HOCHSCHULE HEIDELBERG
Intelligence in Learning

Student grading System

by

Aalisha Dogra, Jowei Lee, Martin Vogel, Navata Nadkarni

SRH University Heidelberg
Faculty of Information, Media and Design
Applied Computer Science
Software Architecture and Development

Submission Date: 11.05.2017

Reviewers: Christoph Hahn M.Sc.

Table of Content

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Explanation
COTES	Competence Oriented Transparent Evaluation System
ERD	Entity relationship diagram
UI	User Interface
UX	User Experience
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
IDE	Integrated Development Environment
ASP	Active Server Pages
ORM	Object-Relationship Mapping
MVC	Model-View-Controller
DRY	Don´t Repeat Yourself

1 Introduction

1.1 Purpose of this document

This document was written in the course “Software Architecture and Development” at SRH Hochschule Heidelberg within the Master of Science Applied Computer Science Program. It describes the project to develop a prototype for a software product called “Student Grading System”. The project will follow the principles of design thinking and will adjust those where it is needed and appropriate. Any methodology used and the results as well iterations to get to this result will be described within this document. Together with a code repository that is hosted on github (1.3) it serves as the final project submission for the Software Architecture and Development course.

1.2 Project Goals

This project aims for a software implementation of the grading methodology defined in the “COTES – Competence Oriented Transparent Evaluation System” research project. COTES is a research project done by Guido Kühn, Michael Hebel and Christoph Hahn. The research project goes on while this first implementation is getting done. Therefore, some specifics of the implementation might change in a later step.

The team has the following vision:

“Provide high quality software support in order to enable professors to give fair, transparent and comparable grades”

1.3 Team Structure

The team consists of 4 students in the above mentioned masters program:

Last name	First name	Matriculation Number	E-Mail address	Github Username
Dogra	Aalisha	11008335	Aalisha.Dogra@stud.hoschule-heidelberg.de	AalishaDogra
Vogel	Martin	11009471	Martin.Vogel@stud.hochschule-heidelberg.de	MartinVogelSRH
Nadkarni	Navata	11008330	Navata.Nadkarni@stud.hochschule-heidelberg.de	navata-nadkarni
Lee	Jo-Wei	11008322	Jo-Wei.Lee@stud.hochschule-heidelberg.de	weiwei1128

Table 1: Team Structure

The git Repository for this project will be hosted on Github. It can be found under the following link: https://github.com/MartinVogelSRH/StudentGradingSystem.git

In this repository, there is a folder for the clickable version of our prototype. It can be found here:

https://github.com/MartinVogelSRH/StudentGradingSystem/tree/master/Clickable_Prototype

Tasks within the project have been split into tasks regarding this document (referred as “documentation”) and tasks regarding working on requirements / the prototypes. Code contributions can be checked using the commit history within the above mentioned git repository.

Task	Main responsible
Documentation – Chapter 1	All
Documentation – Chapter 2	Aalisha Dogra
Documentation – Chapter 3	Navata Nadkarni
Documentation – Chapter 4	Martin Vogel
Documentation – Chapter 5	Jowei Lee
Documentation – Chapter 6	Martin Vogel
Documentation – combine individual work	Martin Vogel
Personas – creation	All
User stories – creation	All
Paper Prototype – Draft	All
Low-fi prototype – Draft	Navata Nadkarni
Mid-fi prototype – Draft	Navata Nadkarni
Mid-fi prototype – finalizing	All
User Interviews	Martin Vogel
ERD – Draft	Aalisha Dogra & Martin Vogel
Input for technology decision	All
Deciding on technology	Navata Nadkarni & Martin Vogel
Setup of Development environment	Martin Vogel

Table 2: Tasks and responsibilities

2 Requirements

2.1 Design Thinking

In order to develop new applications, a well-known methodology that can be used is the approach of design thinking, which will be used in this project as well. “Design thinking is a method used by designers to solve complex problems and find desired solutions by the clients. […] Design thinking is based upon the logics, imagination, systemic reasoning to inquire the possibilities of what could be and to create a solution that is beneficial for the end user.”[1] Design thinking is also an approach that is used to consider issues, with a mean to help settle issues, more widely than within a professional design practice.

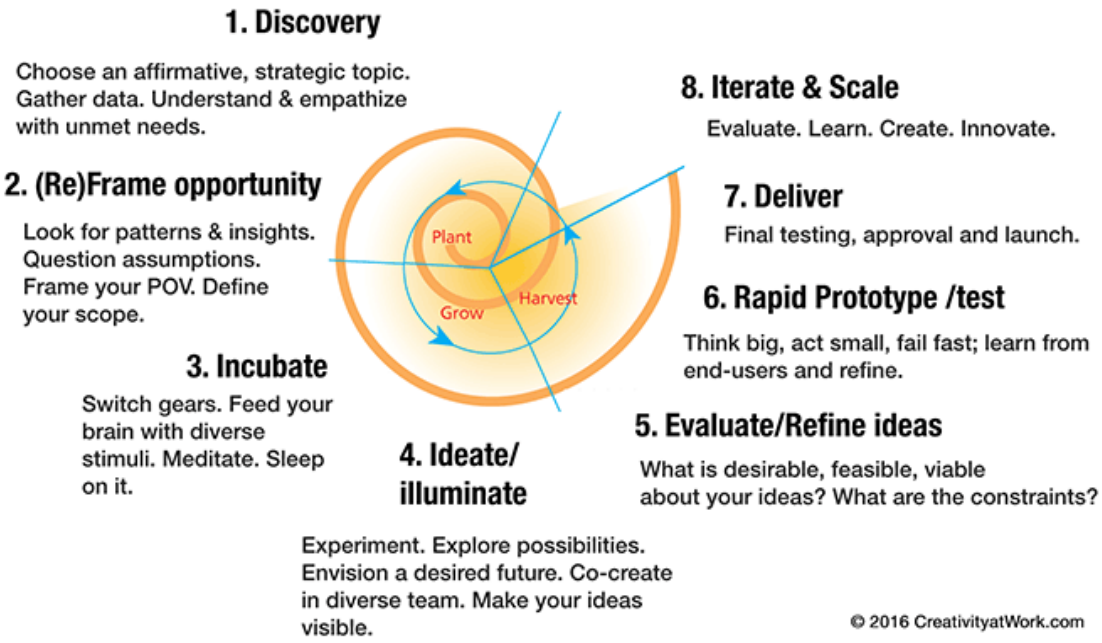


Figure 1: A Framework of design thinking[2]

[1] [CREATWORK1]

[2] [CREATWORK1]

In design thinking, the following phases are being used:

Discovery – In this phase a project team can choose a topic , collect data and requirements needed for the project.

Reframe opportunity – The team can than decide the patterns, assumed questions and define the scope.

Incubate – In this phase an individual needs to clear the mind with all the information he/she has gathered .

Ideate/Illuminate – experiment with the data you have collected , explore all the possibilities , create different ideas.

Evaluate/Refine ideas – Examine your ideas

Rapid prototype/test – In this phase you test the gathered information.

Deliver – In this phase the collected information is tested ,approved and launched.

Iterate and scale – Examine , Learn ,Create , Innovate.

2.2 Benefits of design thinking

Within this project, there are a number of benefits seen by the use of design thinking:

- End-User Focus

  As described before, a key consideration of this project is the focus on end-users and providing an easy to use User Interface. Design thinking supports this by putting customers at the core of the process. It aims to produce beneficial product and solutions according to the requirement of the customer. The process keeps in mind

all the human values and experiences in order to make their living meaningful and more fulfilling.[3]

- Leverages collective expertise

  As described before one of the main focus of this project is to build collaborative teams and bring out various opinions, we come out of our respective fields to hold our cumulative foresight, experience and ability. We need to assemble our skills and philosophies to bring out the best in every way.[4]

- Design thinking leads to innovation

  As the project team main focus lies on the creation of new, different, and innovative ideas that are novel to a situation. Same logos or website cannot be used by the same or two different businesses can never have the same principles, plans and regulations.[5]

2.3 Personas

Personas are one of the important key factor in our project. Personas are user models that characterize persons of a target group in their characteristics. Personas have goals and behaviors, preferences and expectations.

Within the design thinking approach, personas can be used in the phases given below:

- Analysis:

  At the moment our project team is mainly focusing on the analysis phase. At the very beginning of a project it is important to focus on getting a better understanding

[3] [DREWL1]

[4] [DREWL1]

[5] [CMSWRE1]

of the end user. Therefore, personas are a helpful tool in the beginning of the requirement analysis phase to identify and describe potential end-users with their characteristics.[6]

- Operation:

  If the application is already published, this persona approach is suitable. It helps in comparison with the actual needs of the end users. It helps in ensuring that the user requirements are successfully implemented and if not, get an idea of where improvements can be made. Other user groups can be addressed to extend the scope of the application.[7]

While discussing, the team came up with some personas that are Persona-Student, Persona-Professor and Persona-Program-Manager. Mainly the personas are dealing with a person's goal, characteristics and emotions. According to the project, the respective team decided to keep the personas rather simple because this project is based on a the COTES research project that already describes specific conditions that cannot be changed. The COTES project for example already describes the process flow of grading a student[8]. All the personas in our case have main responsibilities , goals and the task , that are listed below:

Personas – Student

"I would like to get my results sooner and to know where my strengths and weaknesses lie so that I can improve myself"

Main responsibility	Comply with course goals
Main Goals & Tasks	Understand the Background of the grades Ensure High Quality by providing feedback

Table 3: Persona-Student

[6] [ONLNMRKTPRA]

[7] [ONLNMRKTPRA]

[8] Annexure F

Personas – Professor

"Transparency and efficiency of giving grades"[9]

Main responsibility	Teach & Grade Users Course Development
Main Goals & Tasks	Give fair grades Give fast grades Improve Quality

Table 4: Persona-Professor

Personas – Program Manager

Main responsibility	Ensure high Quality of study Programs
Main Goals & Tasks	Improve Quality

Table 5: Personas-Program-Manager

2.4 User Stories

As discussed above like Personas, User stories are another key factor in our project. They are used to create time estimates for the release-planning meeting. User Stories are written by the customer as things that the system needs to do for them.

User Stories are an agile approach that focuses on discussing the requirements with the users and within the team. All user stories are described in a sentence or two aiming at the required functionality.[10]

[9] Annexure F

Since user stories are very similar to use cases, it is important to differentiate between the two. User stories can be used in project planning. They help in reducing the number of hours and how challenging it would be to develop the application, while use cases are more generic to provide estimation on the needed development time. In agile development, a single iteration can generally comprise of just a part of a use case but by convention it focuses on a single user story.[11]

Within this project, user stories for student, professor and project manager have been created keeping in mind what they individually want and why they want that particular thing. So the user stories in our case have been mentioned below:

As a	Student
I Want to	Give Feedback
In Order to	Help ensure high Quality

Table 6: User Story 1: Student Feedback

As a	Student
I Want to	Understand details of my grades
In Order to	Know areas I can improve in

Table 7: User Story 2: Student Understanding

[10] [MNTNGTS1]

[11] [MNTNGTS2]

As a	Professor
I Want to	Fill out a standardized questionnaire
In Order to	Give comparable and fast grades

Table 8: User Story 3: Professor Filling

As a	Professor
I Want to	Run statistics
In Order to	See the overall class Performance

Table 9: User Story 4: Professor Statistics

As a	Program Manager
I Want to	Create standardized questionnaires
In Order to	Ensure comparable grades

Table 10: User Story 5: Program Manager Standardization

As a	Program Manager
I Want to	Run statistics
In Order to	Understand and improve Quality

Table 11: User Story 6: Program Manager Statistics

2.5 User Experience Map

"Experience maps are a visual representation of the users' journey over time, and they provide a handy communication tool for teams to inform product direction. The benefit is that it tells a visual story in one-page that can be easily shared to communicate a product's current state and opportunities."[12]

In our case, using user experience maps does not entirely fit to the purpose as it deals with the emotional side of the user and our project has a fixed rules and regulation that will affect the entire study program if changed. However, it is still seen as beneficial to have an overview over the process flow of the application in order to refer back to this in a later stage. Therefore, the project team developed the following "reduced" user experience map:

[12] [BLINKUX1]

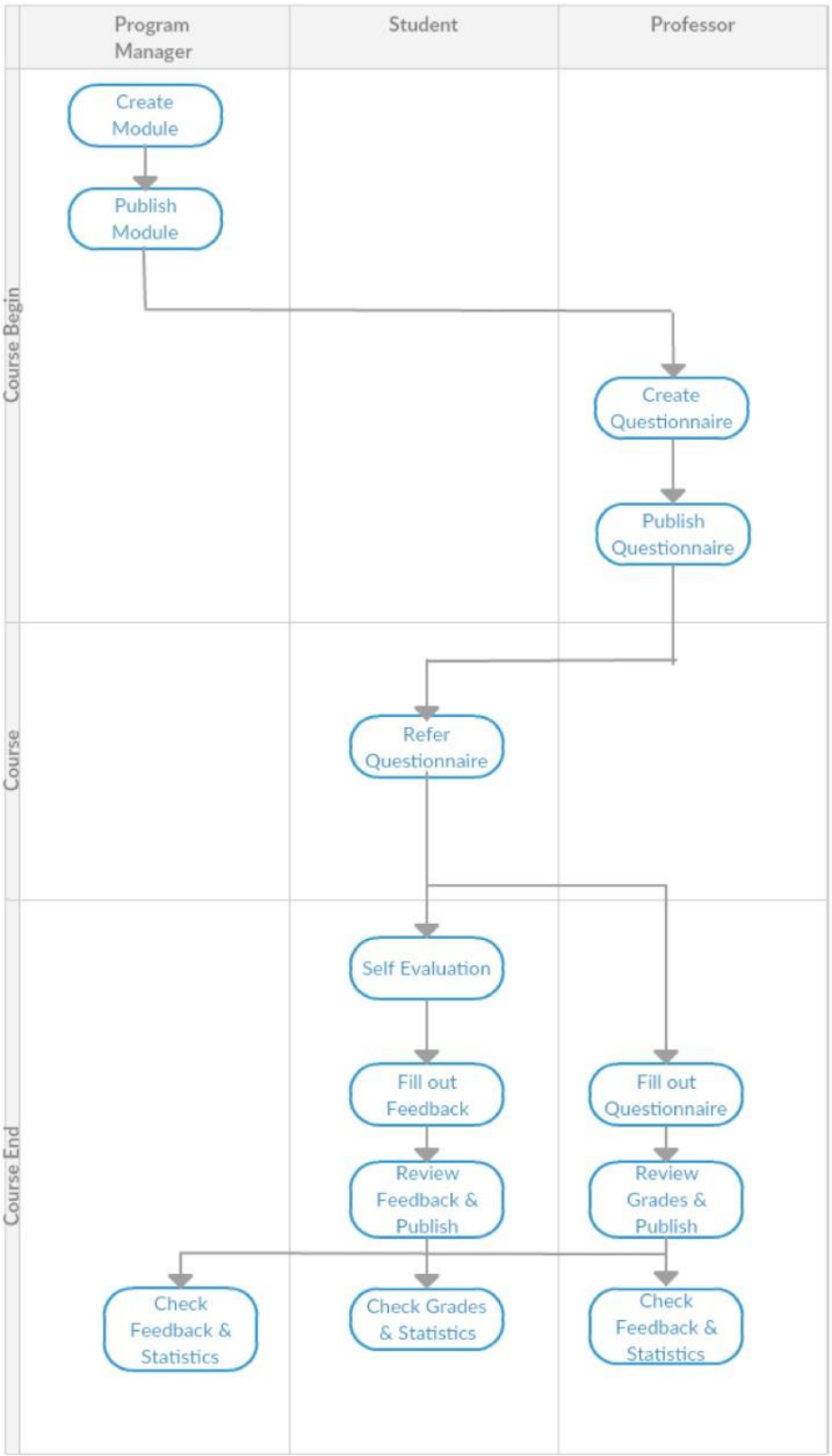


Figure 2: User experience map

2.6 Requirement List

A requirement list is used to capture all the requirements needed. Requirements can change according to the wish list of the user. Since the purpose of this project is to create a prototype which can later be used to create a minimum viable product, these requirements might be prioritized in a later step. Within this project, the following requirements have been identified:

No.	Requirement	Description
1.	A Systematic grading system	The system needs to provide the ability to grade users based on defined criteria and questions.
2.	Standardized questionnaire	The system needs to provide the ability to use standardized questionnaires for courses
3.	Transparency	The system needs to provide transparency about the given grades to all different kinds of users
4.	Cross Platform	The system must work on all the platforms, screen sizes and operating systems
5.	Analytics	The system needs to run statistics in order to view the overall class performance, show progress of the student in certain fields, compare classes to each other.
6.	Feedback	The system needs to provide a proper feedback in order to ensure the quality of study program.
7.	Exporting	The system needs to ensure the calculated results can be exported and further used in other applications
8.	Central Login	The system needs to use the central login credentials at SRH

9.	Multi-Language	The system needs to be able to support multiple languages
10.	Research Support	The system needs to allow the professor to enter a "gut feeling" grade as well as the student to perform a self-evaluation, which will both only be used for research purposes
11.	Import from Scan	The system needs to provide the ability to scan printed questionnaires and automatically import them.

Table 12: Requirements

3 Low Fidelity Prototype

3.1 Paper Prototype

As mentioned in the design thinking process, the user is an integral part for the successful implementation of a project. Prototyping is a way of keeping the users informed of our understanding of their requirements and the workflow expected from us.

There are different types of prototypes that the project team comes across along the implementation of the project. The prototype used at the very beginning, in the requirement-gathering phase, is the Low fidelity prototype. It is mainly paper based with sketches of the screens. They are created based on the previously identified personas. The main purpose is to illustrate design ideas and concepts and to be able to easily modify the prototype based on user feedback.

The initially created paper prototype was a rough draft of the workflow and the user requirements[13]. It gives the user an impression of the UI and especially the application flow but only limited details on the functionality and tasks that will be undertaken. Testing the low fidelity prototype with the user helps to get an idea of the user's needs and consider their perspective throughout the following iterations.[14]

3.2 Mid fidelity prototype

Since the paper prototype had limitations regarding the general transitioning from one screen to another and the interactions with the user, we decided to adapt the next type of prototype – The mid-fidelity prototype. This prototype is a simple clickable version[15], which would satisfy our need to test the user's interaction with the pages.

---

[13] Annexure A

[14] [SJWM1], page 115

[15] [VISHIER1]

The following sections will show how we started with defining the activity flow, which in turn helped in the building the layout and navigation. The team used “Pencil” which is a prototyping tool for the same.

3.2.1 Activity Flow

As mentioned earlier, we are focusing on the 3 personas defined in the requirements chapter, The program manager, the professor and the student. The graphical representation of the workflow is called an activity diagram.[16] The activity flow of each persona in respect to their tasks is discussed in this section along with their activity diagrams.

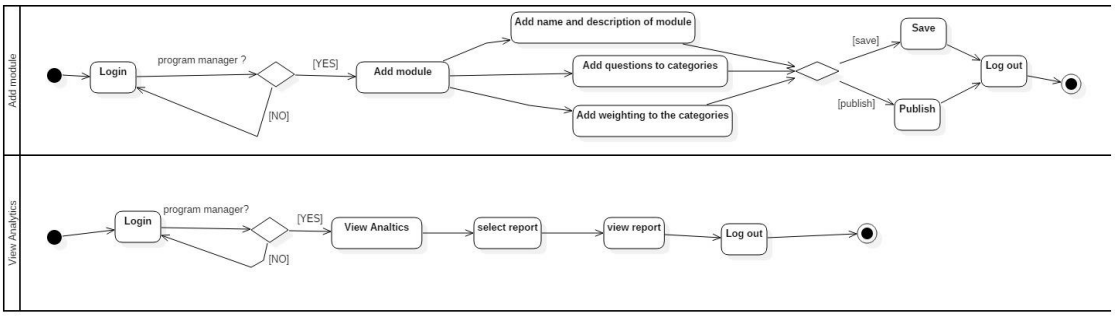


Figure 3: Activity Diagram for Program Manager

As the program Manager ,the user is responsible for :

- Creation of modules per faculty.
- Deciding on the questions for each category in the questionnaire.
- deciding how much each category weighs.

Our application will provide him with analytics, comparing different modules so as to support him in making informed decisions.

