Design Class Diagram 17

3.3 Interaction Diagram 19

3.3.1 Design Sequence Diagram 19

3.3.2 Design Communication Diagram 21

3.3.3 Most Important Design Decisions 23

4 Implementation 24

4.1 Test report 24

4.2 Installation guide 24

5 Result 25

5.1 Objective Summary 25

5.2 Open Tasks 25

5.3 Further Development 26

6 Appendix 27

6.1 Table and figures 27

6.2 Glossary 28

6.3 User Manual 30

6.4 Project Management 35

6.4.1 Risks 35

6.4.2 Phase and Iteration Plan 37

6.5 Acknowledgments 40

1 Project Idea

1.1 Current situation

Creating and gathering learning resources can be a tedious and time-consuming task that students must deal with. To distribute this work load, students exchange notes, summaries and other learning resources. Acquiring these resources requires to get in touch with the right people and to have access to the right channels. The exchanging process usually happens through file sharing solutions that get the job done but are not specifically catered towards the problem and miss out on a lot of potential use cases.

1.2 Idea

To address the aforementioned inconveniences, we plan to create a web platform independent from lecturers and educational institutions, for students by students to share study related resources. Students upload documents they are willing to share and categorize them by allocating them to a general subject or a course within the curriculum of their study program. Besides the obvious search functionality, users will be able to easily locate documents through a sophisticated browser. While checking them out, they will be provided with suggestions of related media and related courses of different institutions.

As means of quality control on a per upload basis, a resource can be rated and discussed in the comment section by all users as well as reported as inappropriate or illegal. All document uploads are available through a user's profile as well as uploading and rating statistics to help others assessing the quality of their work.

1.3 Customer value

The primary target group consists of students looking for learning resources. By using the platform, they get an easy way to share their documents and have access to a vast collection of learning resources across the boundaries of educational institutions. Where your own school may be lacking material, students of other schools may help you out! Users will be able to efficiently locate these materials through an advanced searching and browsing capabilities. The document rating system allows you to jump right into material that was perceived as valuable by others. The main benefit is receiving a specialized tool serving one purpose only while providing the best possible experience doing so.

1.4 State of the art / competitor analysis

There are already lots of different Learning Management Systems (LMS), some of them are open source and free to use for universities while others are published under a commercial license. One of the biggest and most used LMS in Switzerland is OLAT and openOLAT which are maintained by the University of Zurich respectively by a spin off called Frentix. Other popular solutions are Ilias, Moodle and many more.
However, universities rely on their own implementation of the mentioned platforms and these are not designed to allow document sharing between different schools. Also, the content is maintained and controlled by teachers and professors. If students want to share their own documents they often need to use third party cloud systems like Dropbox or OneDrive.
There is already a platform in Switzerland which is maintained by students. This web service called Uniboard is not primary designed for document sharing but acts as a forum.

Taking all mentioned aspects into consideration, it is worth an effort to create a platform specifically made for students to share documents, as there is a big untapped market in Switzerland at the moment.

1.5 Main use cases

1.5.1 Main use case 1: Uploading a new document

- The user is signed in and on the upload page.
- The student chooses the document and appropriate title he wants to upload.
- The student has now two options:
    - On the one hand, the student has the possibility to choose a path from a collection of existing paths.
    - On the other hand, he can specify the path to his document on the platform. The student is requested to comply with the following guidelines: School/Department/Study course.
- The student submits his upload and is now redirected to the page where he can see his uploaded document.

1.5.2 Main use case 2: Search, download, rate and comment a document

- The user is signed in and on the browse page.
- As the uploaded files all have a prescribed guideline (School/Department/Study Course), the student browses now until he finds his desired file.
- The student has now three options:
    - If he wants to download an existing document, ha can retrieve the file.
    - If he wants to rate the document, he can choose to give between 1 and 5 stars. The application then recalculates the number of stars based overall ratings.
    - If he wants to comment on the document, he can type in a message to leave as a feedback.

1.6 Economic viability

Since the project is estimated to take 8 months of work and requires a team of 5 engineers with an average pay rate of 100 $\frac{\text{CHF}}{\text{year}}$, a realisation of the product will cost

5 software engineers (full time)	$5 \cdot 100 \frac{\text{CHF}}{\text{h}} \cdot 8 \frac{\text{hours}}{\text{day}} \cdot 20 \frac{\text{days}}{\text{month}} \cdot 8\text{months}$	640’000 CHF
Legal advice		50’000 CHF
Marketing		100’000 CHF
Infrastructure	PaaS provider, Rent	150’000 CHF
Development (first year)		**940’000 CHF**
Infrastructure	PaaS provider, Rent	150’000 CHF
Fix costs (FC) $\frac{\textbf{CHF}}{\textbf{year}}$		**150’000 CHF**
Marketing	5% of revenue	
R&D	extending service and developing new services, 20% of revenue	
Support	20% of revenue	
Partial Redemption	30% of revenue	
Variable costs (VC) $\frac{\textbf{CHF}}{\textbf{year}}$	25% of revenue for uncertainty and investors/shareholders	**75% of rev.**
Operational expenses $\frac{\textbf{CHF}}{\textbf{year}}$	Fix costs + variable costs	**FC + VC**

Table 1: Economic viability

One can see that operational costs depend on revenue. In order to quantify the former, a Break-even analysis is conducted.

Break-even analysis

Let’s assume an investment horizon of 5 years. For the product to repay all its costs within these 5 years, a revenue of **~783’250** $\frac{\textbf{CHF}}{\textbf{year}}$ would be required.

Year	**Revenue**	**FC+VC (75% of rev + FC)**	**Investment + Redemption**	**25% (uncertainty)**
0	CHF 0.00	CHF 940'000.00	CHF -940'000.00	CHF 0.00
1	CHF 0.00	CHF 0.00	CHF 0.00	CHF 0.00
2	CHF 783'250.00	CHF 737'437.50	CHF 234'975.00	CHF 45'812.50
3	CHF 783'250.00	CHF 737'437.50	CHF 234'975.00	CHF 45'812.50
4	CHF 783'250.00	CHF 737'437.50	CHF 234'975.00	CHF 45'812.50
5	CHF 783'250.00	CHF 737'437.50	CHF 234'975.00	CHF 45'812.50
Sum	CHF 3'133'000.00	CHF 3'889'750.00	CHF ~0	

Table 2: Break even analysis

Compared to the number of students in Switzerland, which is about 246’000, a revenue of this kind is realistic. Keep in mind, at this level a 25% would lead to 45’812 $\frac{\text{CHF}}{\text{year}}$ which is put on side for forecast and investors & shareholders; therefore, generate some yield. A minimum revenue of 600’000 $\frac{\text{CHF}}{\text{year}}$
is needed, if the project has to cover its costs. At this rate the project is for an investor not interesting anymore, since the yield would be negative and therefore the redemption not possible within 5 years.

2 Analysis

This chapter is about the project analysis and shows the domain model, the use cases and the use case diagram. The main goal of the analysis is to investigate requirements.

2.1 Domain model

In the following section (Figure 2) a visual representation of the conceptual classes is illustrated. The conceptual classes of the prototype are "Document", "User", "Rating" and "Comment" and are complemented with attributes. Associations join these classes. In this domain model only the conceptual classes of the main use cases are illustrated.

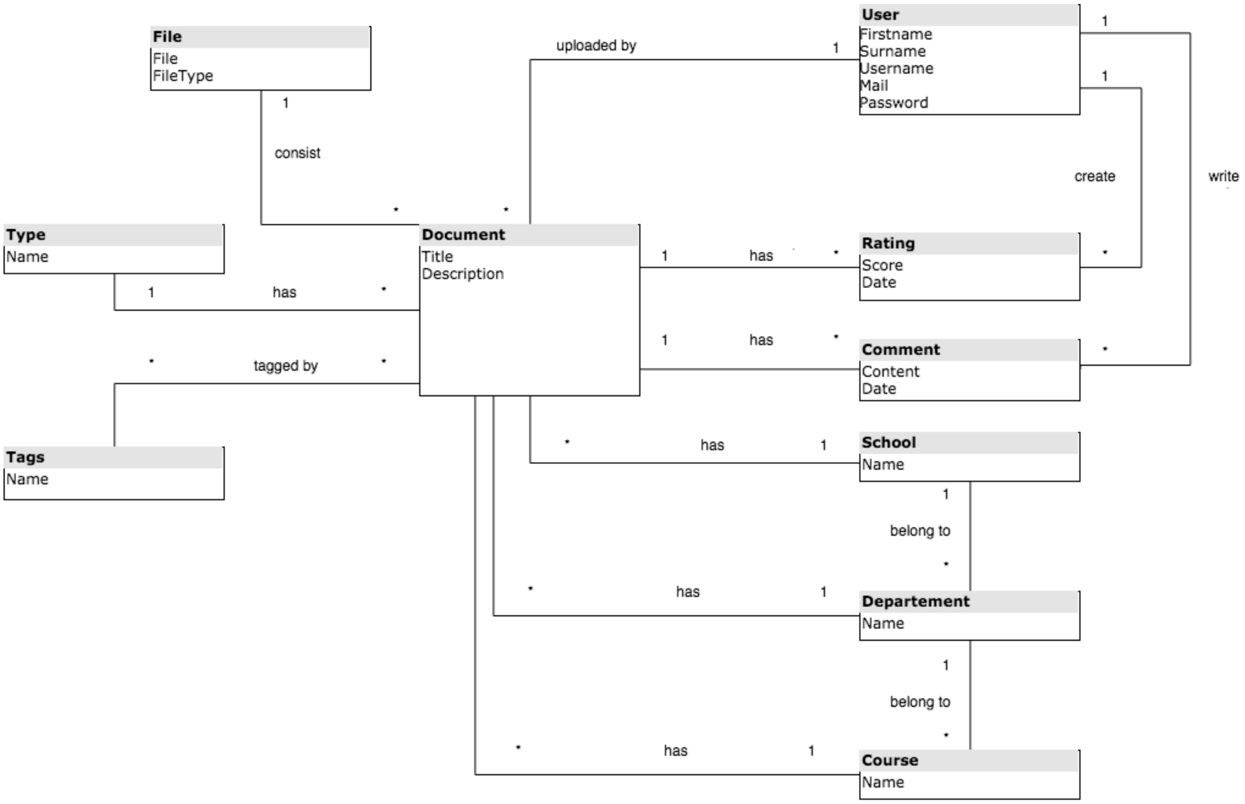


Figure 1: Domain model

2.2 Use cases

This chapter defines the gutenshare network's use cases. Following is a brief overview.

The main use case is described "fully dressed":

- UC 1: Search for a document

Other important use cases described “casual” are:

- UC 2: Upload a document
- UC 3: Comment a document
- UC 4: Rate a document
- UC 5: Browse documents

The below listed use cases are described “briefly”:

- UC 6: User statistics
- UC 7: Sign up
- UC 8: Log in
- UC 9: Log out

2.2.1 UC 1: Search for a Document (“fully dressed”)

2.2.1.1 Scope

The gutenshare network web application

2.2.1.2 Level

User-goal

2.2.1.3 Primary Actor

The gutenshare network user (A student)

2.2.1.4 Stakeholders and Interests

User: Wants a fast and reliable way to locate documents on the platform that match his search criteria.

Platform operator: Wants to satisfy customer expectations. Specifically provide accurate search results in a fast manner in a functional and visually appealing way.

2.2.1.5 Preconditions

The user must be logged in.

2.2.1.6 Success Guarantee

The user is provided with a list of documents that match the entered search criteria.

2.2.1.7 Main Success Scenario

- The user locates the search bar on the guentshare network web application.
- The user enters the search criteria.
- The user confirms the search.
- The platform redirects the user to the search results page.
- On the search results page the user is provided with a structured list of documents that matched the search criteria.
- The user is able to sort the results by relevance, rating score, number of downloads or alphabetically.
- The user is able to access the detail page of a document by clicking on an item in the search results list.

2.2.1.8 Extensions

1. The user wants to start a search without providing a search string of at least three consecutive alphanumeric latin characters.

- The button confirming the search will not be clickable and not trigger a search.

2. The user enters search criteria that yield no search results.

- The user gets redirected to the search results page where he is notified that no documents matched the search criteria.

2.2.1.9 Special Requirements

- After triggering a search, the user is provided with the search results within one second.

- Only search criteria provided in latin characters will be considered.

2.2.2 UC 2: Upload document ("casual")

2.2.2.1 Main success scenario / basic flow

1. Student enters the document upload page.
2. Student selects a document the file to the page.
3. Student can set a title and choose a course and a school to which the file belongs.
4. Student can categorize the document. Categories are predefined (i.e. exam or summary).
5. Student can add optional tags, description, school and course to describe the document further.
6. The file and the metadata gets written to the storage and database.
7. When the process is finished the user gets a confirmation in form of a pop up.

2.2.2.2 Extensions / alternative flows

- User cancels the upload or closes the browser

    1. The document will not be uploaded.
    2. Tags and metadata will be deleted from the database.
- Internet connection is interrupted
    1. The user gets an error message and all data will be stored inside the browser.
    2. If the website can reconnect to our servers the process will be continued.
    3. In case of a permanent connection failure or if the user closes the browser the data in the browser will be deleted and the student has to start with a new upload.
- File is too big or has an invalid file type
    1. If the file is to big (more than 40 MB) or if it has an extension which is not allowed (i.e. video file) the user will get an error message and the whole process will be stopped.

2.2.3 UC 3: Commenting a document (“casual”)

2.2.3.1 Main success scenario

A user wishes to express an opinion on a document that he downloaded from the gutenshare network. The user locates the detail page of said document and uses the comment section to write a comment. After submitting the comment, it becomes visible to all users of the gutenshare network on the detail page of this document annotated with a time stamp and the author of the comment.

2.2.3.2 Alternate scenarios

If the user has no internet connectivity an error message will be displayed in the corner. The comment will not be saved locally and the user must submit the comment again.

2.2.4 UC 4: Rating documents (“casual”)

2.2.4.1 Main success scenario

A student had a look at a document. Based on the quality and the informational content the student can assign a rating of one to five stars to the document. This rating will influence the statistics of the user who uploaded the document.

2.2.4.2 Alternate scenarios

- If the user has no internet connectivity an error message will be displayed in the corner. The rating will not be saved locally and the user has to rate the file again.
- If the user wants to rate a document, which she already rated the new rating will override the previous one.

2.2.5 UC 5: Browsing ("casual")

2.2.5.1 Main success scenario

The user starts a search by typing in keywords of the content he wants to find. A result page is displayed with a header section to provide additional search criteria's like tags and the type of the document. Additionally, he can curtail the results by selecting one or multiple schools, departments and courses in a table that lists the relevant documents. Below a header of the matching documents is listed including title, rating, type and tags.

2.2.5.2 Alternate scenarios

- The keywords provided hit no matches. The results page will display content most likely related to the searched content.
- The search cannot get executed because the server is not available due to network issues. A timeout notification will be displayed.
- The user has filtered the search too narrow and no results will be displayed. A notification will suggest removing certain filters to widen the search result.

2.2.6 UC 6: User statistics ("brief")

User statistics can be reviewed on a user's profile page. These statistics contain number of uploaded documents, number of written comments and the average rating amongst all documents uploaded by this user.

2.2.7 UC 7: Sign up ("brief")

To sign up as a user on gutenshare one has to choose a unique username, provide the full name, an e-mail address and a password. After the registration a confirmation link sent to the mail address will complete the process.

2.2.8 UC 8: Log in ("brief")

A user visits the gutenshare network login page. She enters her user name and the corresponding password correctly. The login service checks the correctness of the credentials and grants the user access to the gutenshare network by redirecting her to her profile page.

2.2.9 UC 9: Log out ("brief")

The logged in user is can choose to log itself out by clicking the log out button on each page of the gutenshare network. A web request is then sent to the server, and a confirmation notification ensuring that the log out has been successfully commited.

2.3 Use case diagrams

In the following illustration (Figure 2) the use cases are represented in use case diagrams. It shows the names of the use cases in relation to the actors. As it's visible in the illustration, there are two main actors: the anonymous user and the logged in user, also known as student, whereas the student inherits from the anonymous user.

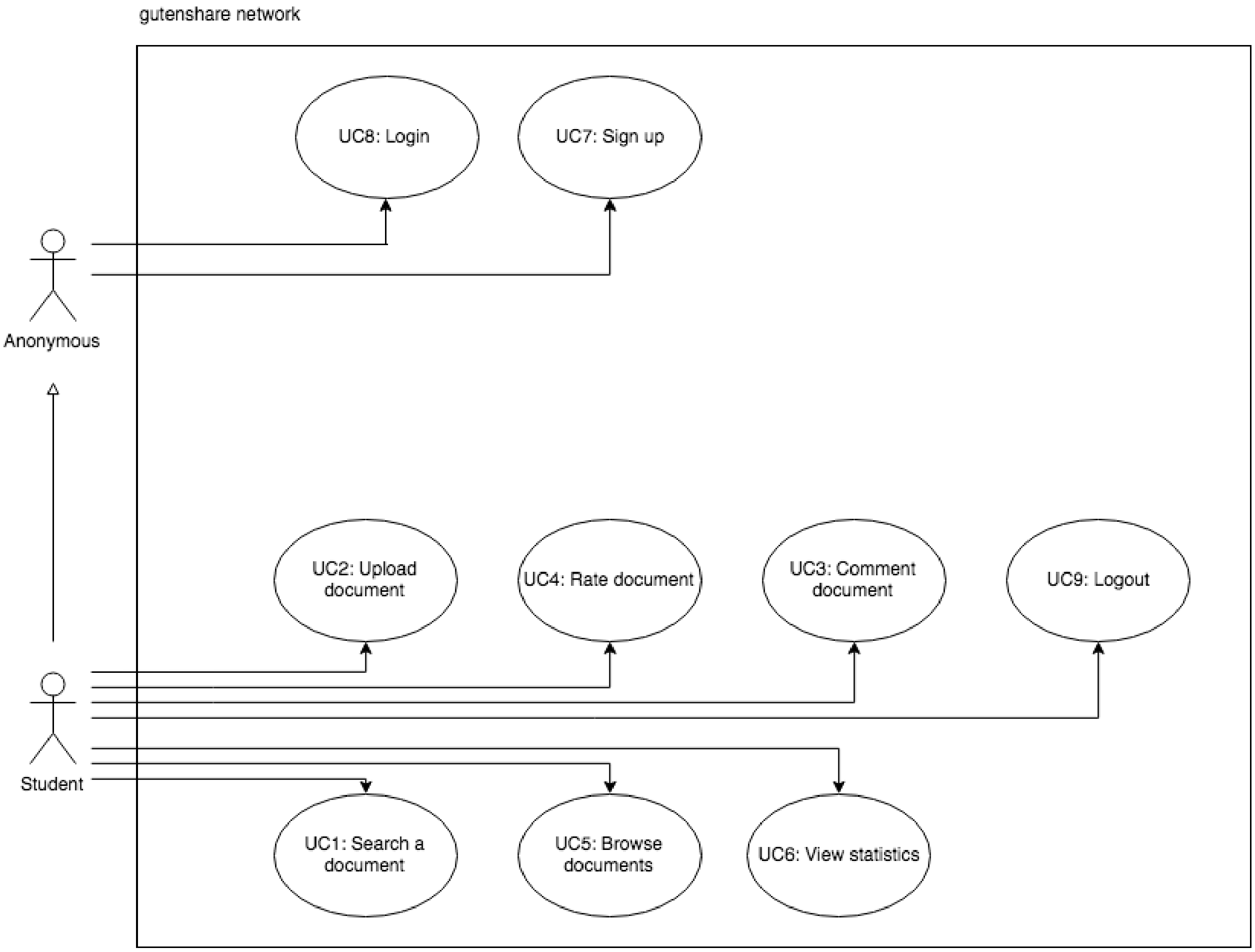


Figure 2: Use case diagram

3 Design

3.1 Architecture

In this subchapter the selected architecture is described and the decisions for the choice are explained. In the figure 3 an overall architectural diagram is present, whereas the different components are described in the subchapters.

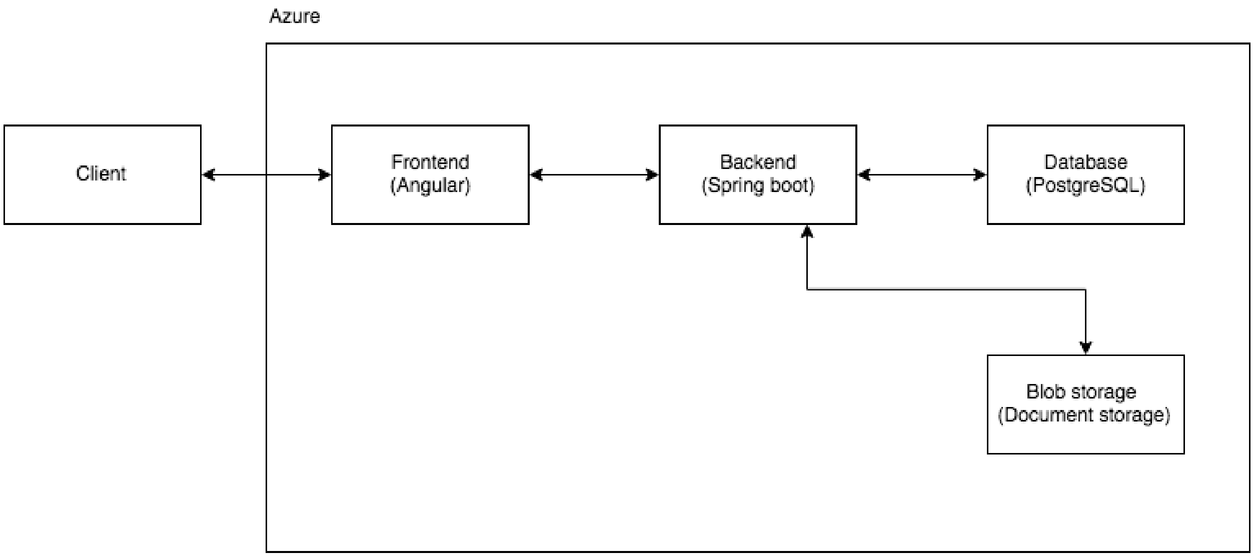


Figure 3: Architecture

3.1.1 Frontend

Figure 4: Angular

We decided to build a web application. While every member of our team has JavaScript knowledge, we lack an experienced frontend developer. This means we got very little bias towards any framework and were looking for something that will not restrict us in the prototype development process. Because it is well documented and established we landed on Angular5.

3.1.2 Backend

Figure 5: Java

Since all members of our development team have at least some expertise in Java it was the obvious choice as a backend language. It is powerful and there are various MVC frameworks to choose from. We decided to use Spring Boot because two developers have already worked with it and it provides a streamlined development process and requires writing very little boiler plate code. It provides most of the required features (ORM mapping, MVC, …) out of the box and alleviates a lot of tedious configuration work.

As for the backend design we got introduced to the onion pattern by our chief technology officer. The heart of the onion pattern is the core module. It contains all business objects and logic. The core is completely isolated and only accessible from the API layer which acts as a wrapper around it. The rest of the application communicates with this API layer, which consists of DTOs and service classes. The final piece is the infrastructure layer represented by the web module containing the RESTful API endpoints in controller classes. This approach ensures strict encapsulation and modularity. Additional modules that are potentially needed in the future can easily be plugged into the API layer without affecting any business logic.