[16] [HGHMHED1]

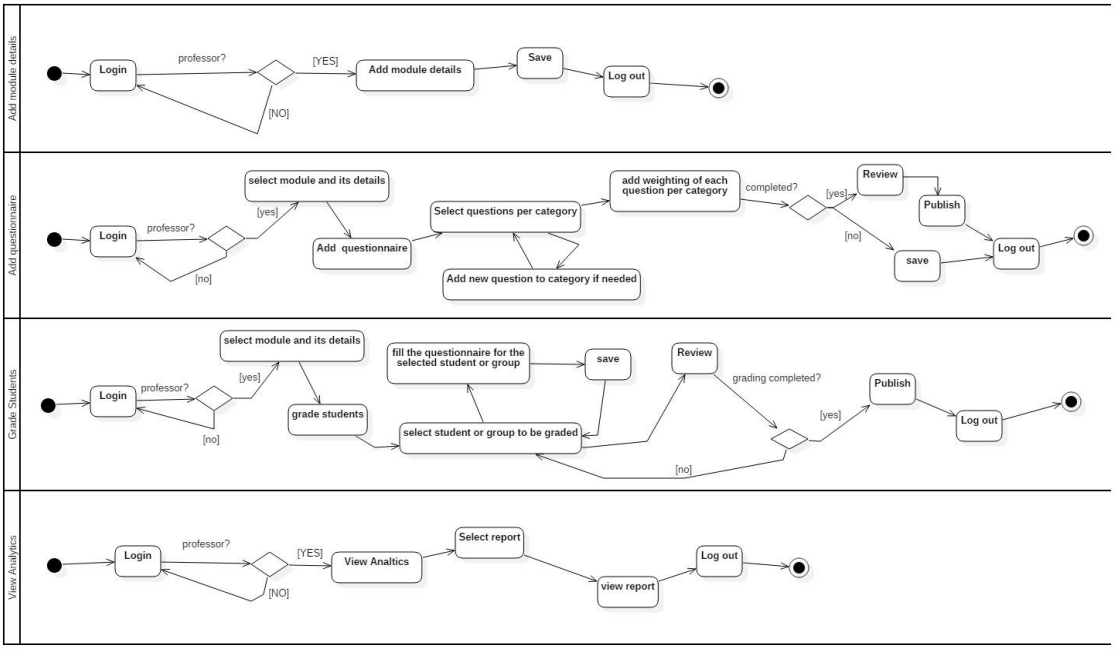


Figure 4: Activity Diagram for Professor

As the professor the user is responsible for

- Adding the module details , for example the duration of the course, the timings etc.
- Adding the questionnaire to the module by selecting the questions provided by the program manager per category.
- Grading the students as per their performance by answering the questionnaire selected at the start of the course.

The Analytics will help the professor compare the entire class grades and also compare the performance of the previous batches with the current one so that he can analyze if his methods have improved and what changes he can adapt further on.

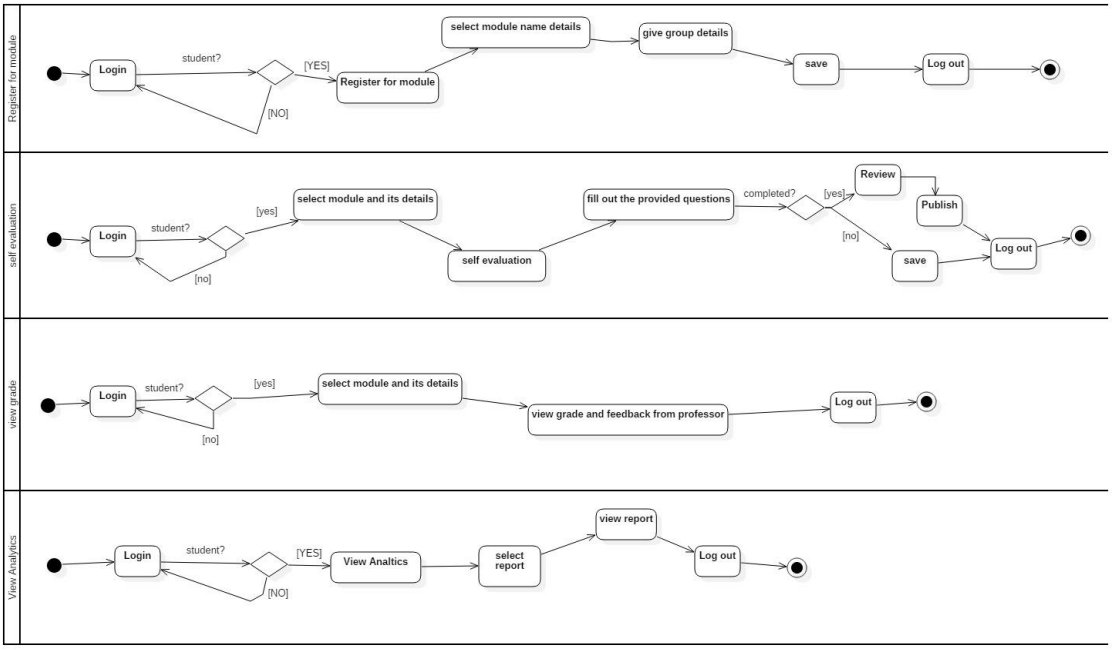


Figure 5: Activity Diagram for Student

As a student, the user is expected to

- Register for the course, give details of the group.
- Evaluate himself based on the questionnaire selected by the professor for that particular module.

The student can view the results with the feedback from the professor. Also he can compare his performance with that of different modules and can analyze where he should improve and which aspects are his strengths.

3.2.2 Layout /Navigation

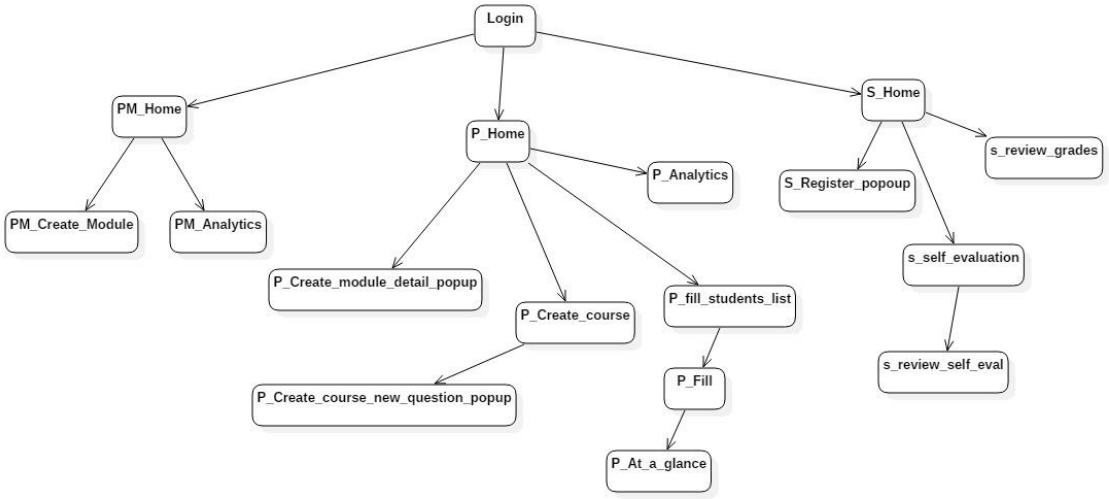


Figure 6: Screen Map

Our process of designing the user interface started with interviewing users for a clear picture of the requirements. Our first draft was a paper prototype[17]. The paper prototype was discussed during the group meetings and on further feedback from the team, a few more changes were made. After that we moved on to create a prototype using "Pencil" which is a prototyping tool. Pencil helped us create a clickable version, which we tested with the users. [18]The users offered a few more suggestions[19]. After considering the suggestions, we designed the final version on pencil[20]. The specific screens of our prototype and the iterations for each of these screens is described in the following section.

[17] Annexure A

[18] Annexure B

[19] Annexure C to D, Suggestions in Annexure F and G

[20] Annexure E

3.2.3 Screens

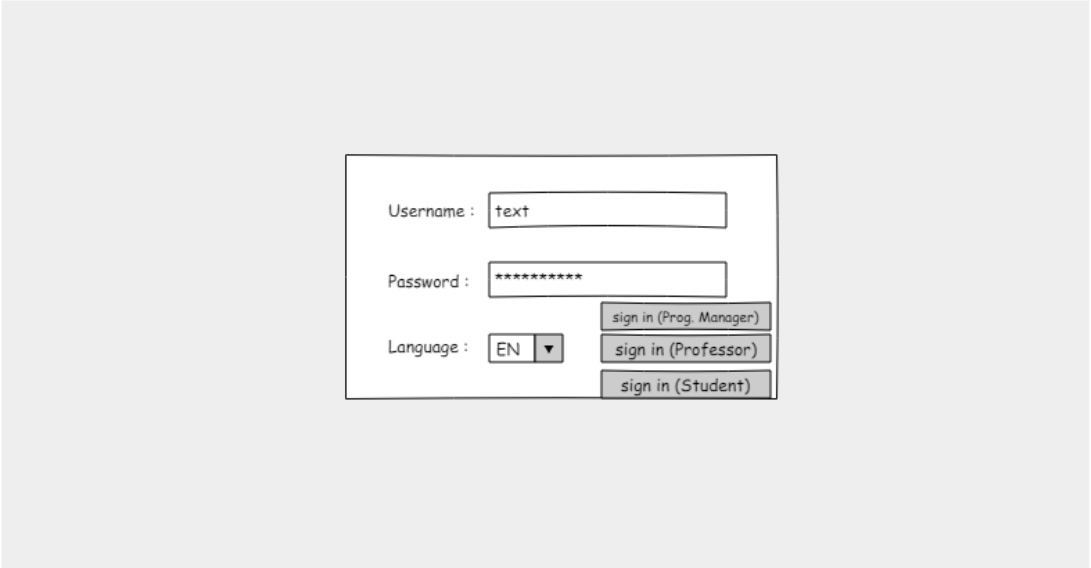


Figure 7: Screens - Login

The login page (Figure 7) shows three buttons for login for the purpose of the usability tests. In the application we intend to have a single button which provides authentication for three different roles, namely the program manager, the professor and the student.

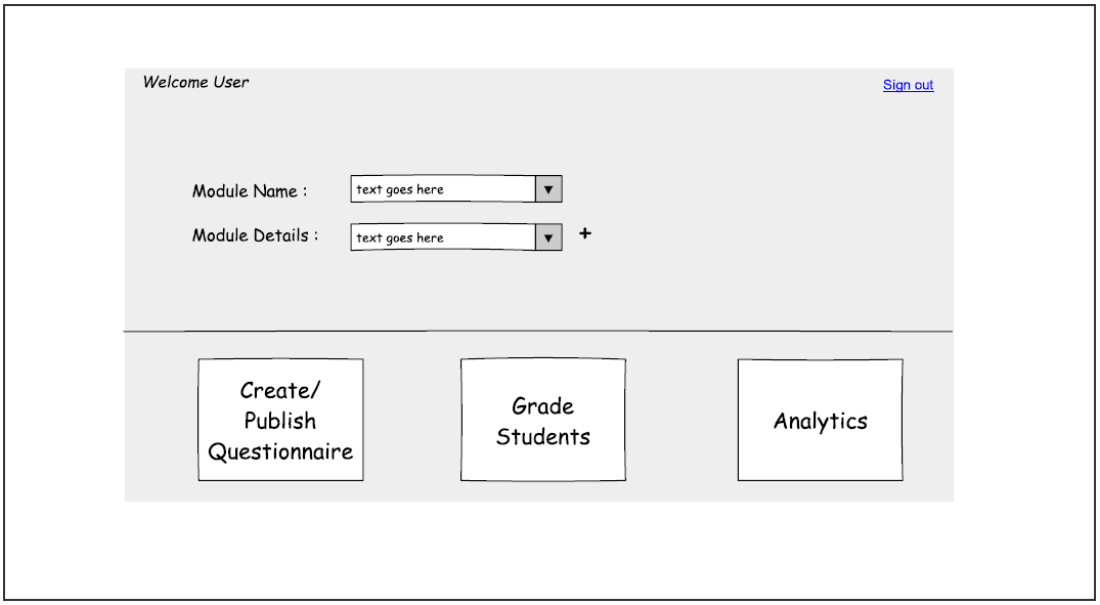


Figure 8: Screens - P_home

When the professor logs in, he is redirected to the home page designed specifically for his role (Figure 8). In our first iteration[21], i.e. the paper prototype we hadn't considered the possibility of being able to create a module or module details. He had an option of creating a questionnaire, grading students and viewing analytics. After the meetings and the discussions we realized, if the professor does not have a module in the first instance he would have to create one and thus we added the option to create module and to add the module details[22]. The module details are the duration of the module and when the module takes place.

After user review, we concluded that the professor does not get to create the module or the questionnaire. So we have given him only the option to select from the drop down as to which module he would be taking. However, he would still have to add the details of the module[23]. (Figure 9)

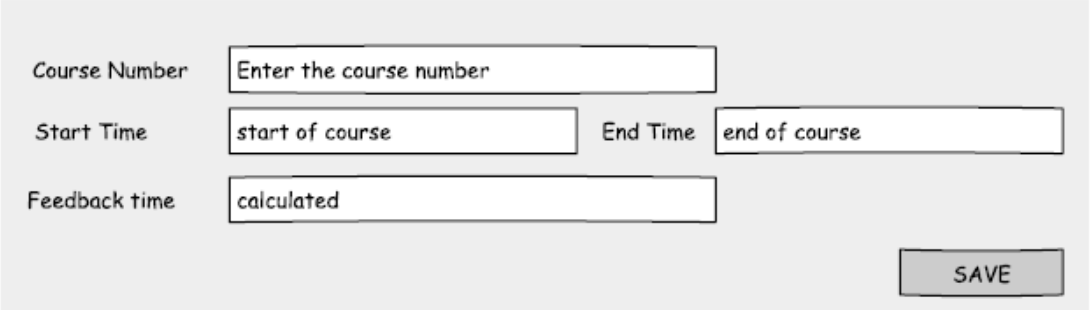


Figure 9: Screens - P_create_module_detail_popup

[21] Annexure A

[22] Annexure B – Screen P_Home

[23] Annexure E - Screen P_Home, P_Create_module_popup

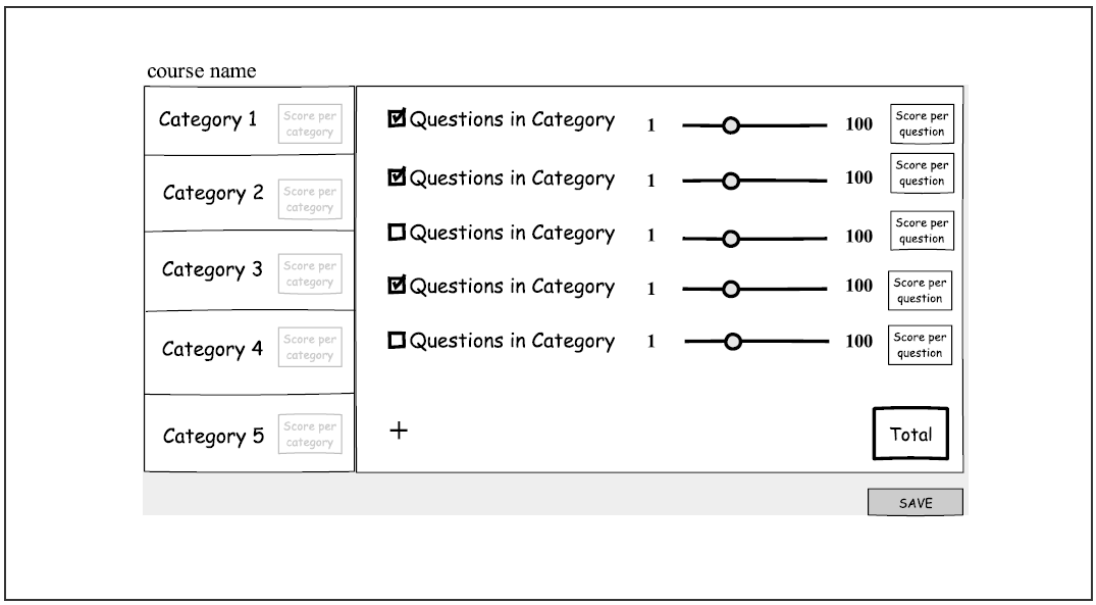


Figure 10: Screens - P_create_course

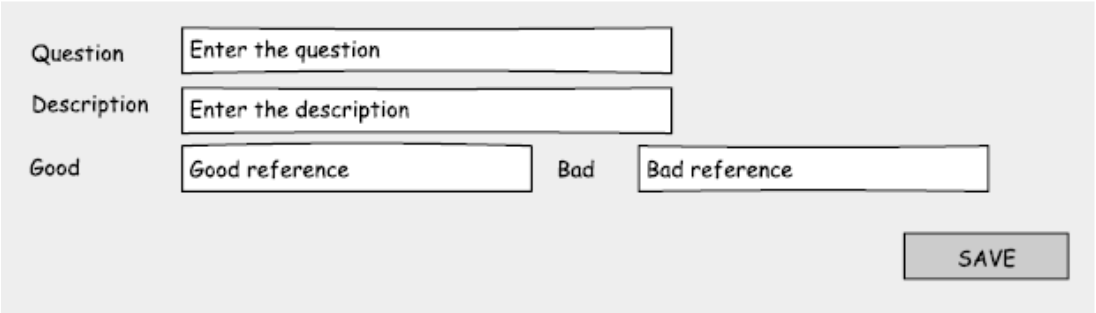


Figure 11: Screens - P_create_course_new_Question_Popup

If the professor, clicks on create questionnaire button, he is redirected to this screen (Figure 10). In the earlier stages of our design process, the professor was given the task of creating the questionnaire for the module. But after the reviewing the feedback from the user, we realized that there would have to be another role to ensure the standardization of the questionnaire amongst different modules. So, at this stage the professor can select from a list of standardized questions per category. If, However, he wants to add a new question to the pre-existing listing he can click on the add (+) symbol. This will show the popup (Figure 11) where he can add his own question in the category selected.

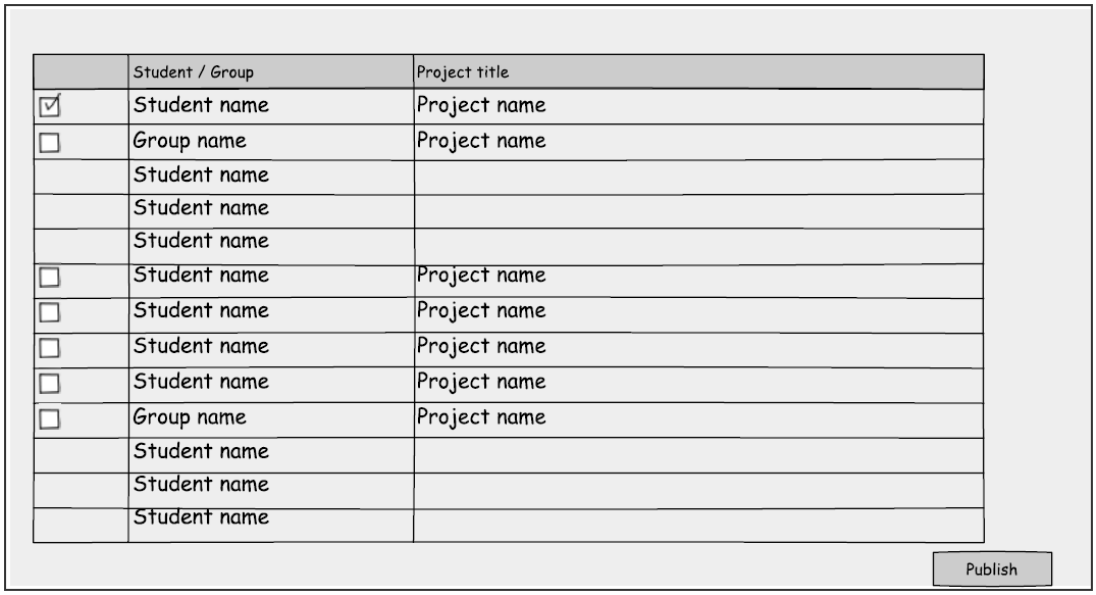


Figure 12: Screens - P_fill_student_list

When the professor clicks on the "grade students" button on his home page, he is directed to a screen (Figure 12) with the list of the students registered for the module. This page was designed after the creation of the paper prototype[24]. This page allows the professor to select the student he wishes to grade. Also if it's a group that wishes to be graded together, the list would have the group name with the students in that group. This page also serves as a review page before the professor publishes the grade.