Figure 7 Spring boot logo

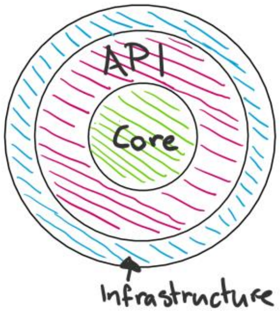


Figure 6 Onion architecture illustration

Since we have limited time for the realization of our prototype we decided to use a simplified implementation of the onion architecture. The main difference is that our core module won't be fully technology independent. Because we already decided to use a relational database, our core will also include the JPA part to connect to the DB.

This will give us the possibility to use any relational database but requires us to do a lot of refactoring if we want to switch to another storage type.

3.1.3 Database

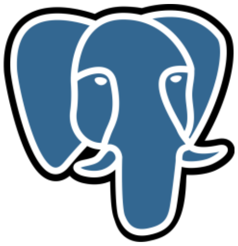


Figure 8: PostgreSQL

In correspondence with our development team's knowledge we decided to use a relational database. In order to avoid unnecessary licensing costs we chose the well documented open source PostgreSQL database.

3.1.4 Infrastructure

For deployment and integration, we are using Microsoft Azure Cloud Services. Our team leader is professionally engaged in this technology and it promises easy to use Infrastructure.

Figure 9: Azure

The gutenshare web application will be deployed in the Azure cloud for elastic resource allocation and scalability to ensure minimized operational costs and enabling the growth of the application as needed. To reduce technological dependencies and guarantee an up to date infrastructure, most services will run as a platform as a service.

Hence the backend as well as the frontend will be hosted as web apps in Azure, with API interfaces to the surrounding environments. The PostgreSQL database will be obtained as a managed service as well.

All keys used in the application will get stored in the Azure Key Vault which is provided as a hardware HSM key storage module to secure the API communication between the services.

The gutenshare web application heavily relies on an efficient and scalable storage service. Azure blob storage provides a georedundant, fast and scalable storage of all documents uploaded to the network.

3.1.5 Architectural implementation

The application is a multi-module java application. This means that each layer of the onion architecture forms its own module and can be recognized by the uniform "gutenshare-" prefix. This includes:

- gutenshare-core: This module includes all our domain classes and JPA interfaces
- gutenshare-api: Contains all services and DTO's
- gutenshare-web: Is part of the infrastructure such as controller, document storage, security for login
- gutenshare-frontend
- gutenshare docker: Docker files for local testing and deployment (usually not part of the onion architecture)

The whole project is built with Maven, where the POM files list the right modules, the right dependencies, and the right libraries.

3.2 Design Class Diagram

In this illustration you see the design class diagram. In the center of this diagram is the document class since it has a direct or indirect connection to all the other classes.

For every class you see its properties and its public methods to implement all of our main use cases.

In addition, to the classes there are two enumerations. One enumeration holds the possible rating values while the other one describes the document types.

Note that in this design class diagram only classes belonging to the core are represented.

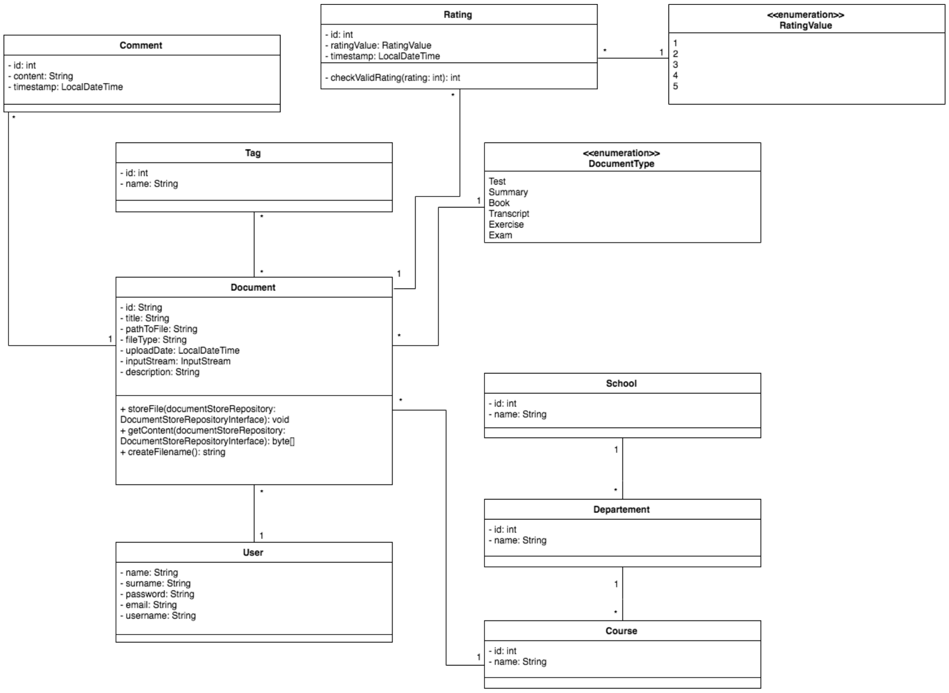


Figure 10: Design class diagram

3.3 Interaction Diagram

This chapter presents the interaction diagrams of the most important use cases of our prototype. The use cases were presented either as communication (browse, comment, rate, search) or sequence diagrams (registration, upload).

Since the user login and logout is handled by spring security and it is therefore not in our application context, they were omitted. It should be mentioned that a successful login sets a token, which is sent with every request. If the user logs out, the token is destroyed and the user must first log in again to communicate with the gutenshare platform.

On the following interaction diagrams, you can see the process through all layers, i.e. from the infrastructure, through the API until the core is reached. Where the documents are stored in the end is a matter of implementation and therefore not illuminated in these diagrams. Furthermore, only the public methods are visible in the following illustrations.

Additionally, some design and programming patterns were included in our application. Firstly, we have a design pattern, more specific the builder pattern, that we use in the backend to build our domain objects. Furthermore, we have a technical pattern called dependency injection, which is used through all layers (infrastructure, API and core).

3.3.1 Design Sequence Diagram

Sequence diagrams are time based interaction diagrams. The starting point is always a user, who interacts with the gutenshare platform.

3.3.1.1 User Registration

If the user wants to register, the “create” method is called on the login-controller. This method requires the first name, the last name, a password, an email and a username. Afterwards the “save” method will be executed at the user-service and the same method will be forwarded to the user-repository. If this username is already taken, a HTTP-Status 409, and if the registration was successful, a HTTP-Status 200 will be returned by the login-controller.

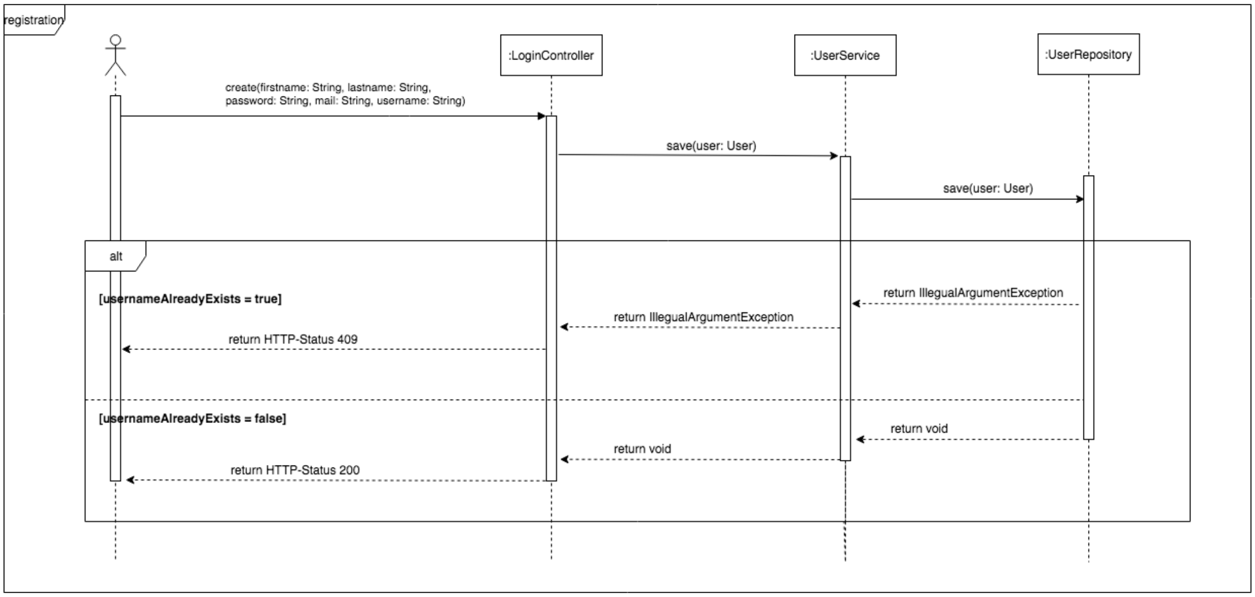

registration
:LoginController
:UserService
:UserRepository
create(firstname: String, lastname: String, password: String, mail: String, username: String)
save(user: User)
save(user: User)
alt
[usernameAlreadyExists = true]
return IllegualArgumentException
return IllegualArgumentException
return HTTP-Status 409
[usernameAlreadyExists = false]
return void
return void
return HTTP-Status 200

Figure 11: Design sequence diagram for user registration

3.3.1.2 Upload a Document

Figure 12 shows the uploading process. If a user wants to upload a document, the “storeDocument” method will be called on the document-controller. As parameters the title, document type, school, department, course, description, document, user and a list of tags have to be passed to the controller. Afterward, the “storeNewDocument” method will be executed on the document-services, which saves the document by calling the “save” method on the document-repository. If the upload was successful, a HTTP-Status 200 will be returned by the document-controller. Otherwise a HTTP-Status 500 will be sent.

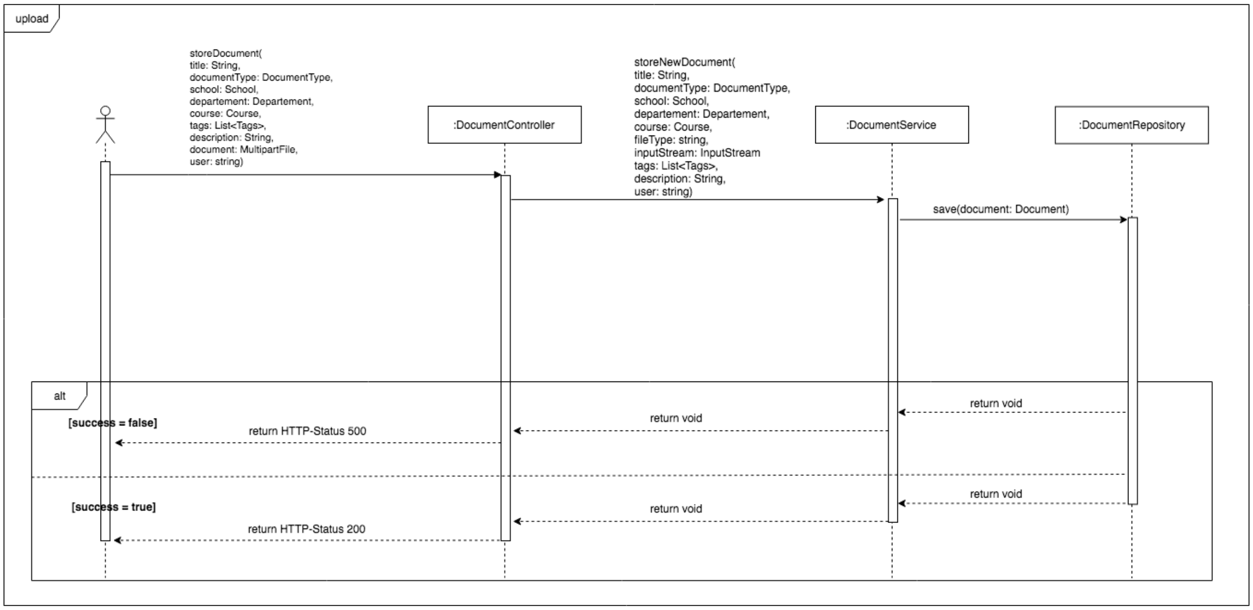


Figure 12: Design sequence diagram for uploading a document

3.3.2 Design Communication Diagram

Design communication diagrams help to illustrate object interactions. The following section displays the object interaction of the remaining gutenshare use cases with the help of design communication diagrams.

3.3.2.1 Comment a Document

The scenario of adding a comment to a document is illustrated in the following design communication diagram (Figure 13). If the user wants to add a comment, the "addComment" method will be called on the document-controller. The metadata will then be forwarded to the Service and afterwards to the comment-repository. If the storing was successful, a HTTP-Status 200 will be returned. Otherwise there will be a HTTP-Status 404.

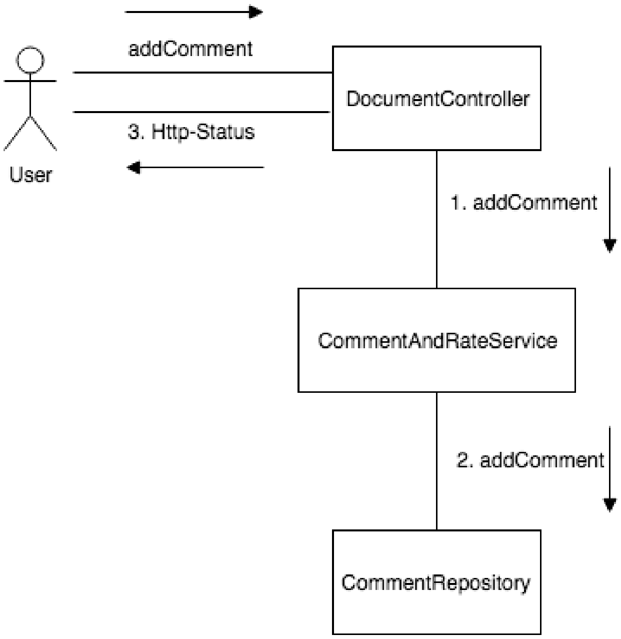


Figure 13: Design communication diagram for commenting a document

3.3.2.2 Rate a Document

Figure 14 describes, the rate functionality in the form of a communication diagram. When a user wants to add a rating, the "addRating" method is executed on the document controller. Afterwards, the "addRating" method will be called on the commentAndRating-service and be forwarded to the rating-repository. If the submission was successful, the HTTP-Status 200 will be returned. If an error occurs, a HTTP-Status 404 will be given back to the user.

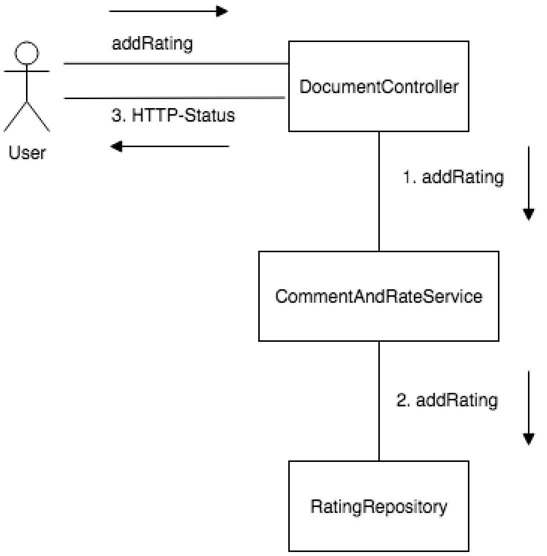


Figure 14: Design communication diagram for rating a document

3.3.2.3 Browse Documents

The browsing process is shown in Figure 15. If the user wants to browse the document archives, he can filter by schools, departments and courses. These metadata are passed to the search controller by the "browseDocument" method, whereas the controller calls the "getDocumntsFromBrowse" on the document service. The documents, which match the search criteria will then be returned from the document repository over the document service and the search controller to the user.

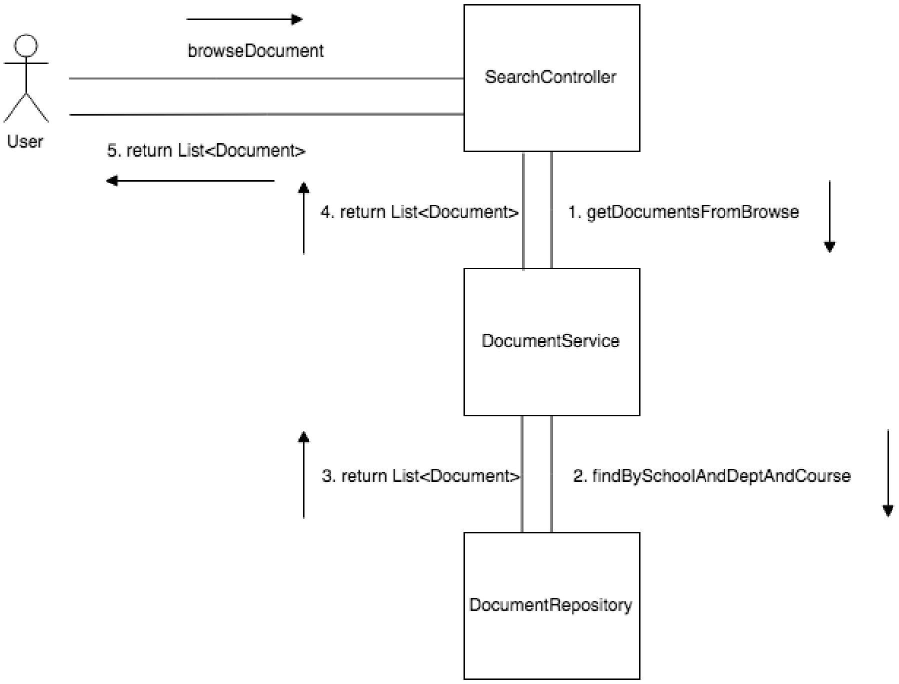


Figure 15: Design communication diagram for browsing documents

3.3.2.4 Search for Documents

The process of searching for documents is displayed in Figure 16. If the user searches for a document, he types in a query and the *"searchDocuments"* method is called with the query on the document controller. The document controller then calls the method “getDocumentsFromSearchQuery” on the document service. Then the document service calls the “findAllBySearchQuery” on the document repository. Latter returns a list of documents which will be forwarded to the user.

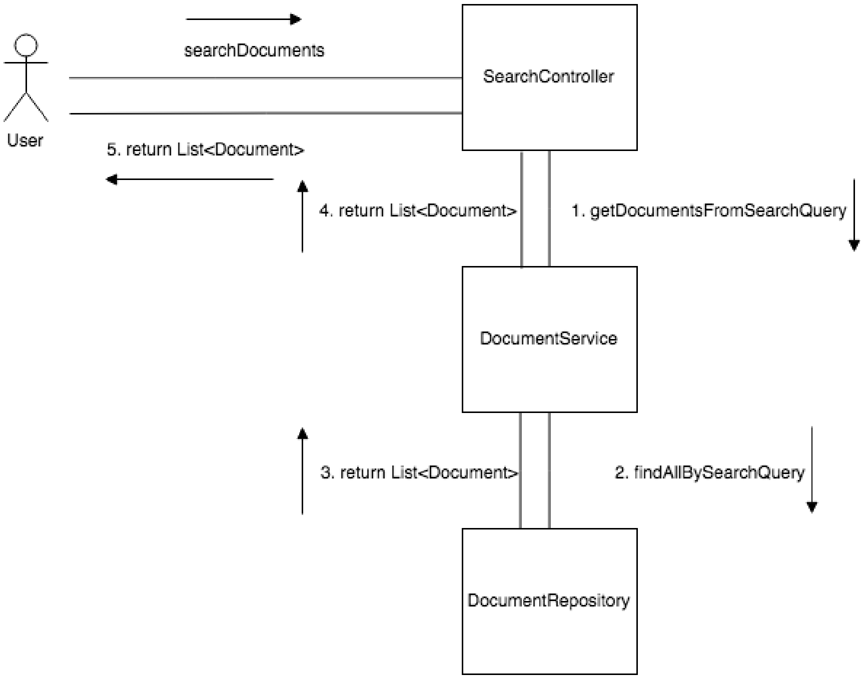


Figure 16: Design communication diagram for searching after documents

3.3.3 Most Important Design Decisions

Given the scope of the app it is highly recommended to make design decisions concerning security and accessibility. Why this section provides information about two technologies which are of high importance to the app and had a big impact on the latter.

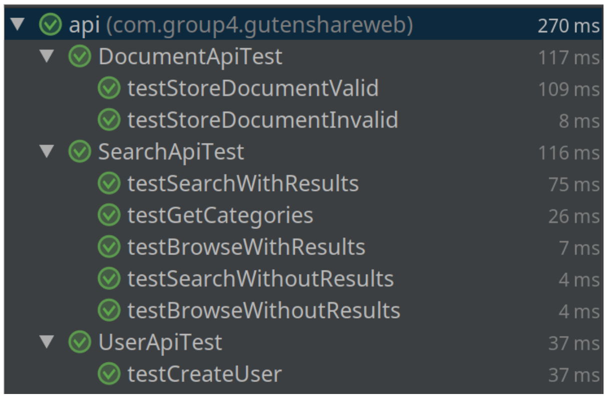
The team decided early on to use a sophisticated framework such as Spring and with that all the freedom in terms of high standard security libraries and workflows. For example, the login feature uses JSON Web Tokens (JWT) for keeping the user logged in. Additional settings such as validity, signature, cryptographic options or authorization make the login highly configurable and come for free when Spring Security is used. The design decision on the framework was important and had not only the benefit of being helpful in the configuration and ease of use, but also in the interaction with the community around Spring.

Another example where a design decision was important to the gutenshare network can be found in the API. Creating a RESTful service has allowed the team to decouple the frontend from the backend. This allows different areas of the app to evolve independently and do not require dedicated software engineers to program e.g. Java Server Pages in order to make the app usable from within the world wide web. In fact, providing a RESTful service has become ‘state of the art’ in the open source community.

4 Implementation

The following chapter documents the most important information concerning the implementation of the gutenshare network web application. An overview of the conducted tests will be outlined as well as a brief installation guide on how to implement and deploy the application.

4.1 Test report



```
▼ ✓ api (com.group4.gutenshareweb)          270 ms
  ▼ ✓ DocumentApiTest                       117 ms
      ✓ testStoreDocumentValid              109 ms
      ✓ testStoreDocumentInvalid              8 ms
  ▼ ✓ SearchApiTest                         116 ms
      ✓ testSearchWithResults                75 ms
      ✓ testGetCategories                    26 ms
      ✓ testBrowseWithResults                 7 ms
      ✓ testSearchWithoutResults              4 ms
      ✓ testBrowseWithoutResults              4 ms
  ▼ ✓ UserApiTest                            37 ms
      ✓ testCreateUser                       37 ms
```

Figure 17: Test Reports

To assure functionality of the implementation to the highest possible degree and aide with future code extension and refactoring, the gutenshare backend application has been thoroughly tested with the help of several state of the art testing frameworks (JUnit, Mockito, MockMVC, Jasmine, Karma). Since our core module contains little business logic and testing spring boot/JPA functionality is futile, unit tests were ignored in favor of integration tests. The test coverage extends to all RESTful API endpoints and implicitly tests the core module along the way. As soon as the project is greenlit for a full implementation, the front end will be reworked according to the model of a user experience designer. Because of the fragile nature of front end test and in anticipation of this redesign, front end testing has been limited to key features.