[24] Annexure B

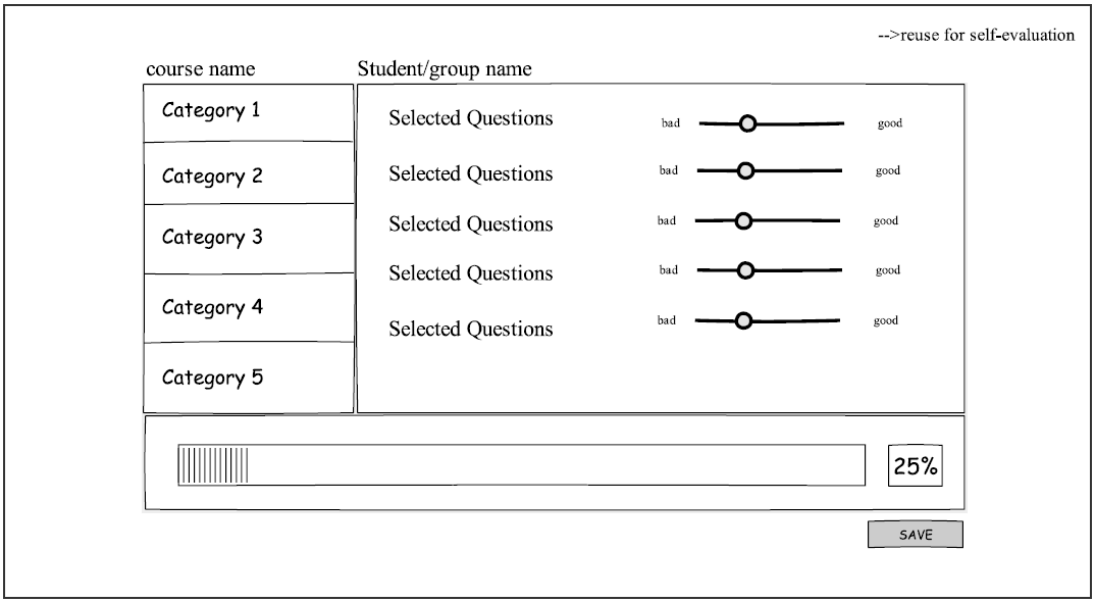


Figure 13: Screens - P_fill

When the professor clicks on one of the students or groups on the previous page, he is directed to the page where the questionnaire that he had customized for the course is available and he fills it out. (Figure 13) This page is the one, which contributes to the final grade of the student. As and when the professor finishes grading a student he is prompted to review his work (Figure 14). Therefore, he can publish it to the list of students, which will be published finally after the entire class is graded.

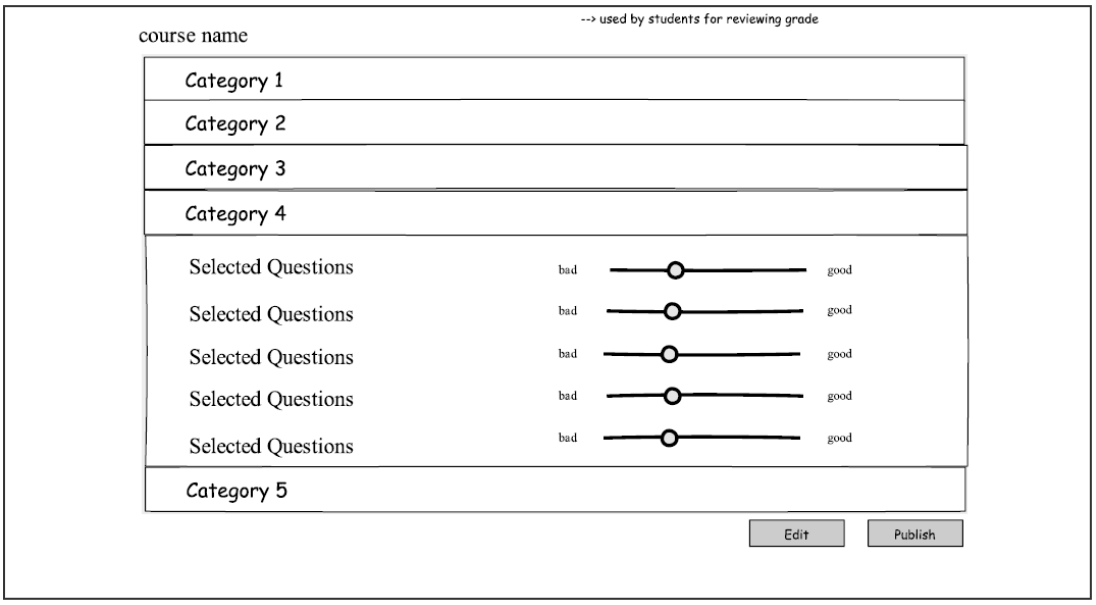


Figure 14: Screens - P_at_a_glance

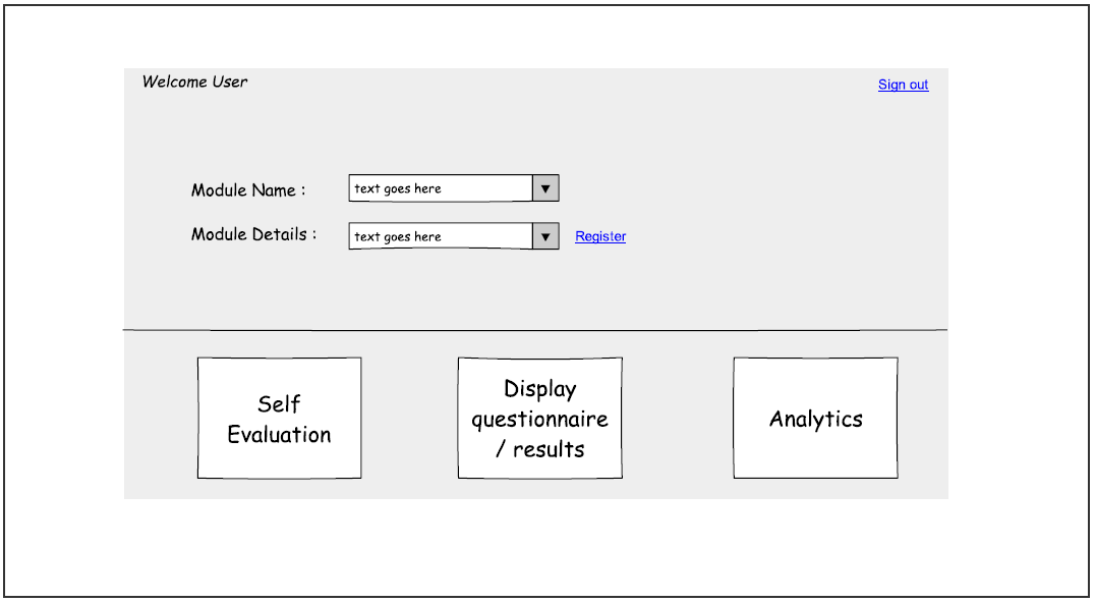


Figure 15: Screens - S_Home

When the student logs in, he is redirected to the home page which is specifically designed for his role (Figure 15) This screen wasn't a part of our paper prototype or the first version of

the pencil prototype[25]. To ensure uniformity between each user, the pages have similar components.

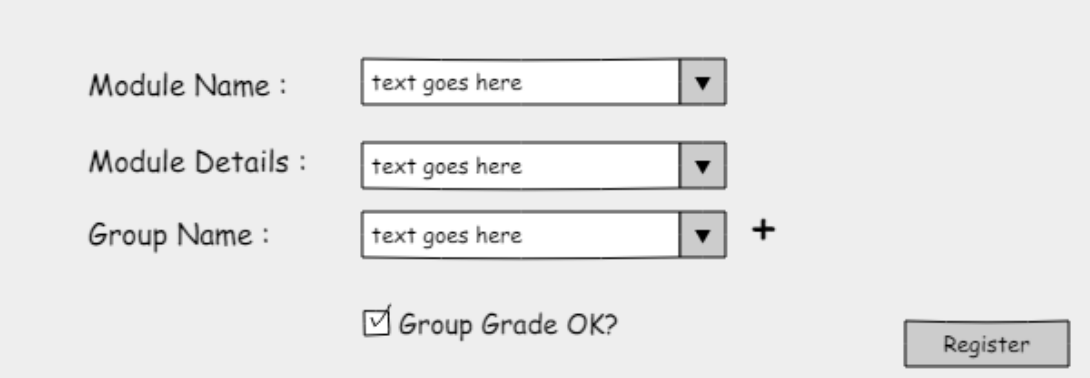


Figure 16: Screens - S_Register_popup

When the student logs in at the start of the module, he must register for the module. He registers by clicking on "register" on his home page which prompts a popup (Figure 16) .He should select from the modules available and provide his group details along with the details of when the module is taking place. In addition, an important aspect of grading in Germany is the legal requirements wherein the entire group cannot get the same grade unless all group members agree to it. Thus, we take the student's input in this regard.

[25] Annexure D

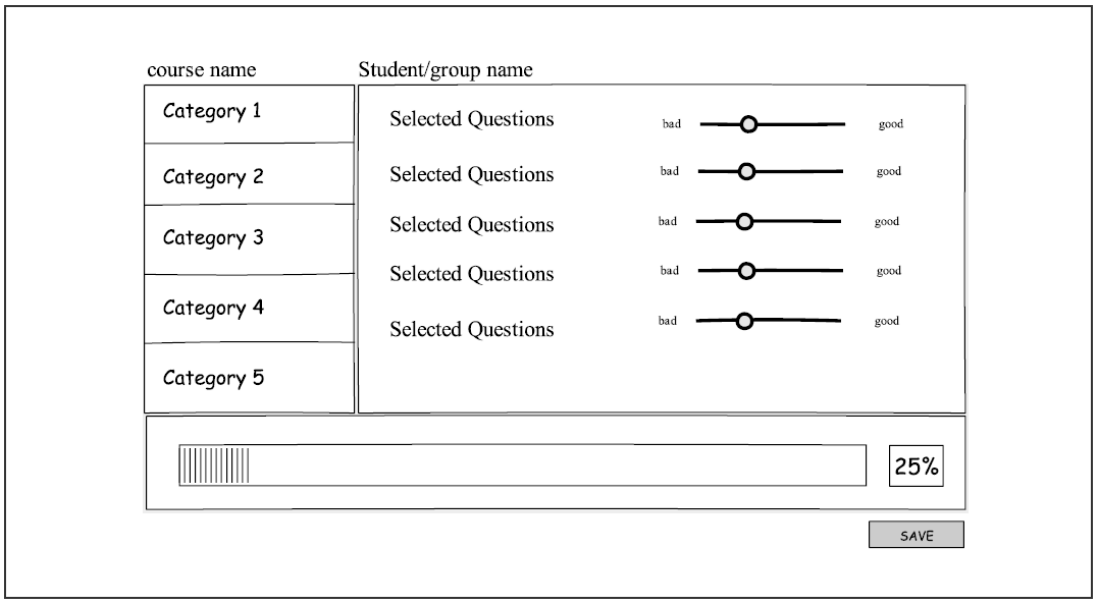


Figure 17: Screens - S_fill_self_eval

The student has to do a self-evaluation based on the questionnaire customized by the professor. Thus, when he clicks on “self-evaluation” on the home page he is redirected to the page that supports this feature (Figure 17)

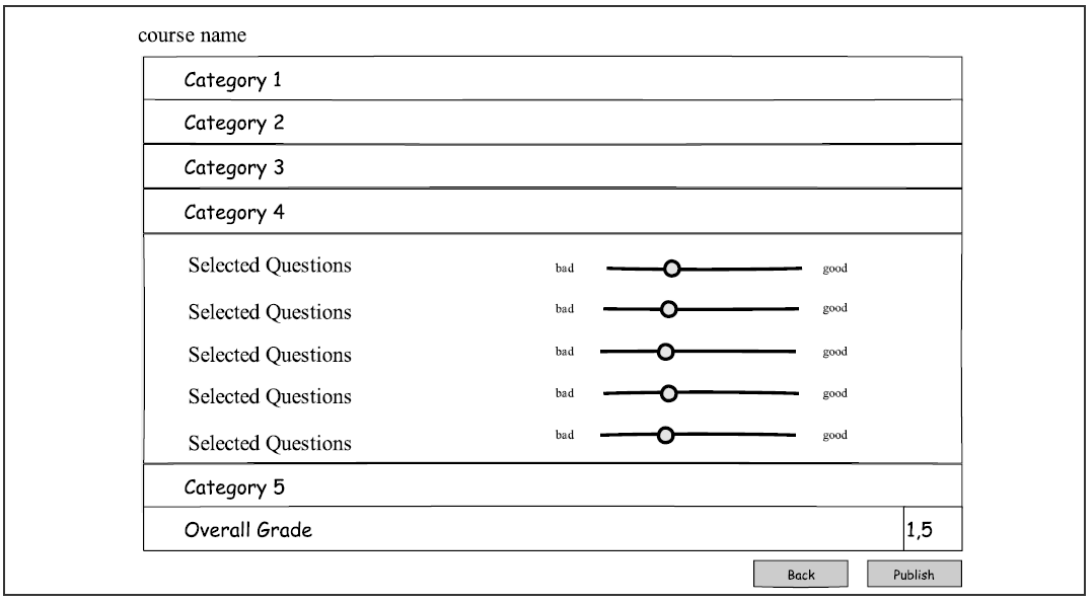


Figure 18: Screens - S_Review_self_eval

The student can review his self-evaluation before submitting it (Figure 18).

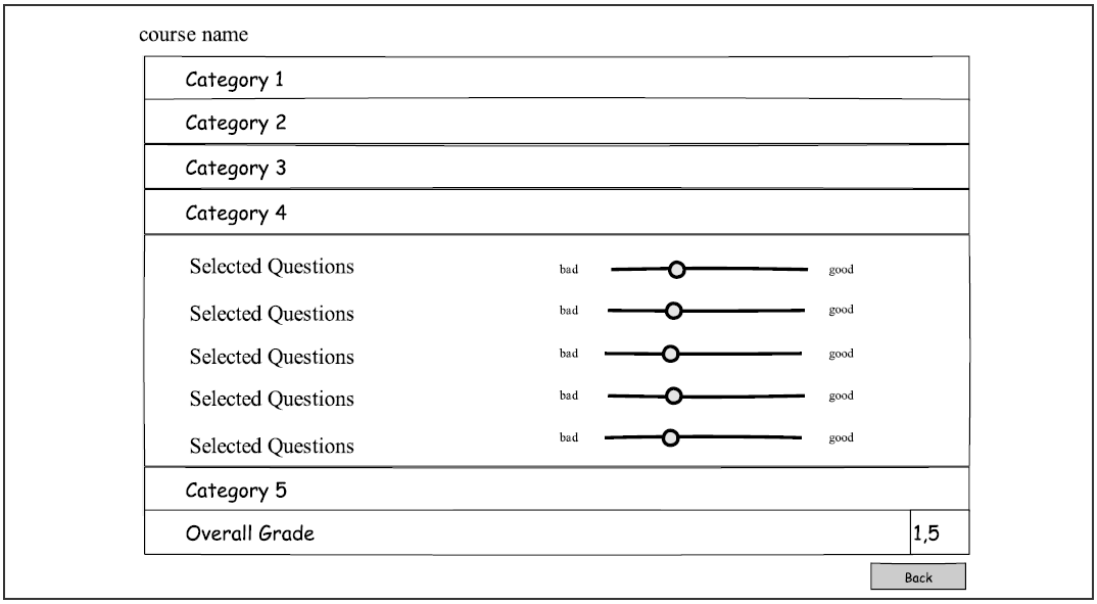


Figure 19: Screens - S_review_grade

At the end of the module, the student can check his grade by clicking on “review grade” on the home page. The student gets to see exactly how the professor filled out his feedback for along with grade associated with it (Figure 19).

Figure 20: Screens - PM_Home

In our final feedback[26] from the user, we found out that we would need a different role to standardize the questionnaires. Thus, The role we added was the one of a program manager[27]. If the user logs in as a program manager, he is directed the home page specifically designed for him (Figure 20).The program manager has to click on + on the home page to add new modules and create questionnaires for each module. These questionnaires will be reflected in the professor's profile to be customized (Figure 21).

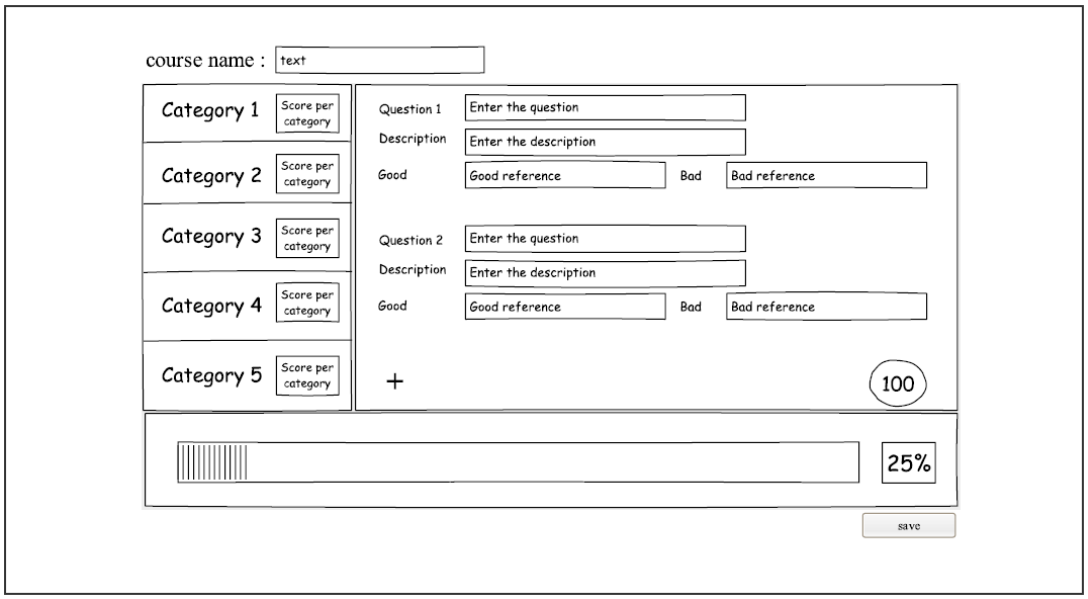


Figure 21: Screen - PM_create_Module

[26] Annexure G

[27] Annexure E

3.3 Requirement Coverage

While developing the user interface, we considered the requirements listed in the previous chapter. The Requirement that are fulfilled through this chapter are tabulated below:

No.	Requirement	Screens
1.	A Systematic grading system	P_fill, p_ fill_student_list, p_at_a_glance
2.	Standardized questionnaire	PM_create_module, P_create_course
3.	Transparency	s_review
4.	Cross Platform	-
5.	Analytics	-
6.	Feedback	-
7.	Exporting	-
8.	Central Login	-
9.	Multi-Language	-
10.	Research Support	S_fill_self_eval, s_review_self_eval
11.	Import from Scan	-

Table 13: Mid fidelity - requirement coverage

4 Technological Considerations

During the analysis phase of this software development project, it was identified that one of the key requirements is the ability to use the final product seamlessly on multiple platforms[28]. These platforms will include desktop systems running on Linux, MacOS or Windows as well as mobile devices (smartphones as well as tablets) running on iOS, Android or Windows. Since there will be multiple devices that all need to access the same data source, a network-based approach will be needed. Therefore, this project will implement a web-based application.

When developing a new web-based application, there are several considerations that need to be taken into account in regards of the architecture of the program. Specifying the architecture upfront will help during the development of the application itself. In addition, a clear architectural design in the beginning will minimize the risk of having to refactor code due to actually developing a "Big Ball of Mud", which is considered an anti-pattern in software development[29].

4.1 Overall Architecture

In general, a web-based application will always be implemented using separate layers. By its nature, there needs to be a server and a client-side of the application. When dealing with data, most likely there is also a need of implementing a database layer. As of this, the general architecture of this application will follow the pattern of a 3-layer architecture[30]:

[28] See requirement 4

[29] [LAPUTAN1]

[30] [FOWLER1], Page 19 ff.

Layer	Responsibility
Presentation	Provision of services, display of information, HTTP requests
Domain	Logic that is the real point of the system
Data Source	Communication with databases, messaging system, transaction managers, other packages

Table 14: Three Layers ([FOWLER1], Page 20)

4.2 Presentation

As described in the chapters before, this development project follows the approach of "User First". With reference to the three-layer architecture, the layer with which the user interacts is called the "presentation" layer. Therefore, a key decision to be taken at the beginning of the development is the specific architecture and technology used for this layer. As a base for this decision, a key requirement identified in this area is the development of an application that runs on several platforms with different screen sizes, resolutions and input methodologies (requirement 4). In order to support this requirement, the following solution possibilities have been discussed by the team:

1. Develop native applications for all platforms

   Using this approach means to develop separate applications for the major smartphone platforms (Android, iOS, Windows) as well as developing applications for the desktop platforms (MacOS, Windows, Linux). Even though there are several frameworks available to help develop cross-platform code (e.g. Xamarin or Cordova), this usually results in quite high development effort.

2. Using web-technology (HTML, CSS, Javascript) to deliver specific website layouts to the different platform

Using this approach, the web-application tries to determine, which device, screen size and browser the user accesses the website. With this information, the system delivers a website specifically created for the determined environment. This usually requires more initial development effort, but comes with benefits of rich user-experience. [31]

3. Using responsive web-design to adjust a websites Layout to different Screen sizes

   Responsive web-design adjusts the websites layout and / or functionality automatically by detecting the browsers size as well as the resolution and capability of the device accessing the website. With responsive web-design, the developer only have to maintain one version of the code, which usually leads to lower maintenance costs. However, responsive web-design often comes with higher initial development costs.[32] There are frameworks available which overcome these issues and ease the process of developing responsive websites.