All tests have successfully passed and are rerun before each deployment of the master branch. All developers and team members have verified the use cases as well as the additional requirements defined in the project sketch. The following results concerning the additional requirements have been reported:

- Http traffic is not encrypted. A SSL certificate has not yet been acquired.
- The user login session is nevertheless secured via json web token.
- The file storage is highly available redundantly persisted in the Azure cloud.
- All components seamlessly work with each other.
- The usage of the gutenshare network is reliable and fast.

4.2 Installation guide

For an installation from scratch, for example in a new Azure subscription, a blob storage account with a blob container must be deployed. A relational database is required as well. For development and deployment, we used a PaaS postgresql server on Azure. An initial configuration of the Azure Web Apps and docker image registry will complete the Azure basic configuration. All connection settings and passwords have to be set in application.properties files in the backend and in the environment.ts files in the frontend.

After committing changes to the code repository, the application can be prepared by rebuilding the docker images through the provided dockerfiles. The images can then be pushed into the Azure docker image registry and the fully automated update process of Azure resources can be initialized.

5 Result

5.1 Objective Summary

The gutenshare network has been successfully deployed in a stable prototype release. It can be used by a first user group and further developed by the Group4 GmbH team. In the following section a detailed overview of the implemented and omitted features will be illustrated.

The use cases that have been assessed in the evaluation phase, where implemented in the past two months with the exception of the user's statistics use case UC 6, mentioned in chapter 2.2. Nevertheless, the gutenshare application is ready for use. Students or users in general, can do an initial registration, login to the network, search and browse for resources and share their own documents onto the gutenshare network. Documents are stored in a blob storage cloud container and enriched with metadata for enhanced search capabilities.

As the gutenshare team has advanced in the implementation, the search functionality has been kept on a rudimentary level, to lower complexity and ensure delivery on time. A search against tags and types as well as an elastic search is therefore currently not possible. The objective of an advanced search had to be replaced with the implementation of a basic search.

Regarding the infrastructural goal, the application has been deployed to the Azure Cloud and is ready for scaling. A last technology change has been made in the early implementation phase from the Azure WebApps Service to a Azure Kubernetes Cluster for faster deployment and better scalability. The objective of having a scalable and elastic infrastructure in the cloud was not affected.

Taking all before mentioned facts and objectives into account, the overall objective of the prototype project has been successfully reached and the gutenshare network is ready to advance into the next phase to evolve into a mature application to be used around the globe.

5.2 Open Tasks

As shown in the preceding section the search functionality lacks several advanced search capabilities. The search must therefore be improved in the following ways:

- The search queries must also target the tags and type metadata of a document. Estimated effort for implementation: 8 hours.
- The implementation of an elastic search database, which inspects the content of uploaded documents will enhance the search results extensively. It would also be possible to implement a partial search. Estimated effort for implementation: 24 hours.

The user's statistic profile was omitted because of lack of time. For implementing a user profile with the values about the number of downloads, uploads and ratings, as well as editing the user credentials, an effort of 32 hours is estimated.

Infrastructure wise there are several security relevant tasks open. As this prototype is not yet for a productive environment the project management has decided to fix them not until the productive release. Following security issues must be fixed in a productive environment:

- Enabling SSL encryption on frontend as well as backend servers. Estimated effort for implementation: 8 hours.
- Disabling public IP address on backend servers. Estimated effort for implementation: 4 hours.
- Store all secrets in the Azure Key Vault. Estimated effort for implementation: 8 hours.
- File type integration checks and Antivirus implementation. Estimated effort for implementation: 16 hours.

5.3 Further Development

Before a go-to-market release of the gutenshare network, the product must be branded. This can be done in collaboration with a design and marketing company which can also provide content for advertisement.

Besides the general release of the product, the school collaboration program must be further developed to ensure a stable source of income and first user usage at the schools and universities.

As the gutenshare network will grow the users must get a personalized user profile with statistics in a messaging system inside the gutenshare network. This will enhance the usability of the sharing platform. To support users and to maintain the content, an additional user role “Administrator” will have to be created.

In a next step the marketplace feature, which has been mentioned in the first project sketches can be reevaluated to open new customer segments.

6 Appendix

6.1 Table and figures

Table 1: Economic viability *7*
Table 2: Break even analysis *7*
Table 3: Product risks *35*
Table 4 Project risks- *36*
Table 5: phase and iteration plan *37*
Table 6: Iteration #1 .... *37*
Table 7: Iteration #2 .... *38*
Table 8: Iteration #3 .... *38*
Table 9: Iteration #4 .... *38*
Table 10: Iteration #5 .... *39*
Table 11: Iteration #6 .... *39*
Table 12: Iteration #7 .... *40*

Figure 1: Domain model *9*
Figure 2: Use case diagram *14*
Figure 3: Architecture *15*
Figure 4: Angular *15*
Figure 5: Java *15*
Figure 6 Onion architecture illustration *16*
Figure 7 Spring boot logo *16*
Figure 8: PostgreSQL *16*
Figure 9: Azure *16*
Figure 10: Design class diagram *18*
Figure 11: Design sequence diagram for user registration *20*
Figure 12: Design sequence diagram for uploading a document *20*
Figure 13: Design communication diagram for commenting a document *21*
Figure 14: Design communication diagram for rating a document *22*
Figure 15: Design communication diagram for browsing documents *22*
Figure 16: Design communication diagram for searching after documents *23*
Figure 17: Test Reports *24*

6.2 Glossary

In this chapter, the most important terms in perspective to the gutenshare network are explained and defined.

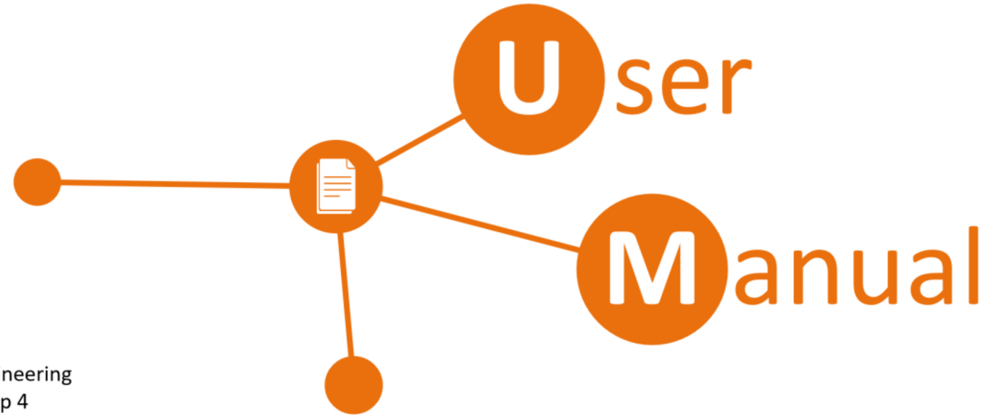
TERM	*DEFINITION AND INFORMATION*	*ALIASES*
USER	A person who participates on the gutenshare network platform.	Student
USER PROFILE	A page where the user can change his personal information.	User account
COMMUNITY	All participants on the gutenshare network platform.	Users
DOCUMENT	A virtual representation of a physical paper that the user has uploaded on the platform.	File
SCHOOL	Metadata of a document. Defines the institution name.	
DEPARTMENT	Metadata of a document. Defines a division of an institution.	
COURSE	Metadata of a document. Defines a program of instruction in a department.	
TYPE	Metadata of a document. Gives a document a meaning. Documents with same types share a particular characteristic that causes them to be regarded as a group.	
RATING	Shows the acceptance and value of a document for the community.	
NAME	The official title of a document	Title
COMMENT	A written text from a user regarded to a document.	
TAG	A user defined keyword to categorize documents by its characteristics.	Synonym
TO BROWSE	To filter and navigate the document archive.	
TO SEARCH	To look for a specific document.	

MEDIA	The physical equivalence to a document.	Book, paper, article

6.3 User Manual

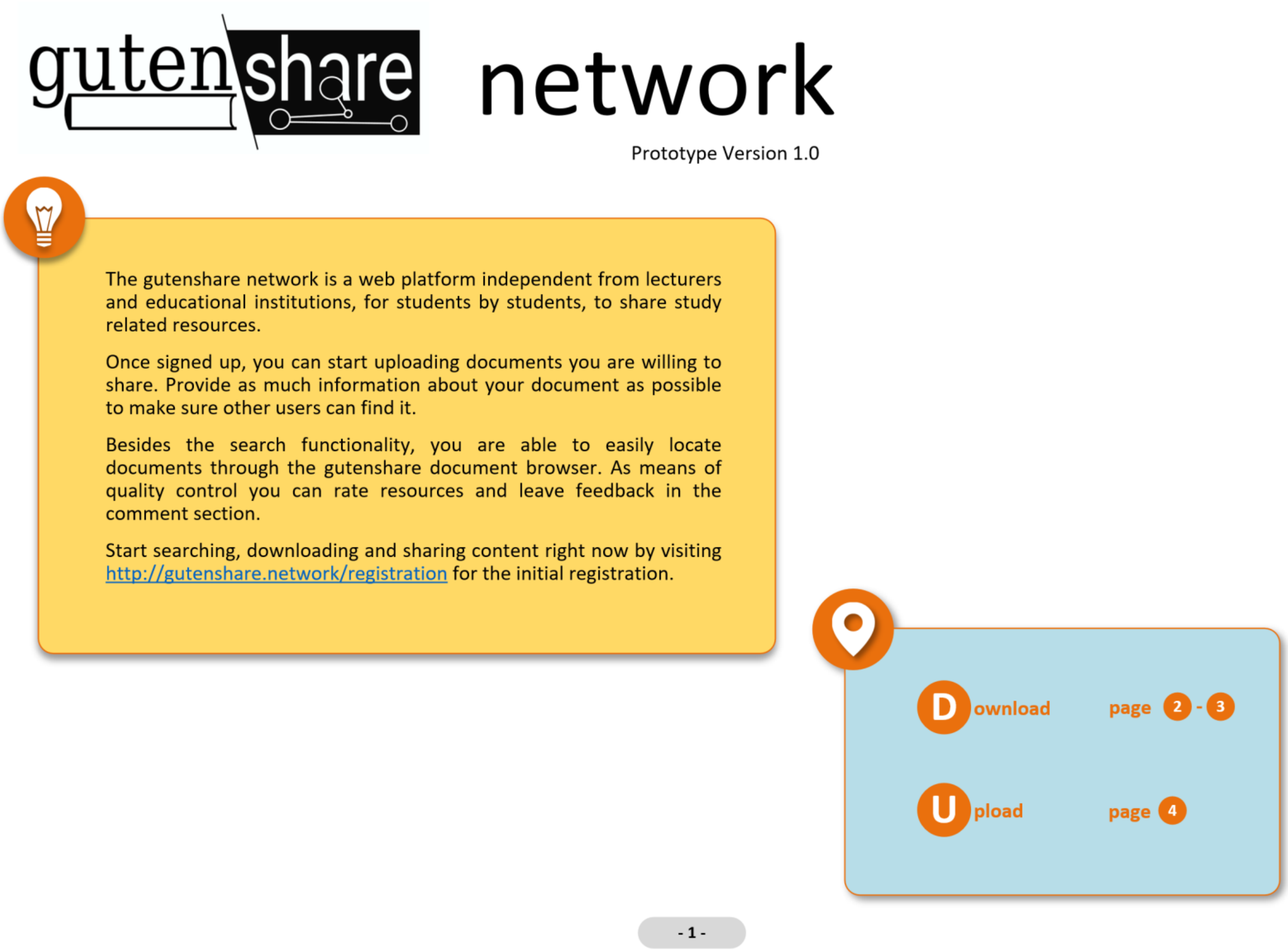
Group4 GmbH

gutenshare network



ZHAW School of Engineering
FS 2018 - PSIT3 Group 4
11.05.2018

Alicia Rüegg
Kaspar Wolfisberg
Louis Leon Müller
Aleksandar Spasojevic
Arik Sidney Guggenheim



gutenshare network

Prototype Version 1.0

The gutenshare network is a web platform independent from lecturers and educational institutions, for students by students, to share study related resources.

Once signed up, you can start uploading documents you are willing to share. Provide as much information about your document as possible to make sure other users can find it.

Besides the search functionality, you are able to easily locate documents through the gutenshare document browser. As means of quality control you can rate resources and leave feedback in the comment section.

Start searching, downloading and sharing content right now by visiting http://gutenshare.network/registration for the initial registration.

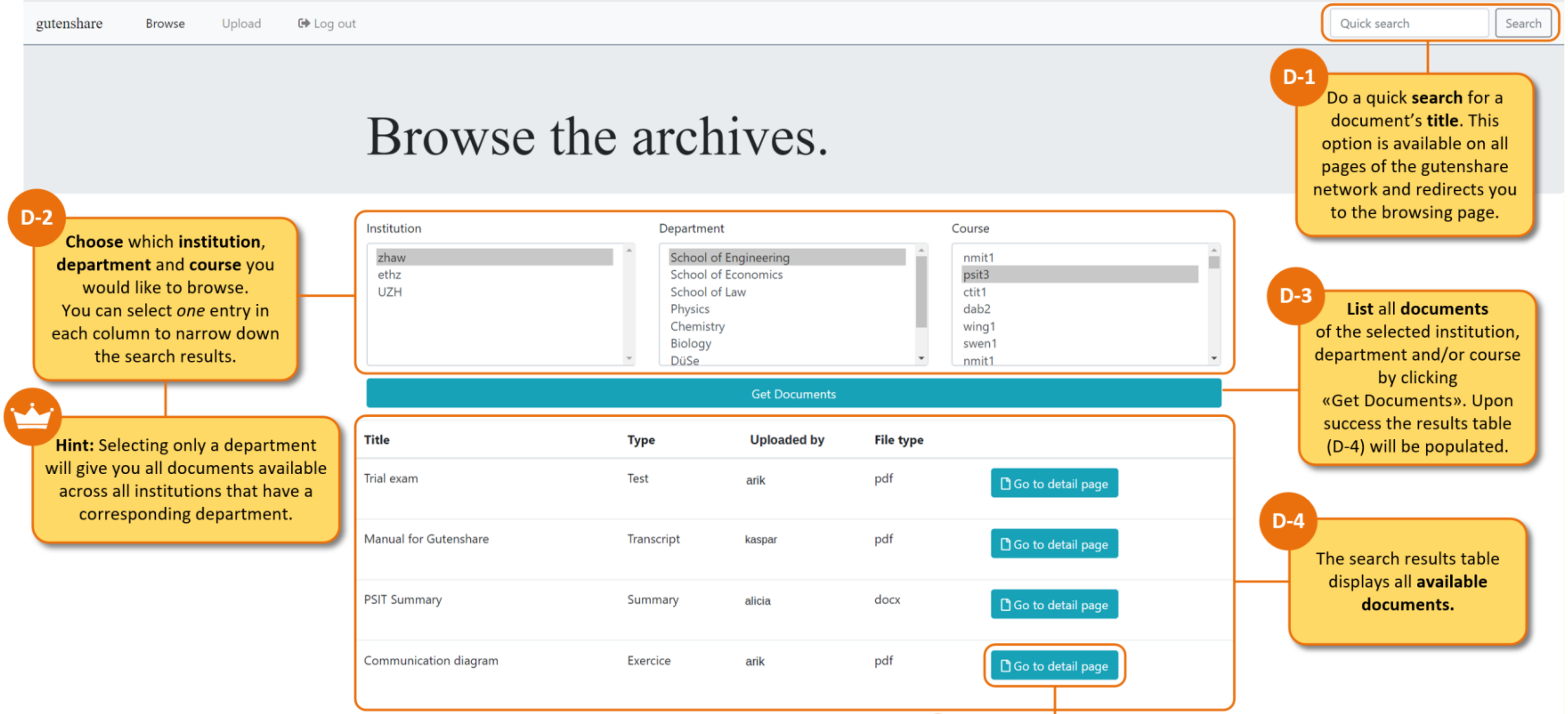
Download	page 2 - 3
Upload	page 4

- 1 -

Download - Browse

Prerequisites
User has signed up and is logged in.

Goal
Retrieve a list of downloadable documents according to your search criteria.



Browse the archives.

Title	Type	Uploaded by	File type	
Trial exam	Test	arik	pdf	Go to detail page
Manual for Gutenshare	Transcript	kaspar	pdf	Go to detail page
PSIT Summary	Summary	alicia	docx	Go to detail page
Communication diagram	Exercice	arik	pdf	Go to detail page

D-1 Do a quick **search** for a document's **title**. This option is available on all pages of the gutenshare network and redirects you to the browsing page.

D-2 **Choose** which **institution**, **department** and **course** you would like to browse. You can select *one* entry in each column to narrow down the search results.

Hint: Selecting only a department will give you all documents available across all institutions that have a corresponding department.

D-3 **List** all **documents** of the selected institution, department and/or course by clicking «Get Documents». Upon success the results table (D-4) will be populated.

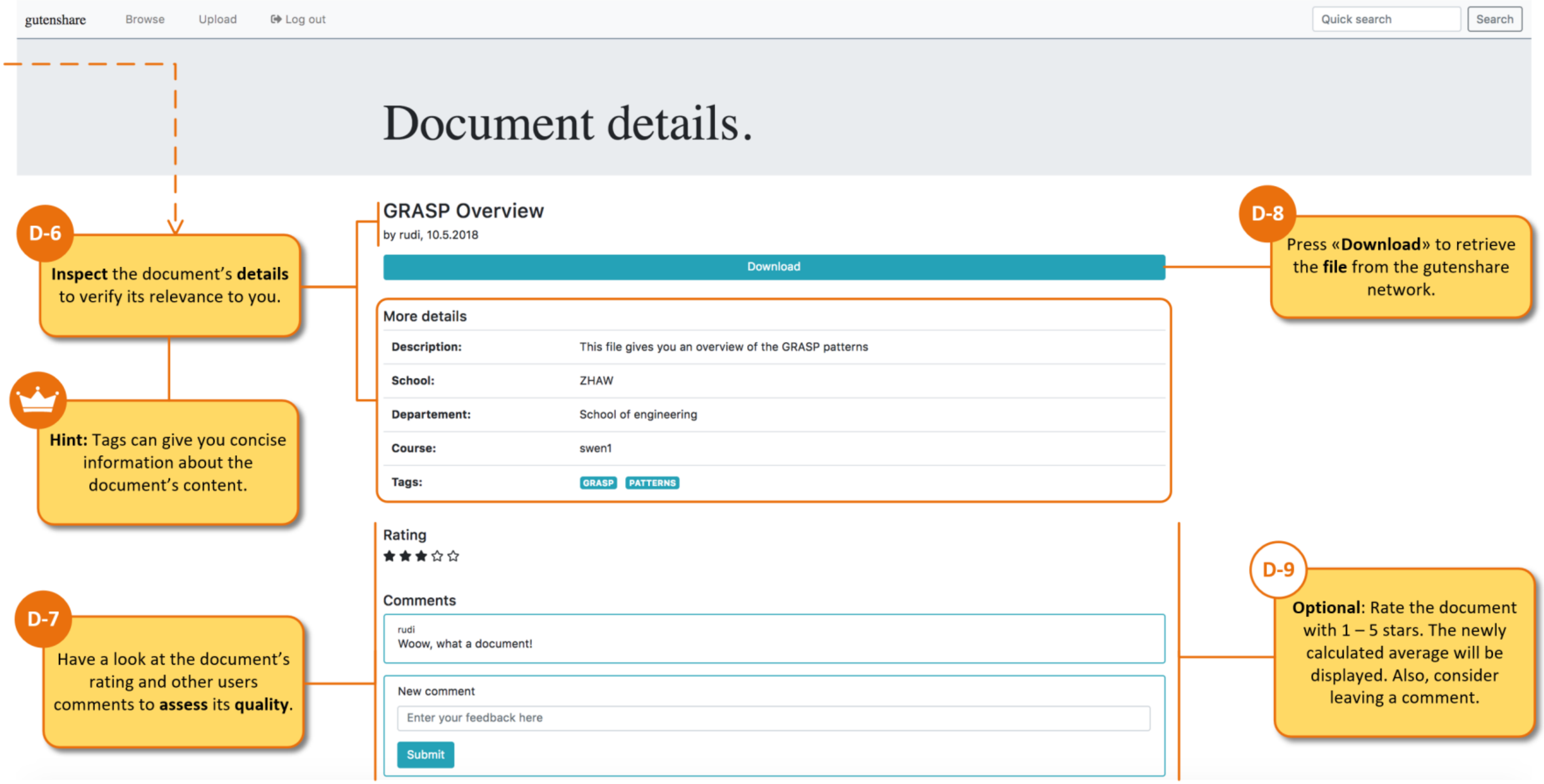
D-4 The search results table displays all **available documents.**

D-5 **Click** this button to be redirected to the document's **detail page** where you can **download** it.

Download – Detail View

Prerequisites
User has chosen to view the detail page of a document.

Goal
Download a document.



Document details.

GRASP Overview
by rudi, 10.5.2018

More details

Description:	This file gives you an overview of the GRASP patterns
School:	ZHAW
Departement:	School of engineering
Course:	swen1
Tags:	GRASP PATTERNS

Rating
★★★☆☆

Comments

rudi
Woow, what a document!

New comment
Enter your feedback here

D-6 **Inspect** the document's **details** to verify its relevance to you.

Hint: Tags can give you concise information about the document's content.

D-7 Have a look at the document's rating and other users comments to **assess** its **quality**.