As the project teams main focus lies in quick development of a prototype and an early version of the application, it was decided to use responsive web-design in order to achieve direct support for all platforms. If one or more features will require the development of native applications for the mobile platforms, this might be developed in a later stage.

In order to develop a responsive web application, there are several frameworks available that ease the development process. In general, it is a best practice to reuse existing frameworks rather than developing an own framework. For the purpose of this project, we decided on the following criteria in order to decide on the specific technology:

1. Responsive web-design

   With this criteria, we want to determine whether or not the framework provides out of the box support for developing responsive websites.

2. Support of modern web technology (HTML5, CSS3)

[31] [RPDVLSL1]

[32] [RPDVLSL1]

In general, it is considered a best practice to use modern web technology when implementing a new application. HTML5 as well as CSS3 help develop applications that are useable across all different clients.

3. Active Development

   Software development in general and web development in specific is an area which is actively developed and improved. In order to be able to comply with new standards as well as reacting on new security threats, there is a need to ensure the used framework is actively developed and receives periodic updates.

4. Other Key considerations

   Each framework has different outstanding benefits or weaknesses. This could e.g. be the cost of the framework, the licensing, documentation, the community or the support / focus on specific other technologies. In case the project team identifies these during the research, they will be taken into account accordingly.

The following frameworks have been considered:

Framework	Responsive	Modern	Active	Other Key considerations
Bootstrap	X	X	X	Most popular; Huge Community
W3.CSS	X	X	X	
Startup Design	X	X	X	SPA-focussed Not Free
Foundation	X	X	X	
YAML4	X	X	X	Not Free

Table 15: Presentation layer - technology overview

With this decision matrix, it was clear that all considered frameworks are supporting the key filter criteria the team defined. Therefore, the decision was based on the “Other Key considerations” which have mainly been the cost of the framework (Bootstrap, W3.CSS and

Foundation are free to use) as well as the community. An active community is very helpful in order to achieve a quick learning curve, since this means there are a lot of examples and tutorials available. The team therefore decided to use Bootstrap for developing the presentation layer.

4.3 Domain

Following the three layer approach as outlined in section 4.1 – Overall Architecture, this application will have a domain layer which will handle the server-side functionality of the application. In order to decide for server-side technology, the following criteria was identified to be relevant:

1. Development on multiple platforms

   In our development team, we have different operating system our developers are working on. In addition, future development[33] might be handled by different development teams of which we do not know which platforms they are using. Therefore, it was identified that there is a need for ensuring the availability of IDEs on all platforms (Windows, MacOS, Linux)

2. Cross-Platform Deployment / available webservers

   Similar to being able to develop on multiple platforms, there is also a need to run the application on multiple platforms. This supports both the possibility of local tests before synchronizing changes with the other developers as well as it makes the future product independent of a specific server landscape.

3. Patterns support

[33] see section 6

As described in section 4.2, there are several patterns available which are considered good / best practice when developing web applications.

4. Ease of use

   Since there are different levels of knowledge in the development team, it was identified that there is a need to be able to learn the used technology quickly in order to get up to speed with the development. In this regards, it is helpful to choose a technology that parts of the team are already familiar with.

5. Available Documentation

   As of the "Ease of use" criteria, it was identified that the availability of clear documentation is a key factor getting up to speed with development. "Clear documentation" in this regards means not only the availability of the documentation itself, but also a clear structure of it.

The following technologies have been considered:

Technology	Cross Platform Development	Cross Platform Deployment	Patterns	Ease of Use	Documentation
ASP.Net	-	-	+	++	++
ASP.Net Core	+	+	+	++	++
Node.JS	+	+	+	-	+
Python	+	+	+	+	+
PHP	+	+	+	-	+

Table 16: Domain layer – technology overview

With the given decision matrix, the team decided to use ASP.Net Core for the server-side development in this project. Main decision drivers have been the familiarity of parts of the team with this technology, the built-in pattern support as well as a clear and structured

documentation. As a template language, ASP.Net Core MVC uses the Razor syntax as a template language in order to enable interaction between views, models and controllers.[34]

4.4 Data source

The last layer to decide on is the data source layer. As described in section 4.1 Overall Architecture, this layer is responsible for storing data and providing it to the domain layer which then applies logic to it. Using abstraction patterns like ORM, the decision on the technology used for the database is less critical for the overall application. Therefore, this section will focus on deciding which ORM to use. Since the domain layer was decided to use ASP.Net Core, the following ORMs are available[35]:

ORM	Database Support
Entity Framework Core	Microsoft SQL Server, SQLite, PostgreSQL, MySQL, Microsoft SQL Server Compact Edition
Dapper	All ADO.Net Data providers
NPoco	Firebird, MySQL, Oracle, PostgreSQL, SQLite, Microsoft SQLServer

Table 17: Datasource layer - technology overview

Even though there are multiple ORMs available for accessing data that work with .Net Core, Entity Framework Core was developed by Microsoft and is seamlessly integrated in ASP.Net Core. In addition, most of the available tutorials in this area target Entity Framework Core. As of this, the team decided to use Entity Framework Core for this development Project. In the development environment, each developer can decide themselves, which exact database to use.

[34] [MCRSFT1]

[35] [MCRSFT6]

Combining the technological considerations with the architectural decisions, the following picture applies for this application:

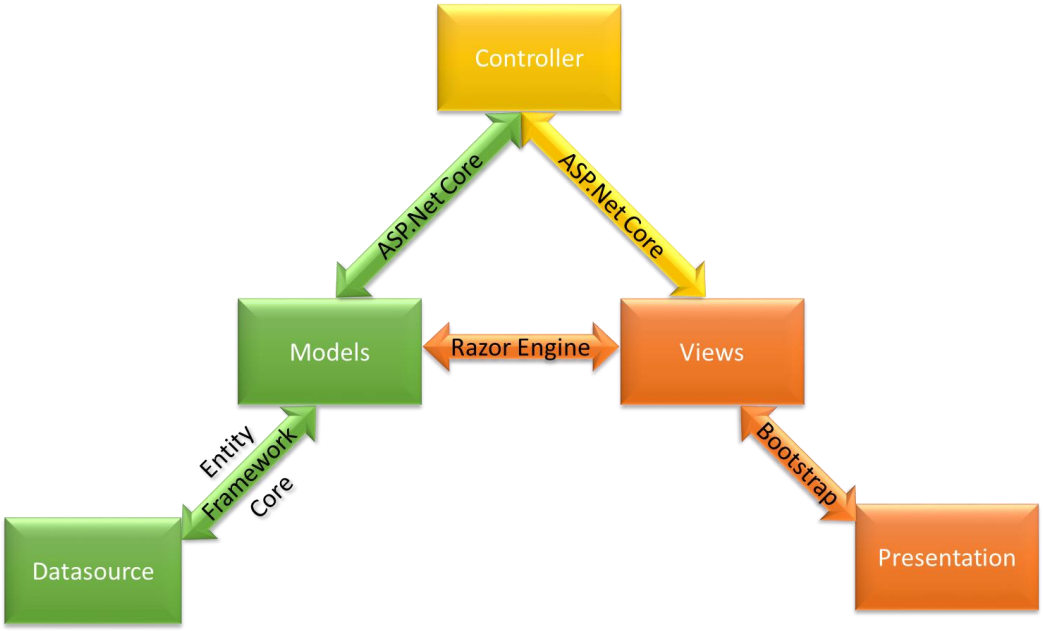


Figure 22: Architecture and technology

5 High Fidelity Prototype

After having created the mid fidelity prototype and having tested it, there is one more step from the final development: high fidelity prototype.

A high fidelity prototype is a computer-based interactive representation of the product in its closest resemblance to the final design in terms of details and functionality. The high fidelity prototype covers not only the user interface (UI) of the product in terms of visuals and aesthetics, but also the user experience (UX) aspects in terms of interactions, user flow and behavior. The main purpose of high fidelity prototypes is their use in the usability testing of the product have target users validate it. However, we have already used the mid fidelity prototype to test the interactions, user flow and behavior. It is important to test the product before launching it in the market to foresee any issues or failures. Getting the most out of the feedback can be done with a prototype that is closest to the final product in its detail and functionality.[36]

The availability of a high fidelity prototype can improve the collaboration with developers as they will have a clearer idea on how the application should behave. High fidelity prototypes provide a good base in terms of project management for making estimates on how much time is needed for implementation and quality assurance testing. Most importantly, by allowing you to test your product, the prototypes can save the company the cost in terms of time and money on building something that would have had little success in the market.[37]

However, there are drawbacks in using high fidelity prototype. High fidelity prototypes normally cost more time and resources to create compared to low fidelity prototypes. In addition, it is more time-consuming to change the designs, so it is harder to make on-the-go fixes during usability testing if you want to update your prototype quickly to get better feedback.

[36] [MOBGEN1]

[37] [NNGRP1]

In order to create the high fidelity prototype, you need to prepare all your ingredients ready. Get your visuals and set your user flows. Identify all the transitions and animations you want to create between different objects or screens.[38]

Although high fidelity prototype almost looks like the final version, we still need to modify it according the testing.

5.1 Development Environment

Based on the considerations in chapter four, we decided to use Bootstrap for the presentation layer; ASP.NET Core for the domain layer and Entity Framework Core for data source layer. Since we are developing on different platforms with different database support, developers on the different platforms decide on their own which specific database they want to use. On windows computers, Visual Studio 2017 (community edition) will be used as the integrated development environment (IDE). On MacOS, Visual Studio Code will be used.

Bootstrap is a front-end development framework. It includes HTML, CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many other, as well as optional JavaScript plugins. So Bootstrap gives us the ability to easily create responsive designs.[39] Since we are using Microsoft Visual Studio, we can choose the default starter template for ASP.NET Core, in which case Bootstrap will come pre-installed.

ASP.NET Core is a free and open-source web framework, and the next generation of ASP.NET, developed by Microsoft and the community. It is a modular framework that runs on both the full ASP.NET Framework, on Windows, and the cross-platform ASP.NET Core.

Microsoft Visual Studio Code is a lightweight source code editor which is developed by Microsoft for Windows, Linux and OS X. It includes support for debugging, embedded Git control, syntax highlighting, intelligent code completion, snippets, and code refactoring. It is

[38] [MOBGEN1]

[39] [BTSTRP1]

also customizable, so users can change the editor's theme, keyboard shortcuts, and preferences.[40]

At the end of this prototyping project, Microsoft released the final version of Visual Studio 2017 for Mac.[41] Therefore the team will check the new software in the next course and decide whether to have the developers working on MacOS to the new release.

5.2 Model-View-Controller pattern

ASP.NET Core ships with MVC core. ASP.NET Core MVC is a web application framework which implements MVC pattern.

Model View Controller (MVC) pattern is a software architectural pattern for implementing user interfaces on computers. It separates an application into three interconnected components: *model*, *view*, and *controller*. This pattern helps to achieve separation of concerns. It can separate internal representations of information from the ways that information is presented to and accepted from the user. Popular programming languages like Java, C#, Ruby, PHP and others have popular MVC frameworks that are currently being used in web application development straight out of the box.[42] [43]

[40] [JHNPP1]

[41] [MCRSFT4]

[42] [MCRSFT1]

[43] [MCRSFT2]

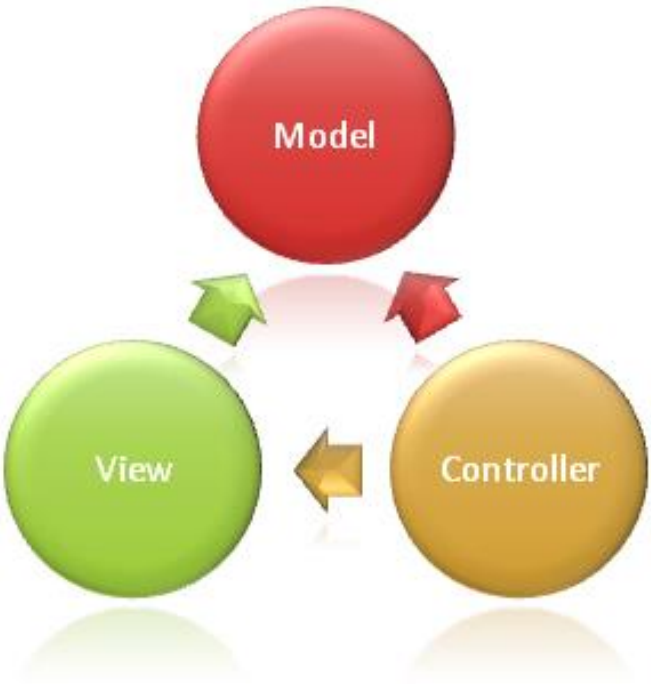


Figure 23: MVC pattern[44]

Figure 23 shows the three main components and the interactions between them.

- The *model* is responsible for maintaining the state of the application in terms of the problem domain, independent of the user interface. It directly manages the data, logic and rules of the application. In other words, the model stores data that is retrieved according to commands from the controller and displayed in the view. In complex applications, it makes sense to have different model types (classes), such as: domain model, view model, binding model, API model and persistence model. By using different model types can sort the data in a more efficient way.

- The *view* is responsible for presenting the content of the application through the user interface (UI). It can be any output representation of information, such as a chart or a diagram. Multiple views of the same information are possible, such as a bar chart for management and a tabular view for accountants. Also the view will generate new output to the user based on changes in the model. [45]

- The *controller* which is the initial entry point, handles user interaction, work with the model, and ultimately select a view to render. In other words, it accepts input and converts it to commands for the model or view, by sending commands to the model

[44] [MCRSFT1]

[45] [SCTGU1]

to update the model's state. It can also send commands to its associated view to change the view's presentation of the model.

5.3 Front-End Development

The front end development of the application is the part that users interact with. Everything that the users see, touches and experiences when the users are navigating around the Internet, from fonts and colors to dropdown menus and sliders, is a combination of HTML, CSS, and JavaScript being controlled by user's computer's browser. The objective of designing a website is to ensure that when the users open up the website they see the information in a format that is easy to read and relevant. This is further complicated by the fact that users now use a large variety of devices with varying screen sizes and resolutions thus forcing the designer to take into consideration these aspects when designing the site. We need to ensure that our site comes up correctly in different browsers (cross-browser), different operating systems (cross-platform) and different devices (cross-device) [Requirement 4], which requires careful planning on the side of the developer.

Following the MVC patter as described before, the following views have been identified as needed for this software:

Controller	View	Responsibility
CourseController	Create	Allow the professors to create the course. [mid fidelity: P_Create_Module]
CourseController	Display	Display the page for the program manager, professors and the students to view the courses. [mid fidelity: PM_Home,

		P_Home, S_Home]
CourseController	Grade	Entry point for the professors to grade the students. This View provides the student / group selection. [mid fidelity: P_fill_student_list]
CourseController	Register	Showing the course list for the student to register. [mid fidelity: PM_Create_Module]
HomeController	Homepage	Display the home page for both the professors and the students to login the system.
ModuleController	Create	Allow the professors to create the module. [mid fidelity: P_Create_Course]
ModuleController	Display	Display the page for the professors and the students to view the module.
QuestionnaireController	Create	Let the professors create the correspond questionnaire for the module.

QuestionnaireController	Create_Template	The page is where the program manager can create the questionnaire template for the module.
QuestionnaireController	Display	Showing the questionnaire to the professors for further modify or publish it. Also the students can review their own grade at this view. [mid fidelity: P_At_a_glance]
QuestionnaireController	Fill	Allow the professors to fill out the questionnaire for each student/group. [mid fidelity: P_Fill]

Table 18: Controller – View and their Responsibility

5.4 Back-End Development

The back end development of a website is a combination of a server, an application, and a database. In other words, it means all the components about the data which the website needs. And in the MVC pattern, it concludes the model and the controller.

5.4.1 Database (Data sources)

A well designed database is capable to provide up-to date and accurate information. For database design, we need to produce a detailed data model of database in order to fully understand what kind of data we really need. To clarify it, we have to create the Entity Data Model (EDM), it uses three key concepts to describe the structure of data: *entity type*,

association type and *property*.[46] These are the most important concepts in describing the structure of data in any implementation of the EDM.

- *Entity type* is the fundamental component which is constructed from properties and describe the top-level concept. In other words, entity type is a template for entities. An entity represents a specific object and each entity must have a unique entity key within an entity set. An entity set is a collection of instances of a specific entity type.

- *Association type* is for describing the relations in the EDM by representing the two relationships between two entity types. Relationships can be classified into three types: one-to-one, one-to-many and many-to-many.

- Entity types contain *properties* that describe their structures and characteristics. A property can also contain primitive data.[47]

[46] [MCRSFT3]

[47] [ROMAN1], page 12 ff.

After finishing the mid fidelity prototype, the following EDM was developed:

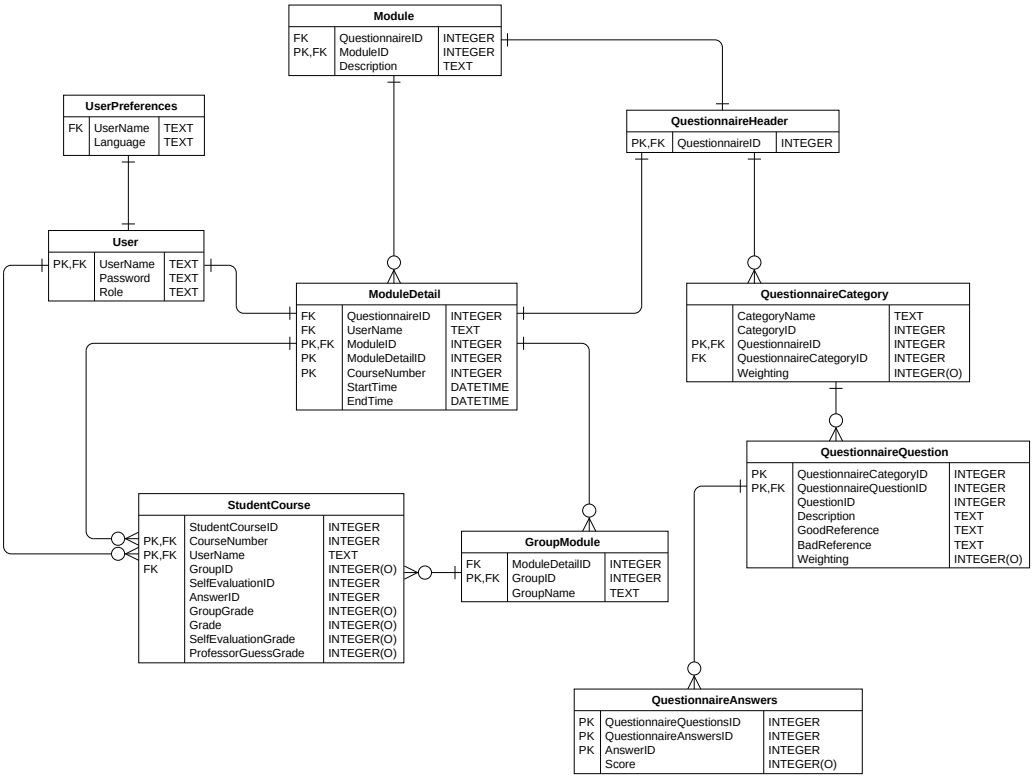


Figure 24: Entity Data Model

In the EDM, we use two different patterns to describe different relationships between the entity types.

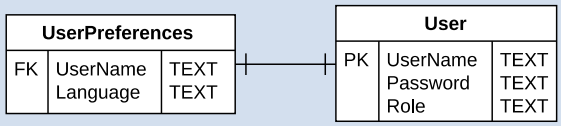
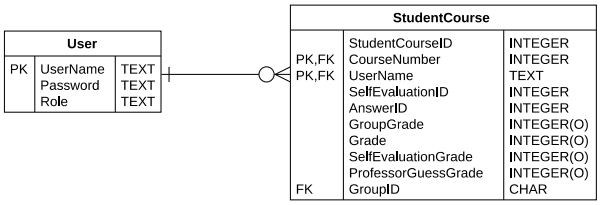
Type	Pattern	Description
One-to-one		A user preference has only one user, and a user has only one user preference.
One-to-many		A user can has many student course, and a student course only has one user.

Table 19: Different relationship pattern

In the diagram, we describe the data type and indicate whether the field is primary key or foreign key by PK and FK. Also, if the field is optional, there will be (O) behind the data type. Otherwise it means the field is required.

5.4.2 Models and back-end (Domain)

ASP.NET Core is a rich framework which follows the MVC pattern. As mentioned in Section 5.2, in the MVC pattern, models contain the representation of the data that the application needs. A view model is a type that includes just the data a view requires for display. Besides, view model types can also simplify model binding in ASP.NET Core MVC pattern. View model types are generally just data containers; any logic they may have should be specific to helping the view render data. We are using Entity Framework Core in order to implement any needed models.[48]

The models needed for this application are directly described in the EDM. Each entity within the EDM represents a model in our project. In order to provide user authentication, ASP.Net Core ships with ASP.Net Core Identity[49], which will be used in this project as an interim solution until the central login (requirement 8) is being developed.

[48] [MCRSFT1]

[49] [MCRSFT5]

5.4.3 Controllers

In the MVC pattern, controllers handle and respond to user input and interaction, and then passes information to the view.[50]

Based on the views needed for user interaction and described in section 5.3, the following controllers will be developed:

Controller	Responsibility
AccountController	Shipped with ASP.Net Core Identity
ManageController	Shipped with ASP.Net Core Identity
CourseController	Manages the course, provides create and display functionality
HomeController	The standard controller, mainly handles the landing page
ModuleController	Manages Modules, provides create and display functionality
QuestionnaireController	Manages the Questionnaire, provides functionality to create template questionnaires, specific questionnaires, review and fill them.
TechnicalController	Will be needed during the development phase. Provides database handling functionality as applying migrations, dropping data and inserting sample data.

Table 20: Controller responsibilities

[50] [SCTGU1]

6 Outlook on Future Development

This document described the prototyping process within the Software Architecture and Development course. The next course will be Applied Computer Science in which the product will be developed until a first usable version (minimum viable product) can be provided.

6.1 Minimum viable product

Given the requirements described in section 2.6, the team reviewed them and prioritized the ones that are seen as relevant for creating a minimum viable product. This does not mean that other requirements might not get implemented in case there is enough time until the first release, but the following requirements have priority:

No.	Requirement	Description
1.	A Systematic grading system	The system needs to provide the ability to grade users based on defined criteria and questions.
2.	Standardized questionnaire	The system needs to provide the ability to use standardized questionnaires for courses
3.	Transparency	The system needs to provide transparency about the given grades to all different kinds of users
4.	Cross Platform	The system must work on all the platforms, screen sizes and operating systems

Figure 25: Requirements - minimum viable product

6.2 Requirements out of Scope

The following requirements are already foreseen as too complex for the given timeframe and will therefore be out of scope within the Applied Computer Science Course.

No.	Requirement	Description
8.	Central Login	The system needs to use the central login credentials at SRH
11	Import from Scan	The system needs to provide the ability to scan printed questionnaires and automatically import them.

Figure 26: Requirements - out of scope

7 References

7.1 Literature

[FOWLER1]	Martin Fowler Patterns of Enterprise Application Architecture Thirteenth printing, 2007 ISBN 0-321-12742-0
[KEES1]	Dorst, Kees Frame Innovation: Create new thinking by design. Cambridge, MA: MIT Press. , 2012 ISBN 978-0-262-32431-1.
[SJWM1]	User Interface Design and Evaluation Debbie Stone, Caroline Jarrett, Mark Woodroffe, Shailey Minocha Morgan Kaufmann, 2005 ISBN 978-0-120-88436-0
[ROMAN1]	Steven Roman Access Database Design & Programming "O'Reilly Media, Inc.", 2002 ISBN 9-780-596-00273-2

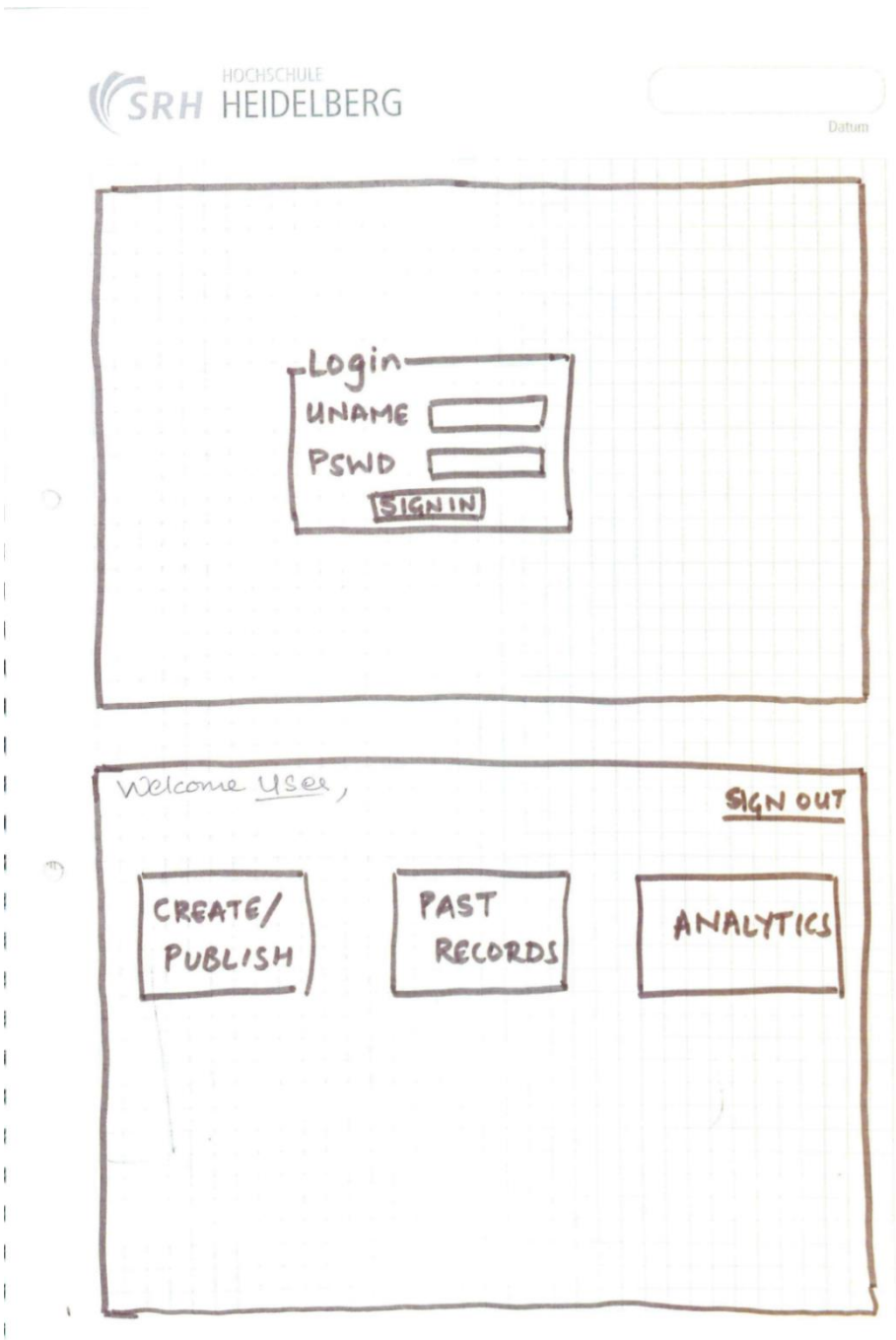
7.2 Web

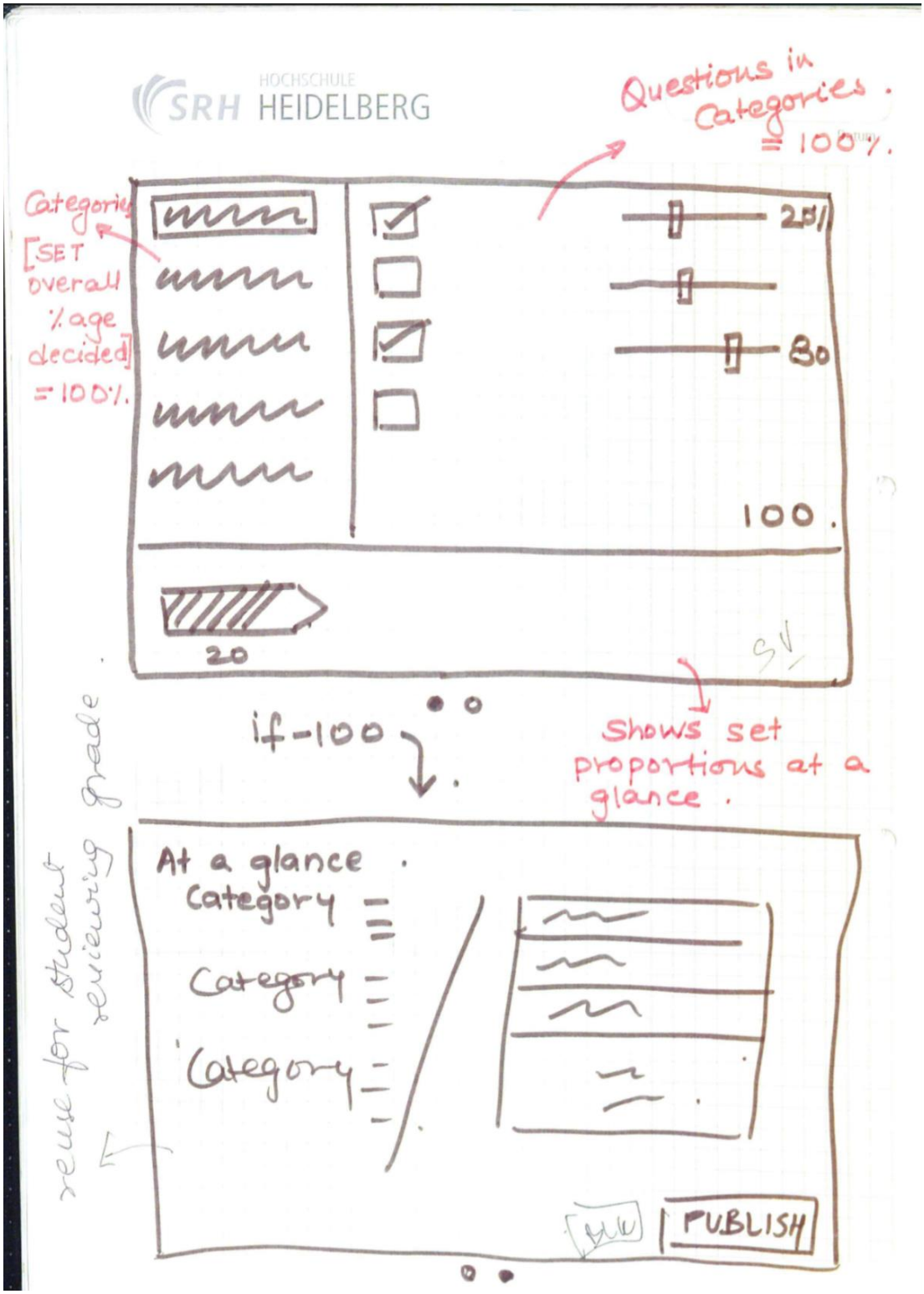
[LAPUTAN1]	http://www.laputan.org/mud/mud.html#BigBallOfMud
[CREATWORK1]	http://www.creativityatwork.com/design-thinking-strategy-for-innovation/
[MNTNGTS1]	https://www.mountaingoatsoftware.com/agile/user-stories
[MNTNGTS2]	https://www.mountaingoatsoftware.com/articles/advantages-of-user-stories-for-requirements
[ONLNMRKTPRA]	http://www.onlinemarketing-praxis.de/glossar/personas
[VISHIER1]	https://visualhierarchy.co/blog/the-continuum-of-high-and-low-fidelity-prototypes/
[HGHMHED1]	http://highered.mheducation.com/sites/0077110005/student_view0/glossary.html
[DREWL1]	http://www.drewlepp.com/blog/5-big-benefits-design-thinking/
[CMSWRE1]	http://www.cmswire.com/customer-experience/why-we-all-need-design-thinking/
[BLINKUX1]	https://blinkux.com/blog/the-fine-art-of-creating-experience-maps/
[MOBGEN1]	https://mobgen.com/high-fidelity-prototyping/
[NNGRP1]	https://www.nngroup.com/articles/ux-prototype-hi-lo-fidelity/
[MCRSFT1]	https://docs.microsoft.com/en-us/aspnet/core/mvc/overview
[MCRSFT2]	https://msdn.microsoft.com/en-us/library/dd381412(v=vs.108).aspx
[MCRSFT3]	https://msdn.microsoft.com/en-us/library/ee382825(v=vs.110).aspx
[SCTGU1]	https://weblogs.asp.net/scottgu/asp-net-mvc-framework

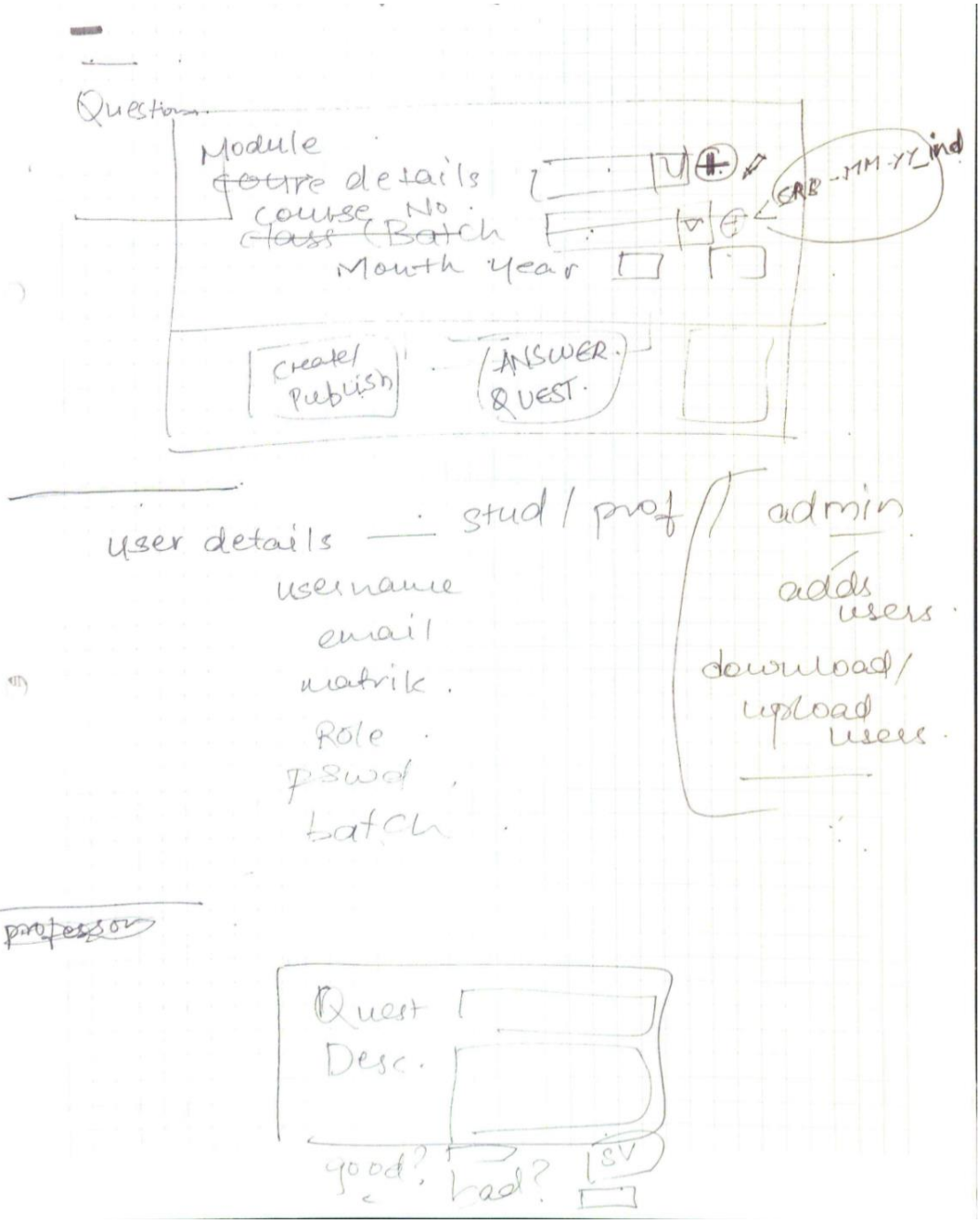
[JHNPP1]	https://johnpapa.net/visual-studio-code/
[BTSTRP1]	http://getbootstrap.com/
[MCRSFT4]	https://blogs.msdn.microsoft.com/visualstudio/2017/05/10/announcing-new-innovations-at-microsoft-build-2017/
[MCRSFT5]	https://docs.microsoft.com/en-us/aspnet/core/security/authentication/identity
[RPDVLSL1]	http://www.rapidvaluesolutions.com/whitepapers/responsive-web-design.html
[MCRSFT6]	https://blogs.msdn.microsoft.com/dotnet/2016/11/09/net-core-data-access/