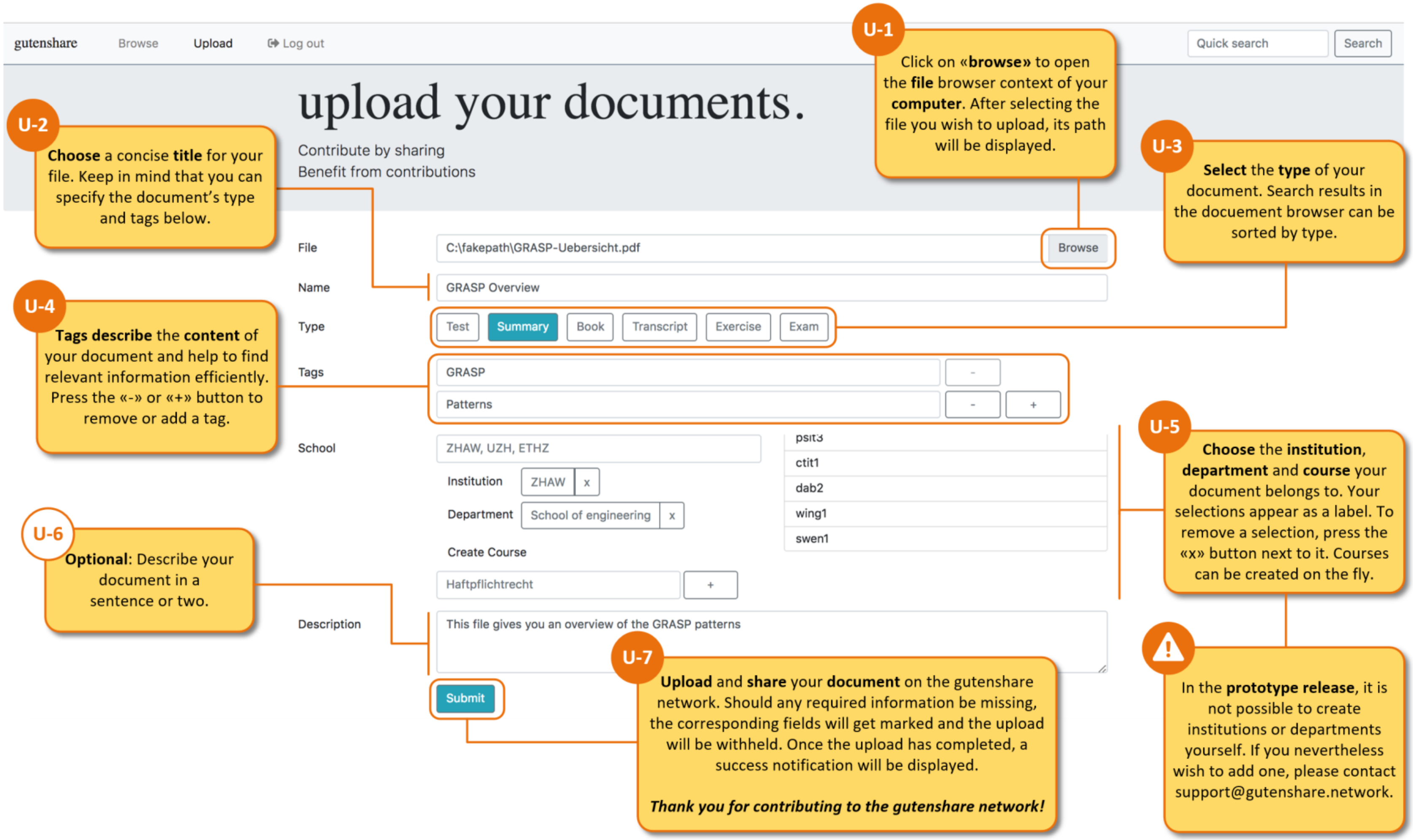
D-8 Press «**Download**» to retrieve the **file** from the gutenshare network.

D-9 **Optional**: Rate the document with 1 – 5 stars. The newly calculated average will be displayed. Also, consider leaving a comment.

Upload

Prerequisites
User has signed up and is logged in.

Goal
Share a document and provide additional information for better retrievability.



upload your documents.

Contribute by sharing
Benefit from contributions

File: C:\fakepath\GRASP-Uebersicht.pdf

Name: GRASP Overview

Type: Summary

Tags: GRASP, Patterns

School: ZHAW, UZH, ETHZ

Institution: ZHAW

Department: School of engineering

psit3
ctit1
dab2
wing1
swen1

Create Course: Haftpflichtrecht

Description: This file gives you an overview of the GRASP patterns

U-1 Click on «**browse»** to open the **file** browser context of your **computer**. After selecting the file you wish to upload, its path will be displayed.

U-2 **Choose** a concise **title** for your file. Keep in mind that you can specify the document's type and tags below.

U-3 **Select** the **type** of your document. Search results in the docuement browser can be sorted by type.

U-4 **Tags describe** the **content** of your document and help to find relevant information efficiently. Press the «-» or «+» button to remove or add a tag.

U-5 **Choose** the **institution**, **department** and **course** your document belongs to. Your selections appear as a label. To remove a selection, press the «x» button next to it. Courses can be created on the fly.

U-6 **Optional**: Describe your document in a sentence or two.

U-7 **Upload** and **share** your **document** on the gutenshare network. Should any required information be missing, the corresponding fields will get marked and the upload will be withheld. Once the upload has completed, a success notification will be displayed.

Thank you for contributing to the gutenshare network!

⚠ In the **prototype release**, it is not possible to create institutions or departments yourself. If you nevertheless wish to add one, please contact support@gutenshare.network.

6.4 Project Management

The milestone of the implementation phase has been achieved and with it the sixth iteration was successfully closed. The last iteration will transition the prototype into a productive project, to aim go-to-market release.

Over the past three months, the Group4 GmbH team has formed as a strong core, to further develop and maintain the gutenshare network. All members have made valuable learnings in their field of application and project delivery. Problems and risks were taken as a team all together and solutions were implemented straight forward.

The following section will give a summary of risks to come, regarding the product as well as an eventual productive project team. It will show a brief retrospect of the past project iterations as well as a prospect of the last iteration of the transition phase.

6.4.1 Risks

6.4.1.1 Product Risks

The following table (Table 3) gives an overview of the anticipated risks concerning the product itself.

ID	*Name*	*Description*	*Probability*	*Damage*	*Priority*	*Measures*
1	Critical mass of users	If the platform is not being used by a critical number of users not enough content is uploaded. New users will see little benefit in signing up.	60%	Extremely high	1	Aggressive marketing, school collaboration
2	Competitors catching up	There are already numerous established learning platforms with the resources available to mimic the gutenshare network concept.	60%	High	2	Extend services
3	Platform abuse	The platform mustn't be abused as a document sharing platform for non-study related content.	40%	High	3	Add a service to check uploaded documents
4	Illegal content	Documents must not infringe on copyrights or be of other illegal nature.	n/a	Medium	4	Legal assistance, service to assess uploaded documents

Table 3: Product risks

In fourth iteration a risk reduction of the "critical mass of users" risk has been achieved by getting in contact with two schools the HTW and BFH, which are interested in a collaboration. The relationship with these and potential other schools should be intensified to ensure further risk reductions. The risk of critical user mass nevertheless remains as the most important product risk.

During the construction phase, no further efforts were put into the reduction of product related risks, hence they haven't changed since. In the last iteration of the transition phase, more attention will be drawn towards product related risks again.

6.4.1.2 Project Risks

Table 4 illustrates the risks concerning the project development, after the implementation phase.

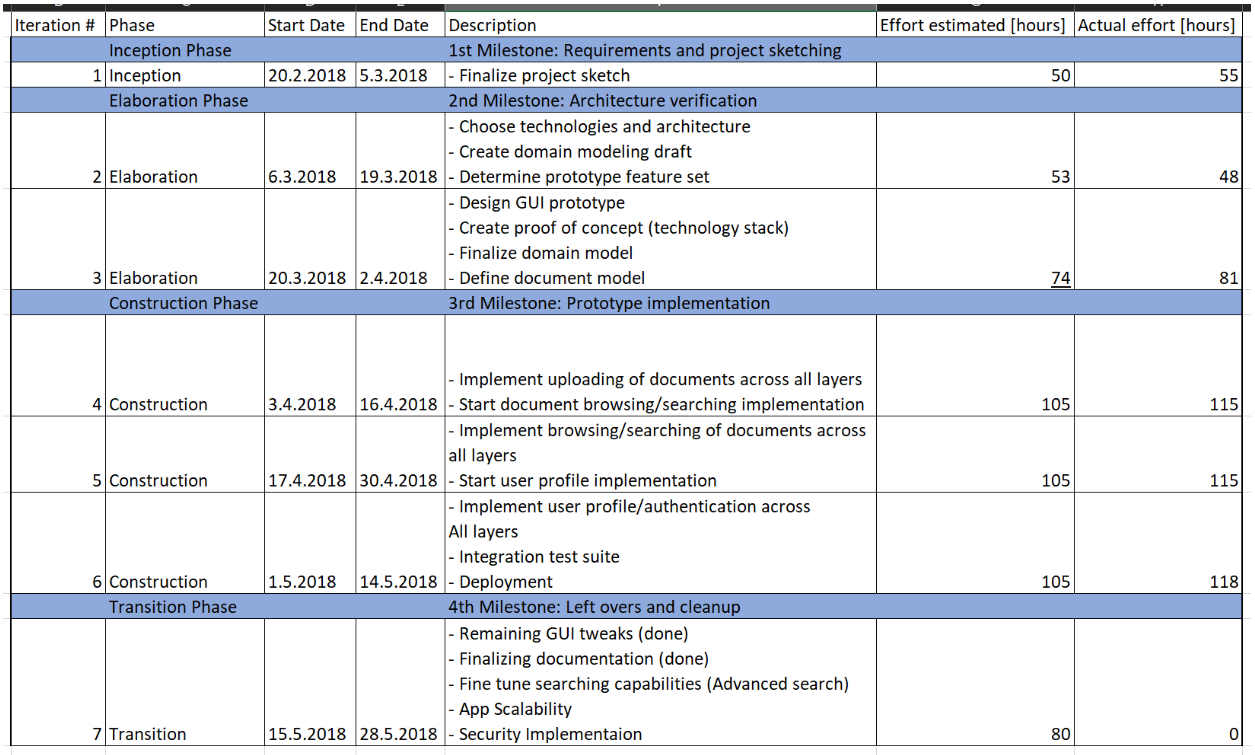
ID	Name	Description	Probability	Damage	Priority	Measures
P1	Insufficient resources	The team fails to build the product within the given time frame or runs out of funding.	0%	Extremely high	-	-
P2	Poor team dynamics	The individuals fail to perform as a team.	0%	High	-	
P3	Inadequate design	Architectural weakness leads to costly redesigns.	40%	High	1	Early user engagement, modular design

Table 4 Project risks-

In comparison to the past iterations the risks of insufficient resources (ID: P1) and poor team dynamics (ID: P2) were fully eliminated.

As for the risk of an inadequate design (ID: P3) we discovered in the construction phase an open issue in the spring boot framework which causes problems with front- to backend secure communication. A workaround is currently in evaluation. The risk has therefore been marked as top priority and the probability has been risen to 40%.

6.4.2 Phase and Iteration Plan



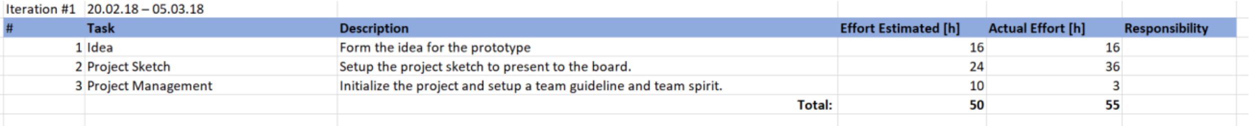
Iteration #	Phase	Start Date	End Date	Description	Effort estimated [hours]	Actual effort [hours]
	Inception Phase			1st Milestone: Requirements and project sketching		
1	Inception	20.2.2018	5.3.2018	- Finalize project sketch	50	55
	Elaboration Phase			2nd Milestone: Architecture verification		
2	Elaboration	6.3.2018	19.3.2018	- Choose technologies and architecture - Create domain modeling draft - Determine prototype feature set	53	48
3	Elaboration	20.3.2018	2.4.2018	- Design GUI prototype - Create proof of concept (technology stack) - Finalize domain model - Define document model	74	81
	Construction Phase			3rd Milestone: Prototype implementation		
4	Construction	3.4.2018	16.4.2018	- Implement uploading of documents across all layers - Start document browsing/searching implementation	105	115
5	Construction	17.4.2018	30.4.2018	- Implement browsing/searching of documents across all layers - Start user profile implementation	105	115
6	Construction	1.5.2018	14.5.2018	- Implement user profile/authentication across All layers - Integration test suite - Deployment	105	118
	Transition Phase			4th Milestone: Left overs and cleanup		
7	Transition	15.5.2018	28.5.2018	- Remaining GUI tweaks (done) - Finalizing documentation (done) - Fine tune searching capabilities (Advanced search) - App Scalability - Security Implementaion	80	0

Table 5: phase and iteration plan

The most up to date phase and iteration plan shows that in each iteration of the construction phase around 10% more than the estimated resources were needed to reach the iterations goal.

The last iteration on the other hand will have less tasks, because the GUI tweaks will be withheld until the finalized branding of the productive gutenshare network will be settled. The documentation also can be taken of the list, as they are all finalized.