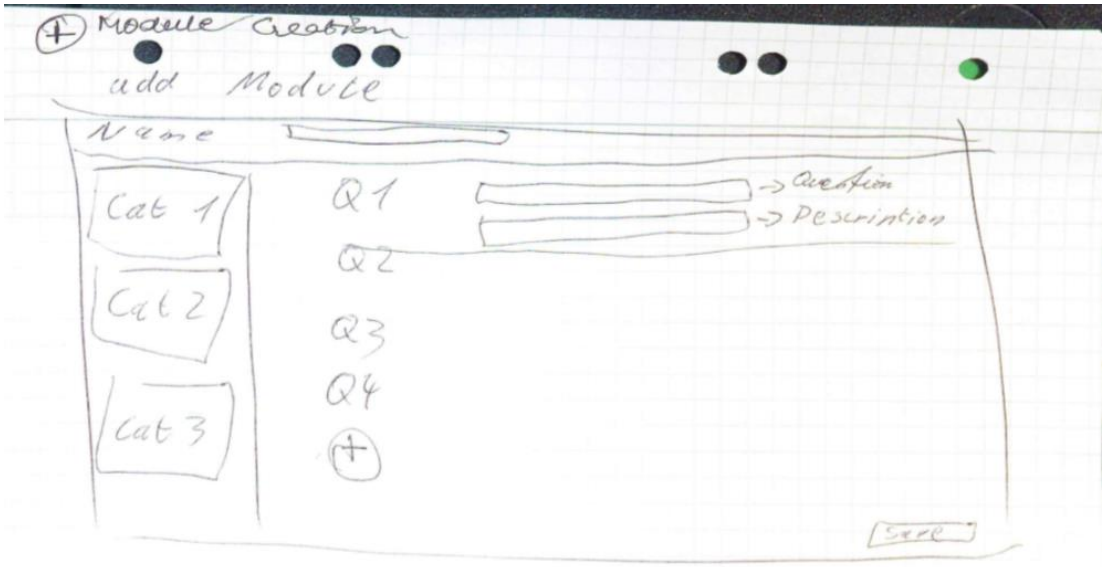
8 Annexures

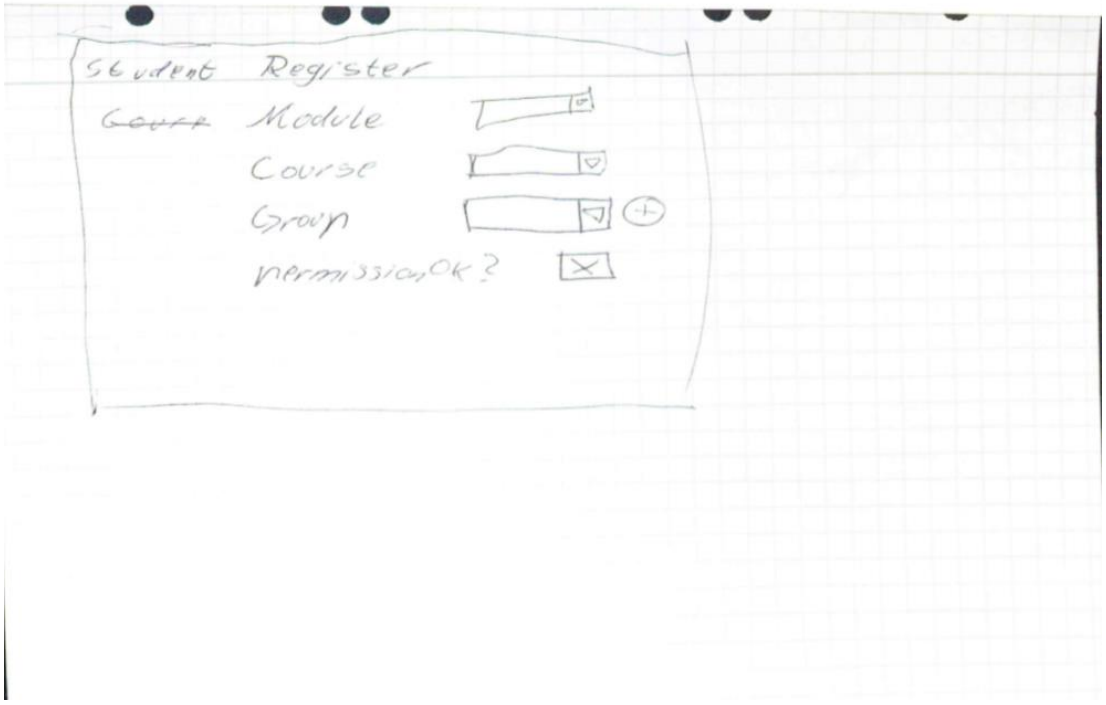
8.1 Annexure A – Paper Based prototype

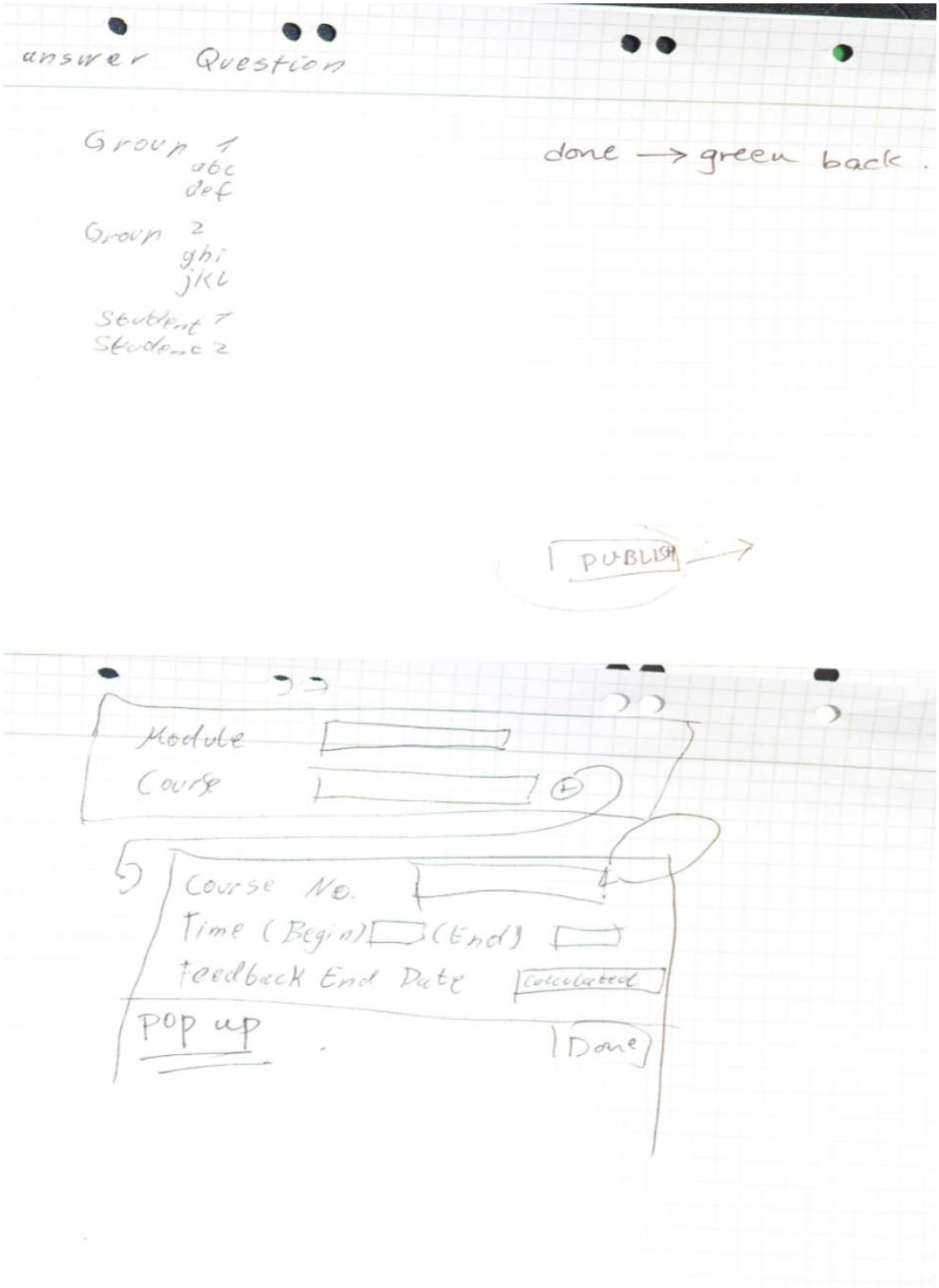



HOCHSCHULE
SRH HEIDELBERG
Questions in Categories. = 100%.
Category [SET overall %age decided] = 100%.
25
80
100.
20
if-100
Shows set proportions at a glance.
reuse for student reviewing grade.
At a glance.
Category
Category
Category
PUBLISH


Question
Module
details
course No.
Batch
Month Year
ERR - MM-YY_ind
Create/ Publish
ANSWER QUEST.
user details
stud / prof / admin
username
email
matrik.
Role
Pswd
batch
adds users
download/ upload users
professor
Quest
Desc.
good?
bad?
SV


Module Creation
add Module
Name
Cat 1
Cat 2
Cat 3
Q1
Q2
Q3
Q4
-> Question
-> Description
Save

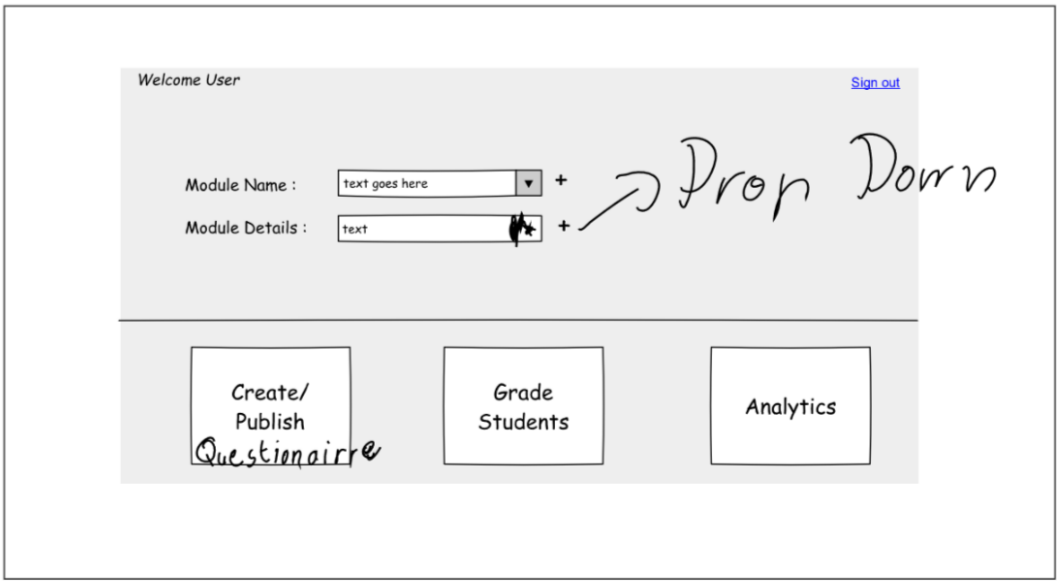

Student Register
Module
Course
Group
permission OK?


answer Question
Group 1
abc
def
Group 2
ghi
jkl
Student 1
Student 2
done → green back.
PUBLISH
Module
Course
Course No.
Time (Begin) (End)
feedback End Date
Calculated
POP up
Done

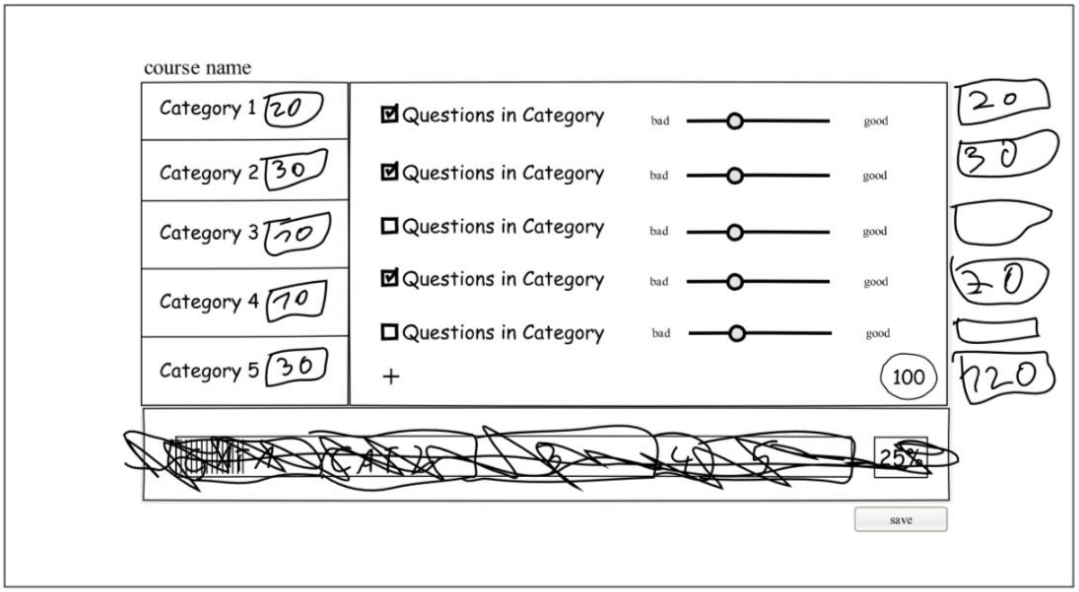
8.2 Annexure B – Mid fidelity iteration 1

Login_

P_home

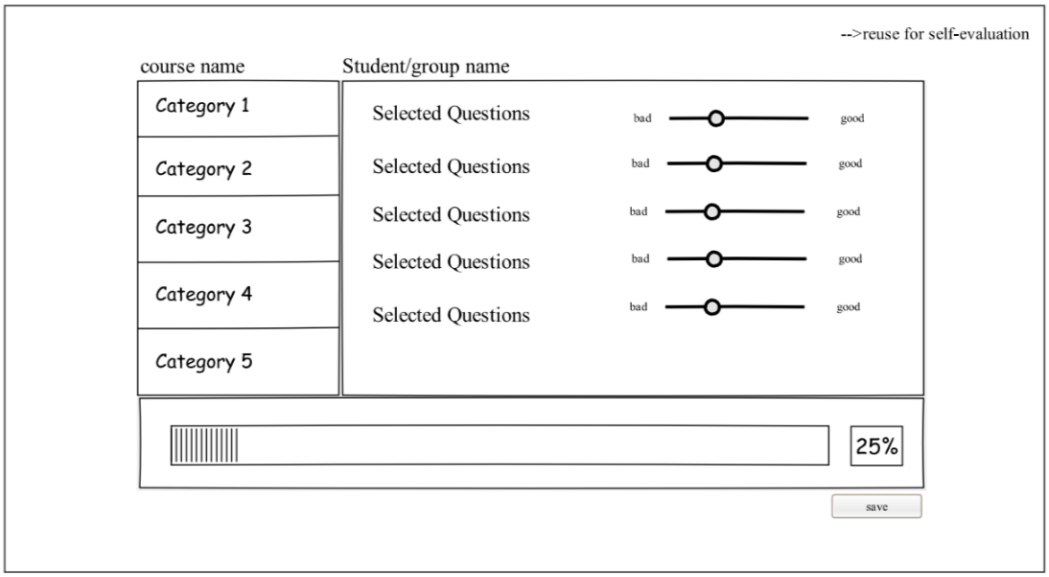


P_create

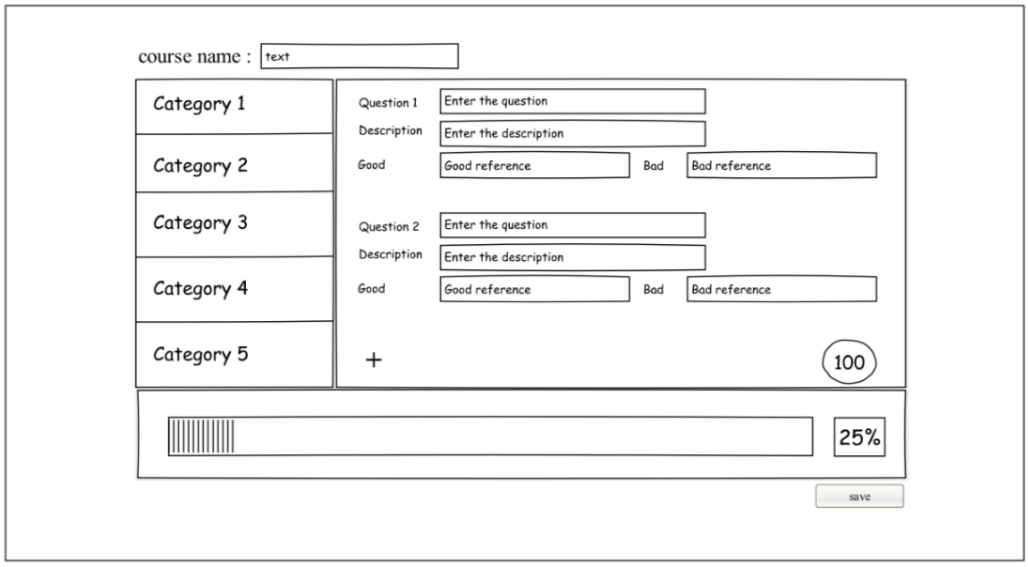


P_analytics

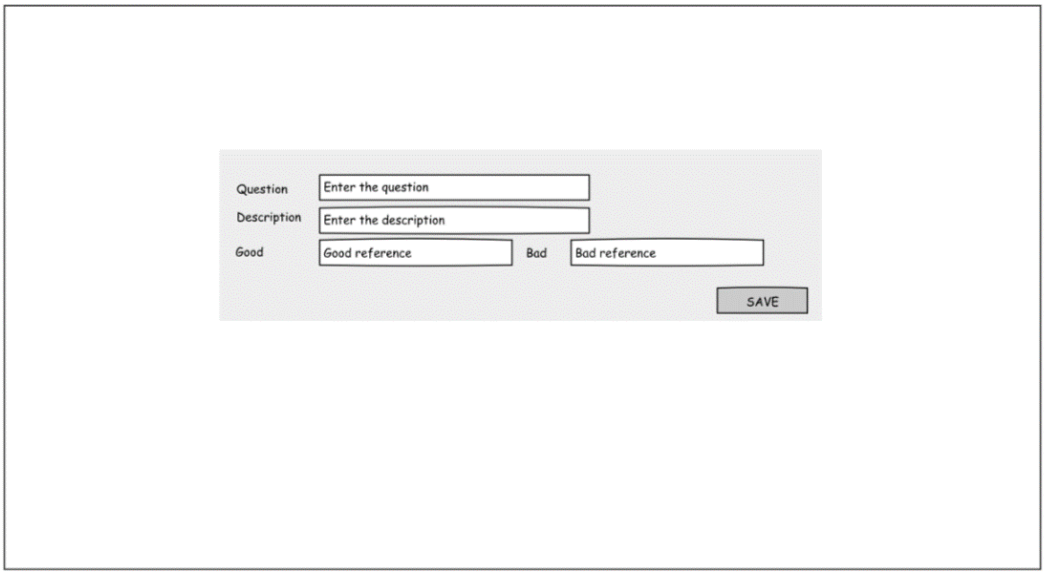
P_fill



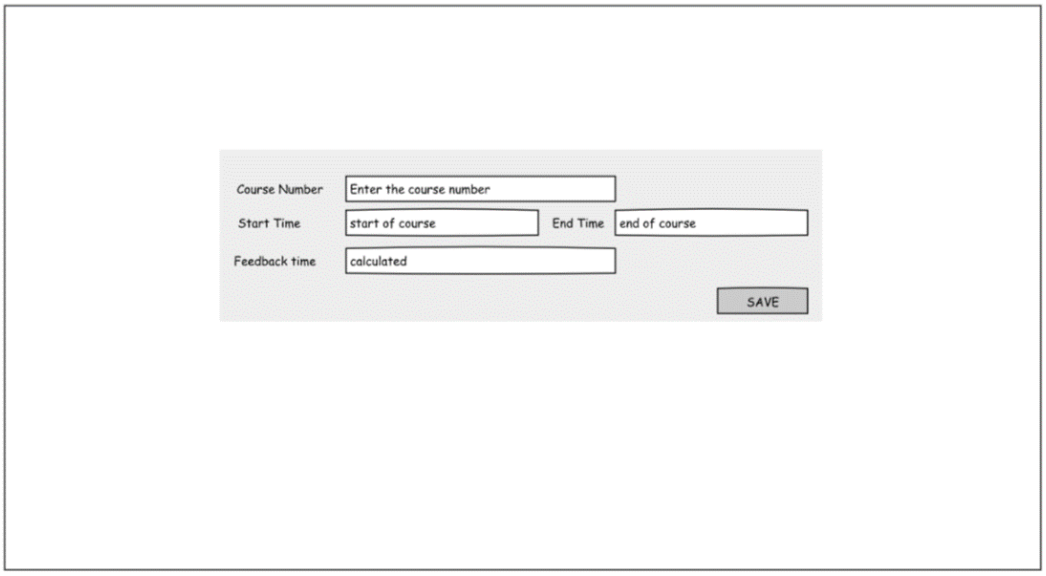
P_create_module



New_question_popup



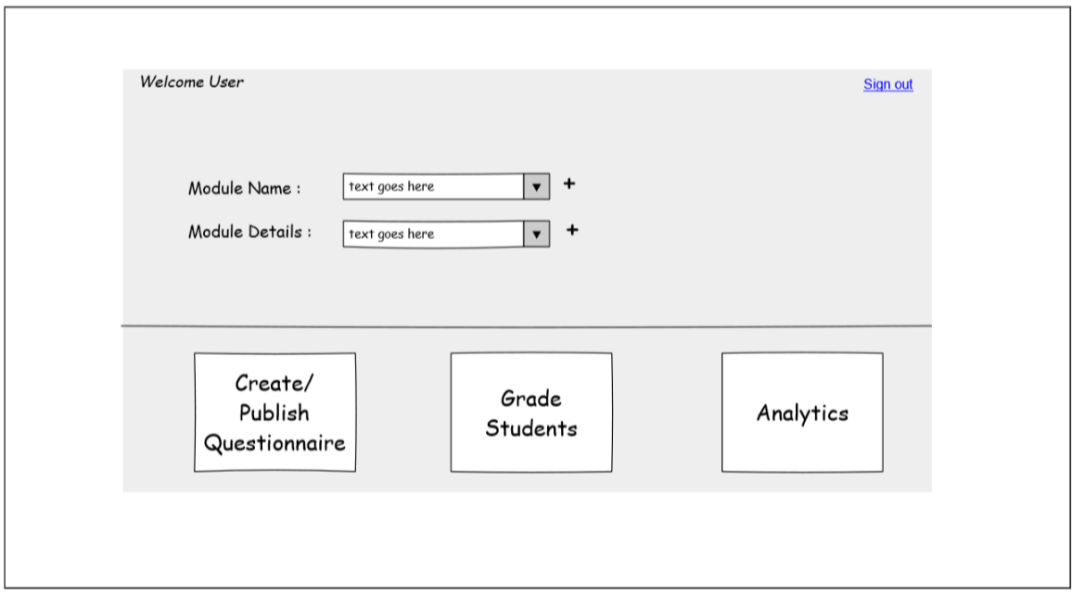
Module_detail_popup



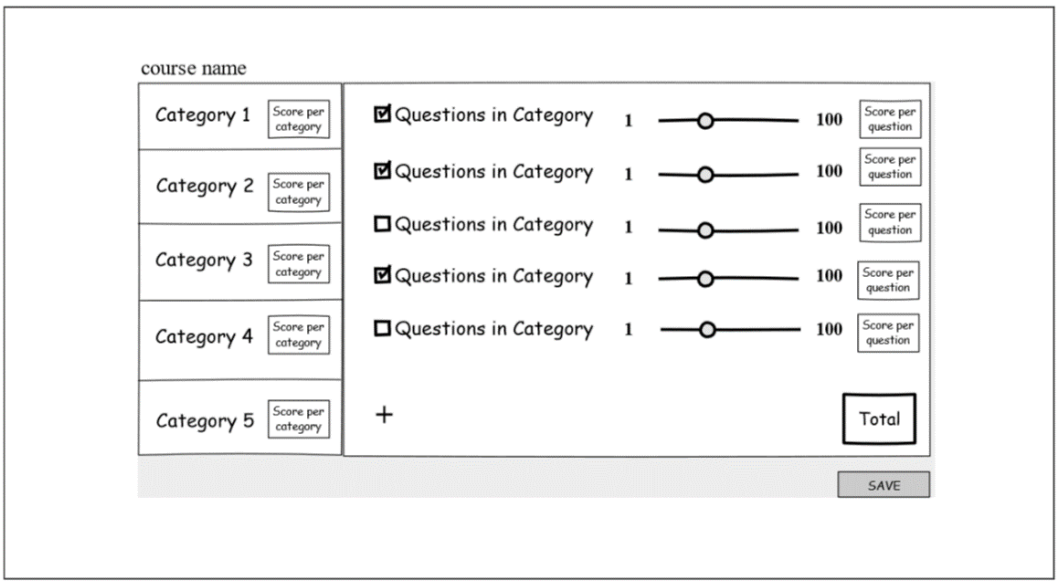
8.3 Annexure C – Mid Fidelity Iteration 2

Login_

P_home

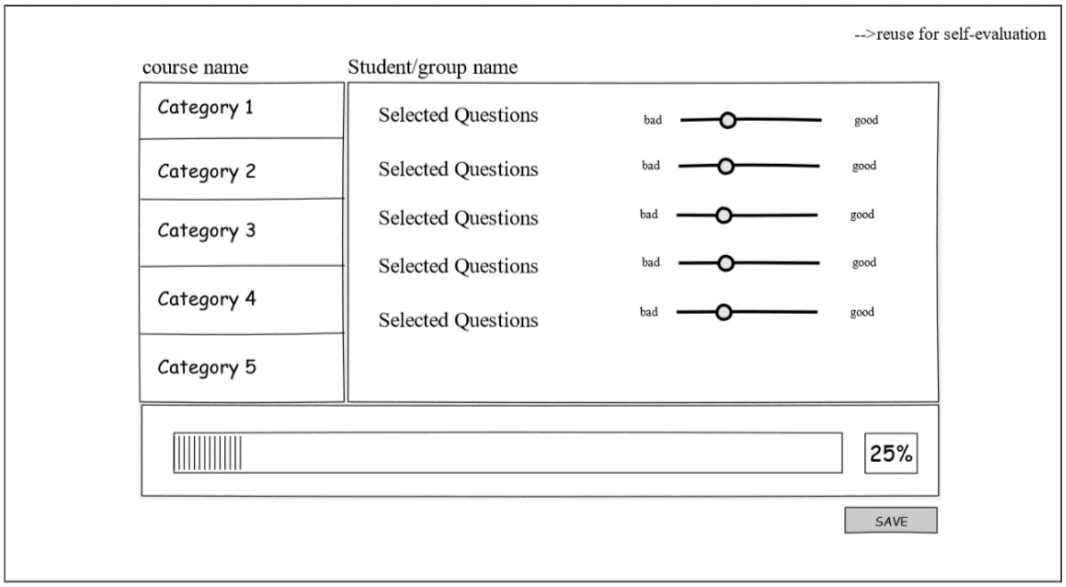


P_create

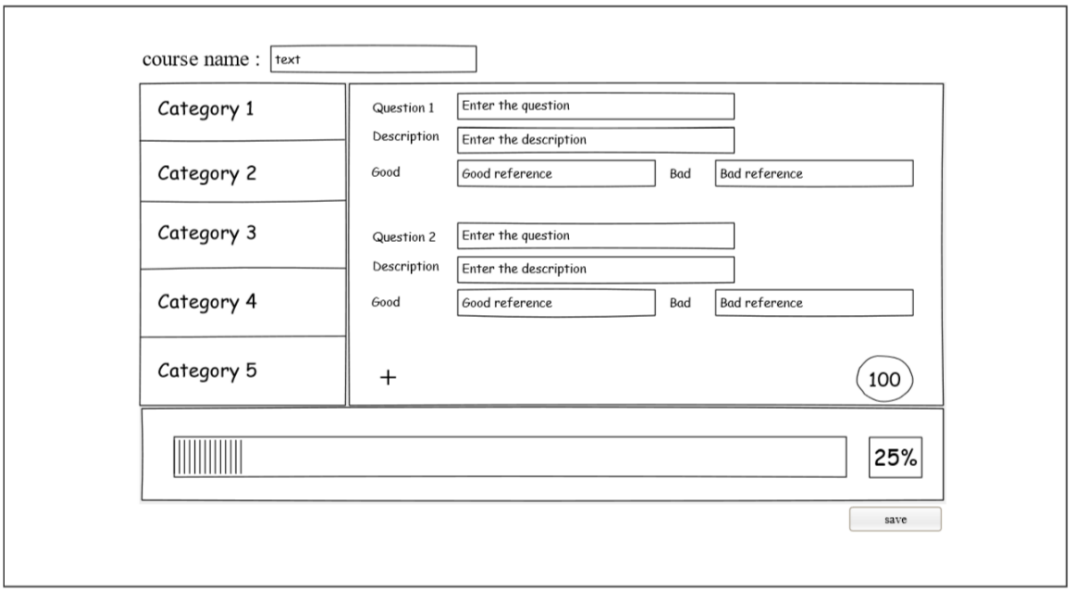


P_analytics

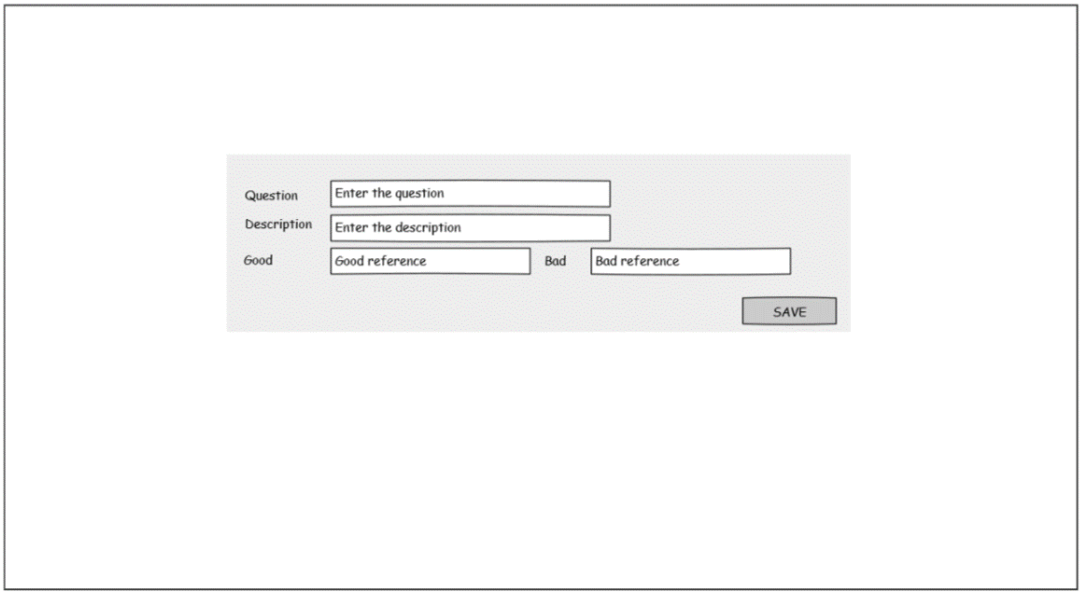
P_fill



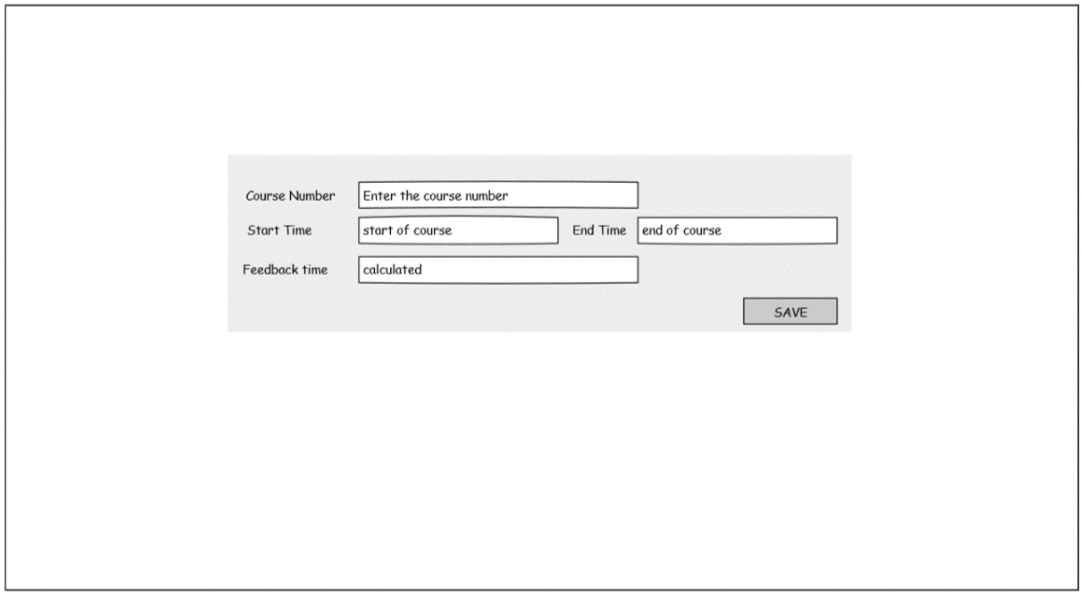
P_create_module



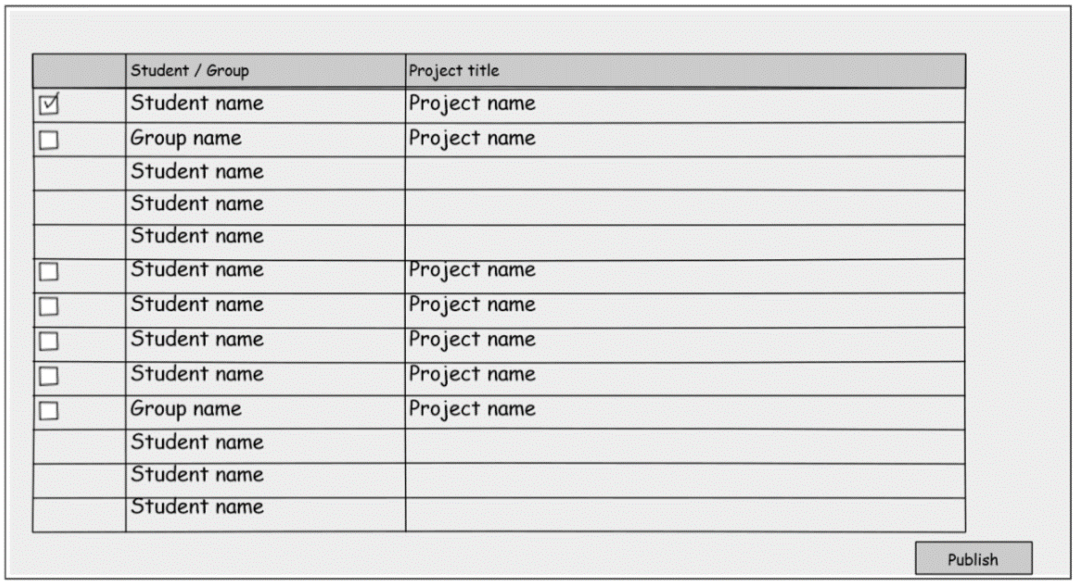
New_question_popup



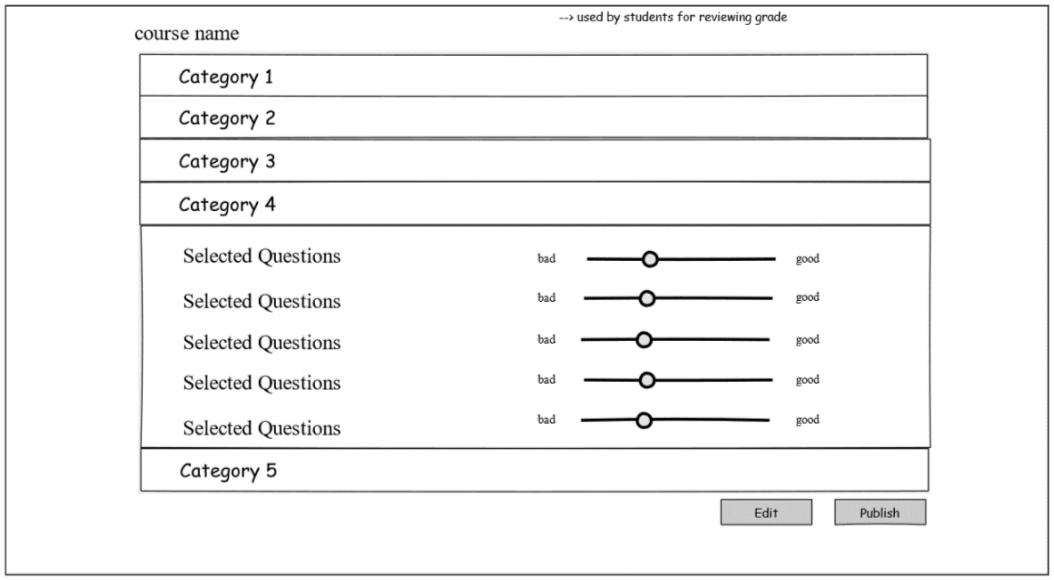
Module_detail_popup



student_list

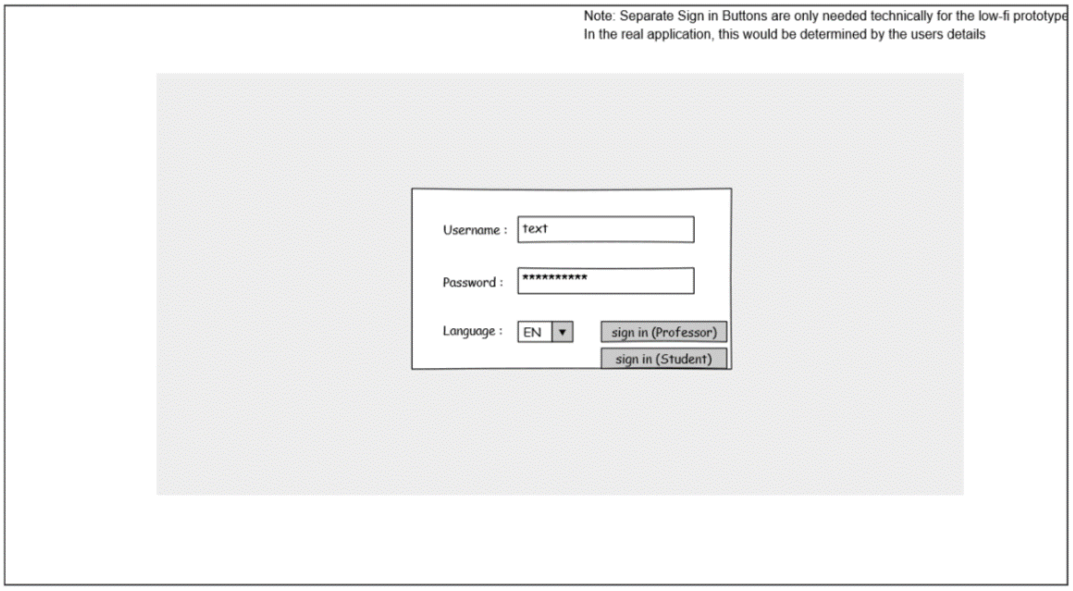


At_a_glance

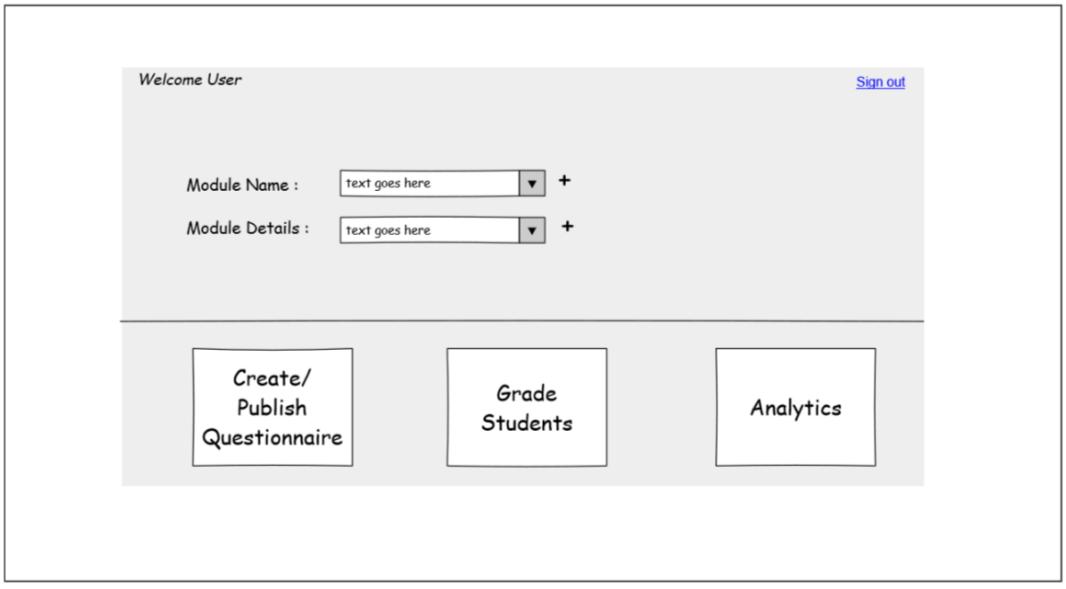


8.4 Annexure D – Mid Fidelity Iteration 3

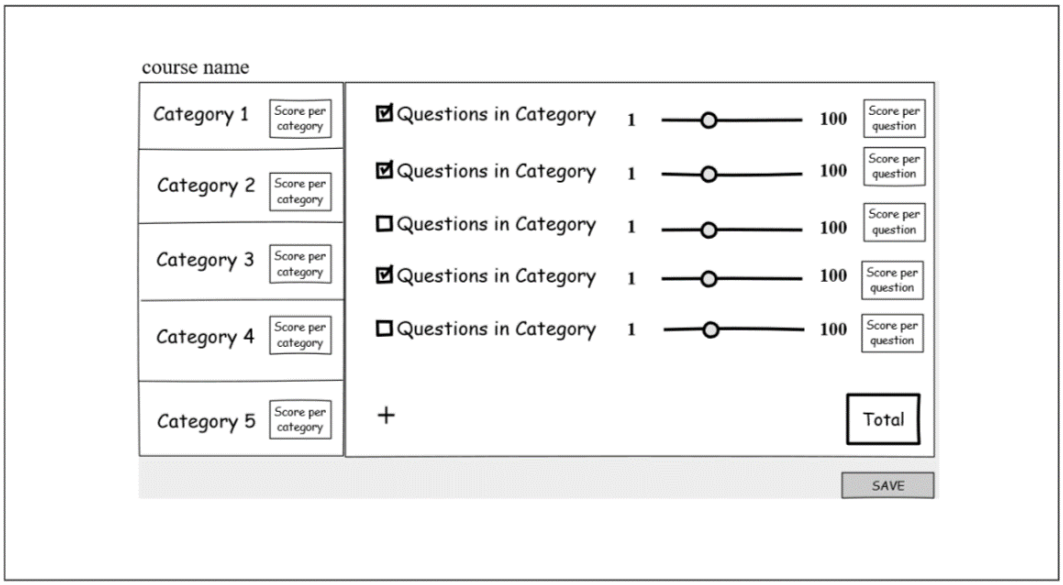
Login_



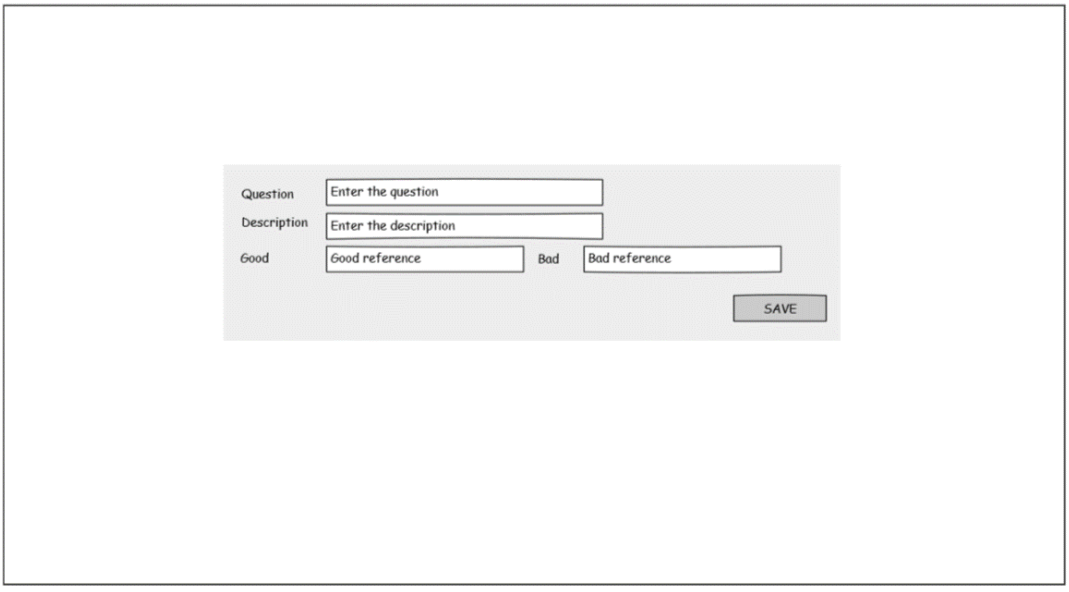
P_home



P_create_course