6.4.2.1 Iteration #1

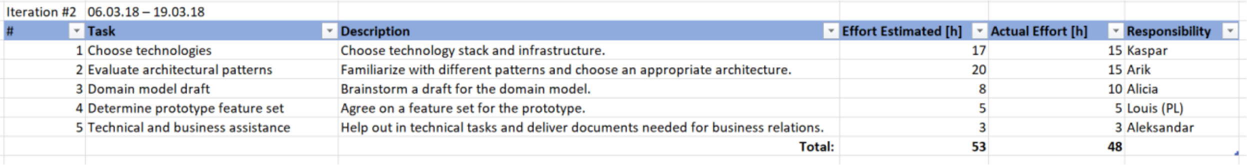


Iteration #1	20.02.18 – 05.03.18				
#	**Task**	**Description**	**Effort Estimated [h]**	**Actual Effort [h]**	**Responsibility**
1	Idea	Form the idea for the prototype	16	16	
2	Project Sketch	Setup the project sketch to present to the board.	24	36	
3	Project Management	Initialize the project and setup a team guideline and team spirit.	10	3	
		Total:	**50**	**55**	

Table 6: Iteration #1

In the first iteration of this project the inception phase was meant to sketch out the project and spot the needed requirements. As a result, the project sketch document was presented to the board and the project budget to develop a prototype was granted.

6.4.2.2 Iteration #2

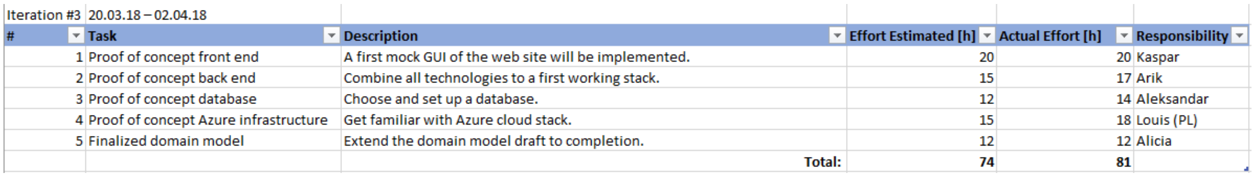


Iteration #2	06.03.18 – 19.03.18				
#	Task	Description	Effort Estimated [h]	Actual Effort [h]	Responsibility
1	Choose technologies	Choose technology stack and infrastructure.	17	15	Kaspar
2	Evaluate architectural patterns	Familiarize with different patterns and choose an appropriate architecture.	20	15	Arik
3	Domain model draft	Brainstorm a draft for the domain model.	8	10	Alicia
4	Determine prototype feature set	Agree on a feature set for the prototype.	5	5	Louis (PL)
5	Technical and business assistance	Help out in technical tasks and deliver documents needed for business relations.	3	3	Aleksandar
		Total:	**53**	**48**	

Table 7: Iteration #2

In the second iteration, first decisions were made regarding which technologies to evaluate. The architectural pattern was also evaluated with the proposal to incorporate an onion layer architectural pattern. The domain model draft gave the team an understanding of what componence and actors are interacting with the system. As much as everyone in the project team would have liked to see all ideas and features from the first iteration in a working prototype, decisions were made to a subset of features to implement in a first prototype. As a result the idea of the marketplace was postponed for a productive system.

6.4.2.3 Iteration #3



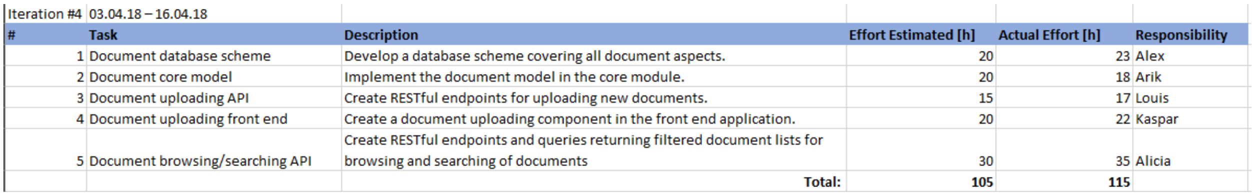
Iteration #3	20.03.18 – 02.04.18				
#	Task	Description	Effort Estimated [h]	Actual Effort [h]	Responsibility
1	Proof of concept front end	A first mock GUI of the web site will be implemented.	20	20	Kaspar
2	Proof of concept back end	Combine all technologies to a first working stack.	15	17	Arik
3	Proof of concept database	Choose and set up a database.	12	14	Aleksandar
4	Proof of concept Azure infrastructure	Get familiar with Azure cloud stack.	15	18	Louis (PL)
5	Finalized domain model	Extend the domain model draft to completion.	12	12	Alicia
		Total:	**74**	**81**	

Table 8: Iteration #3

With the third iteration the elaboration phase has been successfully completed. The architectural decisions have been verified.

As a result, the team has decided that universities and schools must be initialized by the system and therefore in a first step by a system administrator. This ensures data integrity by preventing duplicates of schools and departments. Otherwise documents will not be correctly categorized and won't be found by the users later on. This finding has a low impact on the development plan of the prototype.

6.4.2.4 Iteration #4

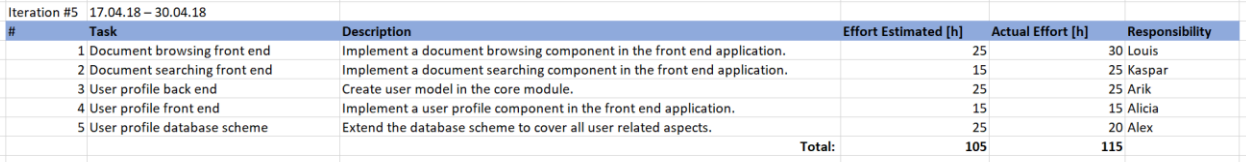


Iteration #4	03.04.18 – 16.04.18				
#	Task	Description	Effort Estimated [h]	Actual Effort [h]	Responsibility
1	Document database scheme	Develop a database scheme covering all document aspects.	20	23	Alex
2	Document core model	Implement the document model in the core module.	20	18	Arik
3	Document uploading API	Create RESTful endpoints for uploading new documents.	15	17	Louis
4	Document uploading front end	Create a document uploading component in the front end application.	20	22	Kaspar
5	Document browsing/searching API	Create RESTful endpoints and queries returning filtered document lists for browsing and searching of documents	30	35	Alicia
		Total:	**105**	**115**	

Table 9: Iteration #4

The forth iteration overall is the first iteration of the construction phase and has been completed successfully. We developed a database scheme, implemented the document model in the core module, created RESTful endpoints for our first use case (the upload of a document), integrated the first uploading component in the front-end application and defined RESTful endpoints for getting a list of documents to complete the task. As shown in the iteration plan we required more time than initially estimated. This had various reasons. For example, we adjusted the database schema several times during the implementation to meet the latest requirements. Furthermore, creating the RESTful endpoints took more time than expected because we had to get familiar with the RESTful guidelines and best practices.

6.4.2.5 Iteration #5



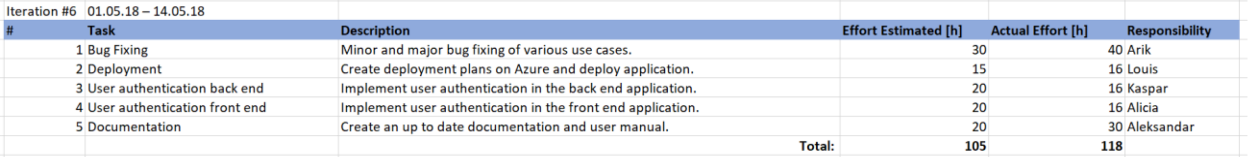
Iteration #5	17.04.18 – 30.04.18				
#	Task	Description	Effort Estimated [h]	Actual Effort [h]	Responsibility
1	Document browsing front end	Implement a document browsing component in the front end application.	25	30	Louis
2	Document searching front end	Implement a document searching component in the front end application.	15	25	Kaspar
3	User profile back end	Create user model in the core module.	25	25	Arik
4	User profile front end	Implement a user profile component in the front end application.	15	15	Alicia
5	User profile database scheme	Extend the database scheme to cover all user related aspects.	25	20	Alex
		Total:	**105**	**115**	

Table 10: Iteration #5

In the fifth iteration the document browsing and uploading of a document was implemented and the development team was working hard to achieve all set goals. An unexpected business event of a member has created a shortage on top. Remote conferencing sessions and additional effort of members could balance these difficulties and made it possible to deliver on time. Nevertheless, some open tasks remained regarding upload and download.

Additional time was also spent to look into the deployment in advance, as there was new information about the Azure WebApps and not clear if they would sufficient the gutenshare network requirements.

6.4.2.6 Iteration #6



Iteration #6	01.05.18 – 14.05.18				
#	Task	Description	Effort Estimated [h]	Actual Effort [h]	Responsibility
1	Bug Fixing	Minor and major bug fixing of various use cases.	30	40	Arik
2	Deployment	Create deployment plans on Azure and deploy application.	15	16	Louis
3	User authentication back end	Implement user authentication in the back end application.	20	16	Kaspar
4	User authentication front end	Implement user authentication in the front end application.	20	16	Alicia
5	Documentation	Create an up to date documentation and user manual.	20	30	Aleksandar
		Total:	**105**	**118**	

Table 11: Iteration #6

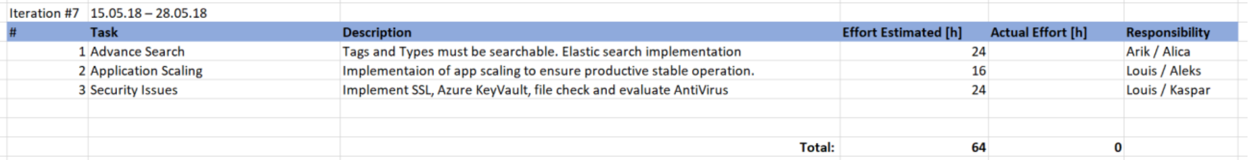
The sixth iteration was affected by ups and downs but still a success in reaching the milestone of a working prototype.

The Azure deployment had to be restructured to achieve a fast and reliable deployment path and stable service. While implementing the authentication to ensure, only registered users can use the gutenshare network, an open issue of the Spring Boot framework has thrown the team back in the planning.

Besides the documentation of the project summary and user manual has consumed a lot more time, as the team has decided to custom draw the user manual in order to achieve a coherent and meaningful documentation.

At the end of the sixth iteration, the prototype is now in a stable release, deployed in the cloud and ready to fulfill its destiny and convince the users and investors to advance in a productive project state, with Group 4 GmbH.

6.4.2.7 Iteration #7



Iteration #7	15.05.18 – 28.05.18				
#	Task	Description	Effort Estimated [h]	Actual Effort [h]	Responsibility
1	Advance Search	Tags and Types must be searchable. Elastic search implementation	24		Arik / Alica
2	Application Scaling	Implementaion of app scaling to ensure productive stable operation.	16		Louis / Aleks
3	Security Issues	Implement SSL, Azure KeyVault, file check and evaluate AntiVirus	24		Louis / Kaspar
		Total:	64	0	

Table 12: Iteration #7

The last iteration has just started and the prototype can be refined to provide a smooth transition into a productive product. The under 5.2 listed open task can be closed by implementing the missing security concepts, an elastic search and improving the search capabilities overall. Additionally, the application will be inspected regarding performance and the infrastructure adapted accordingly.

6.5 Acknowledgments

The Group4 gutenshare network project team would like to thank its advisers for their support in helping to create the gutenshare network prototype. Also, we would like to thank the ZHAW, Supercomputing Systems AG and Swisscom AG for their generosity to use their facilities.

Group4 GmbH

ZHAW FS 2018

Alicia Rüegg

Kaspar Wolfisberg

Louis Leon Müller

Aleksandar Spasojevic

Arik Sidney Guggenheim