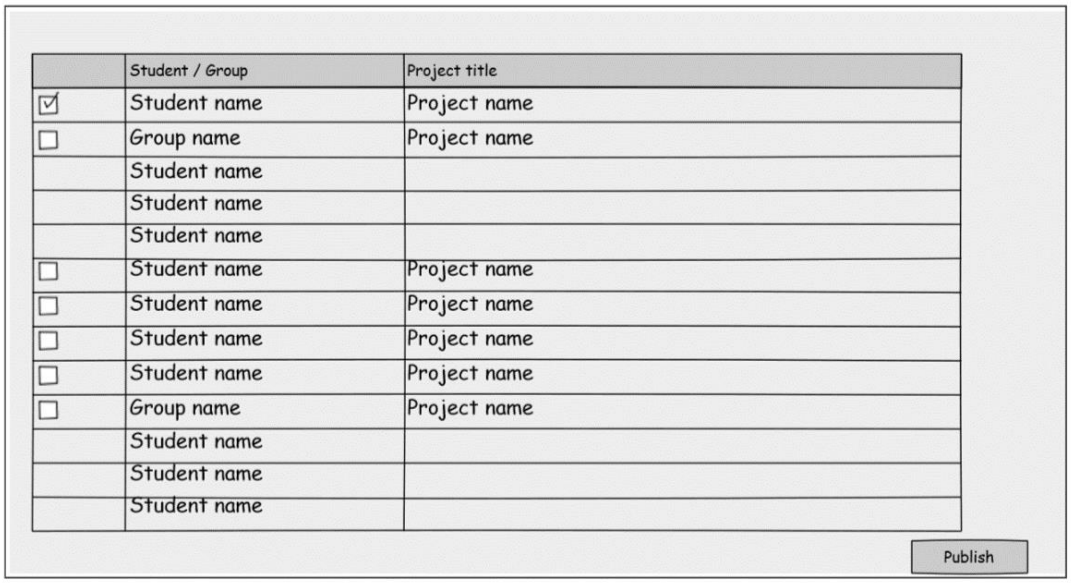


p_create_course_new_question_popup

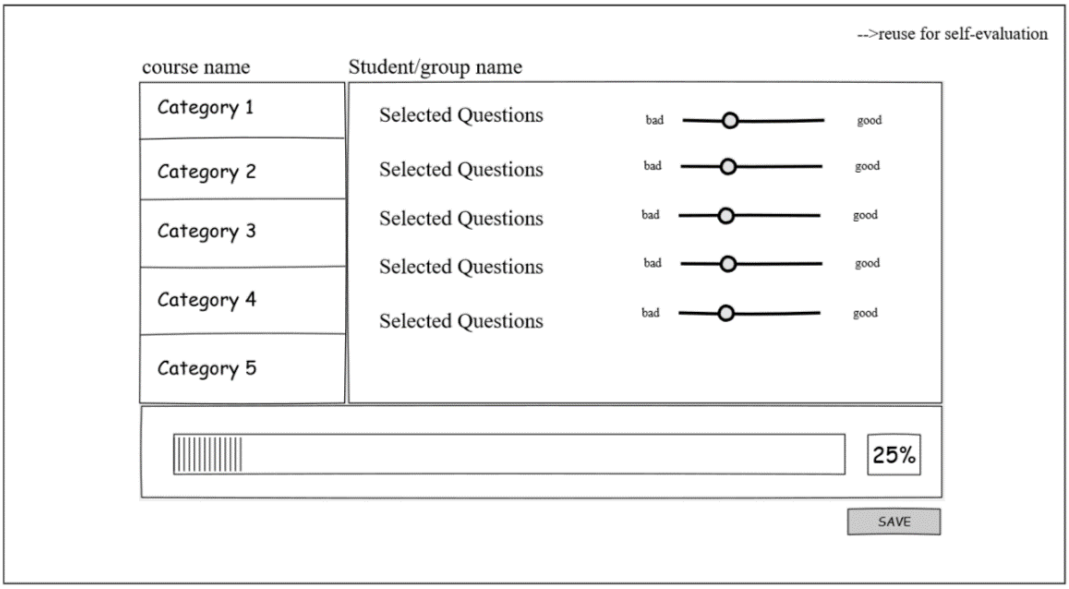


P_analytics

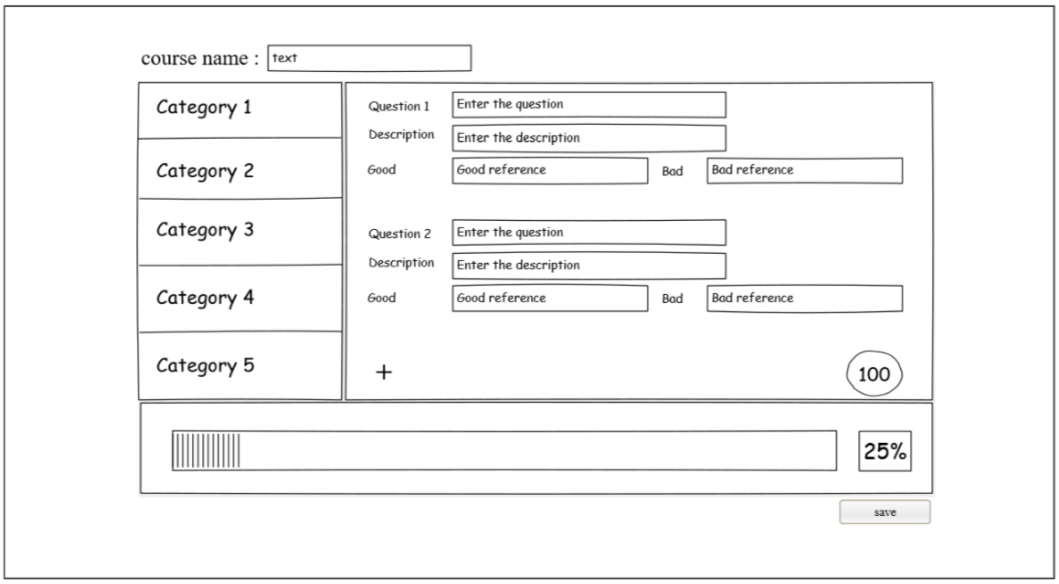
P_fill_student_list



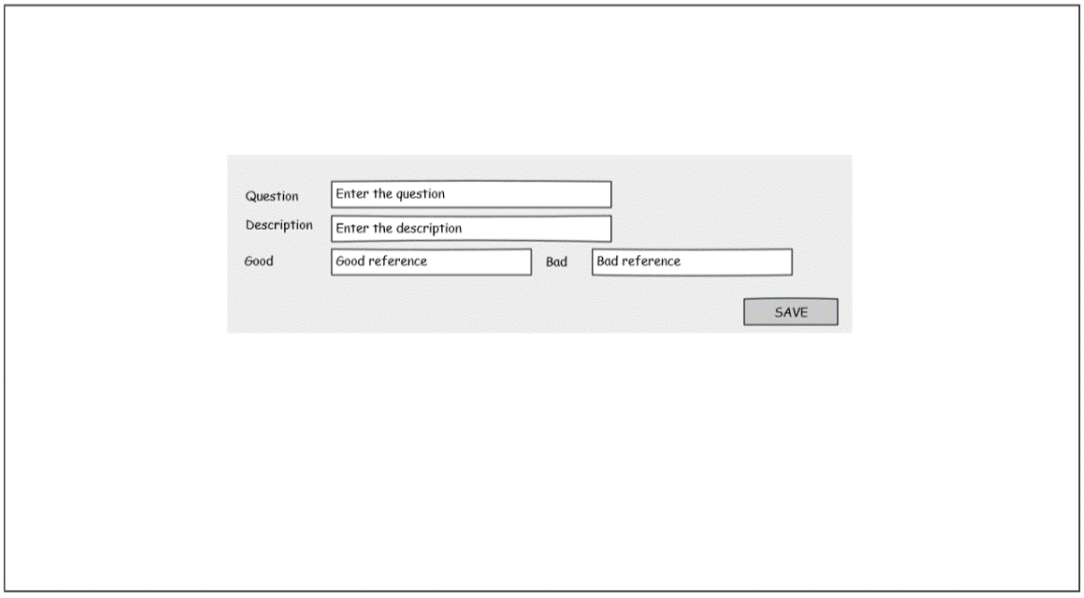
P_fill



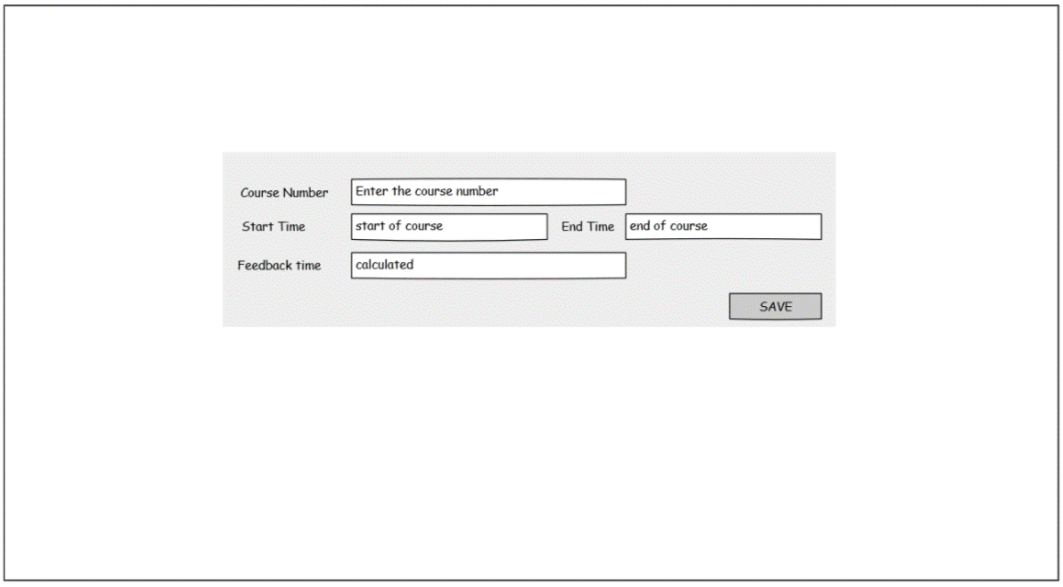
P_create_module



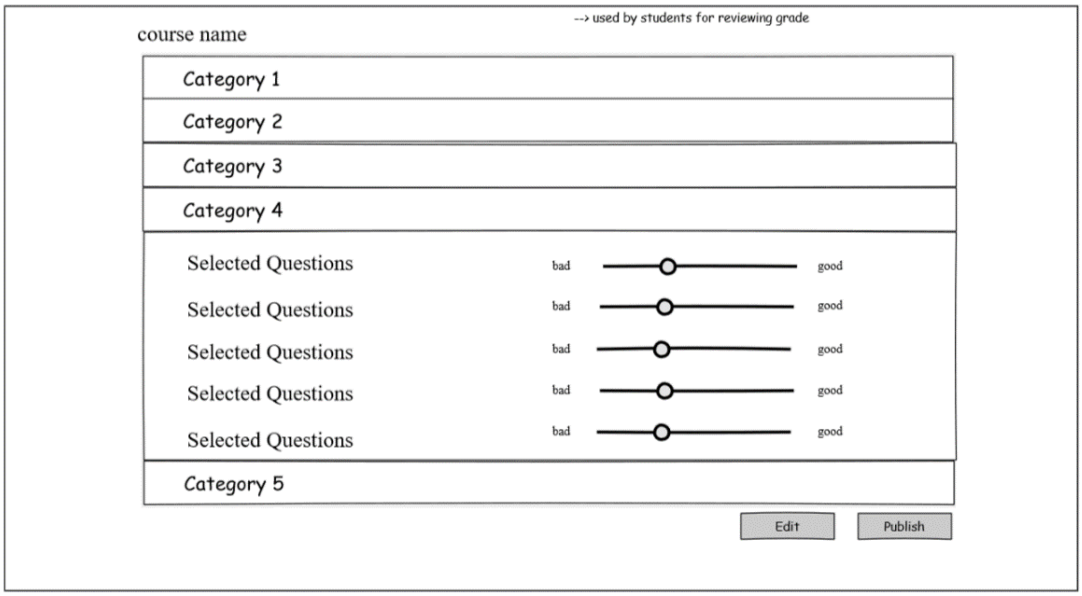
p_create_module_new_question_popup



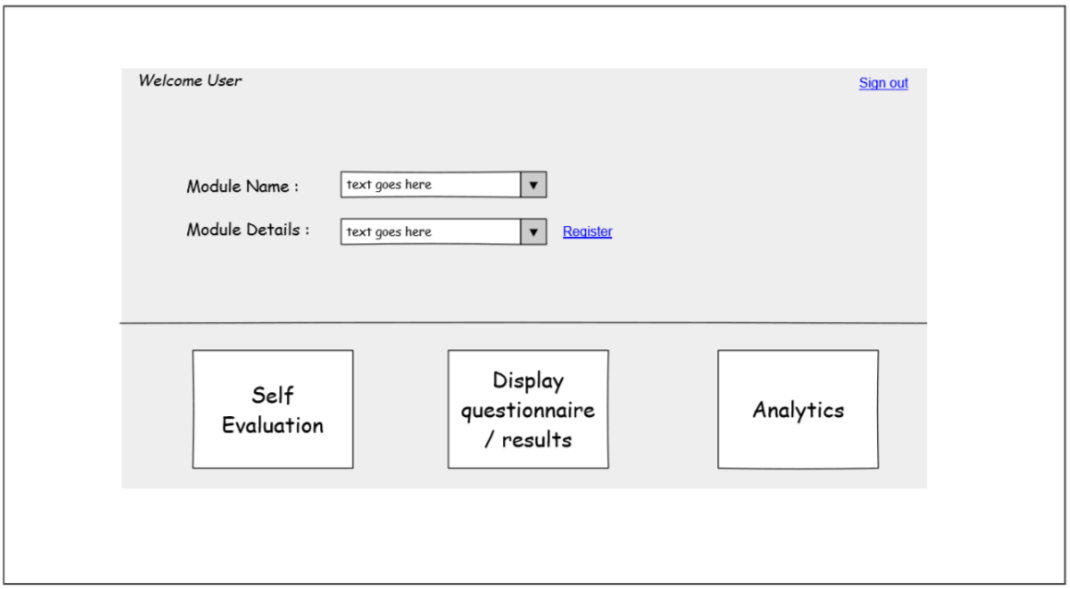
p_create_module_module_detail_popup



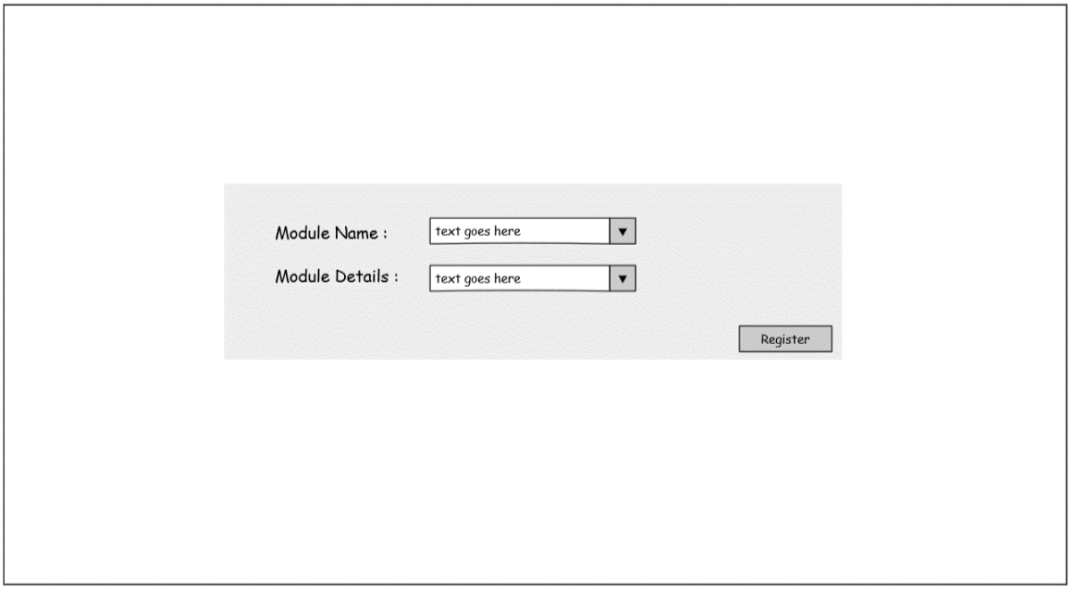
p_at_a_glance



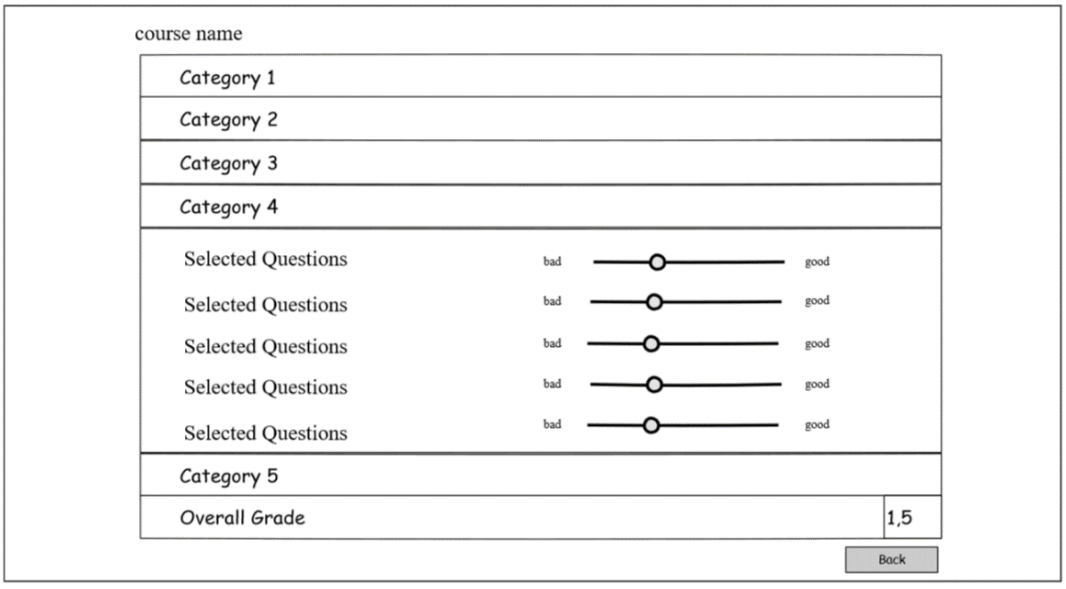
s_home



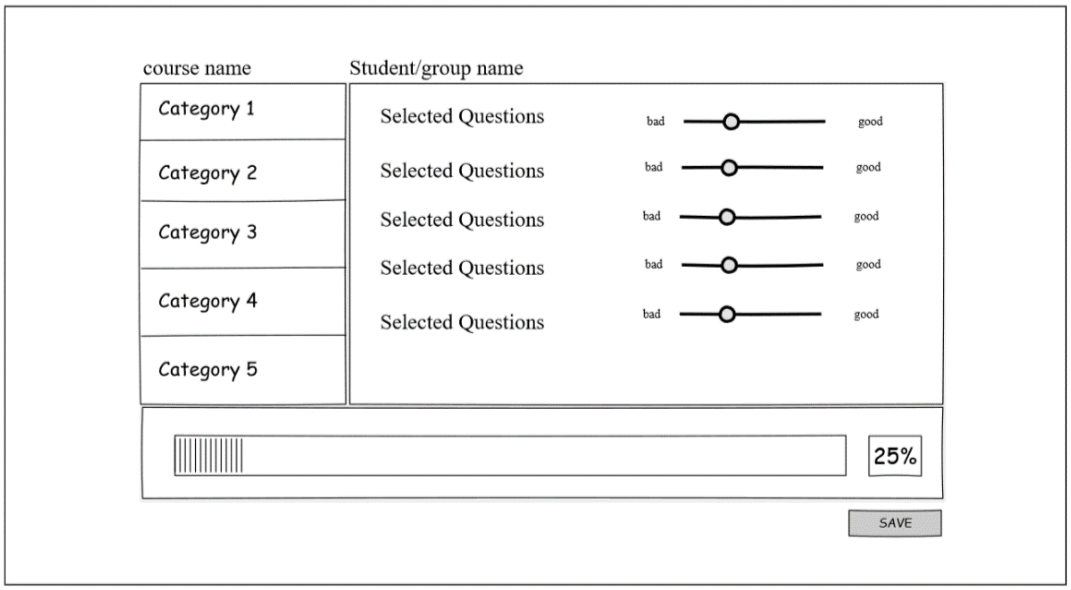
s_register_popup



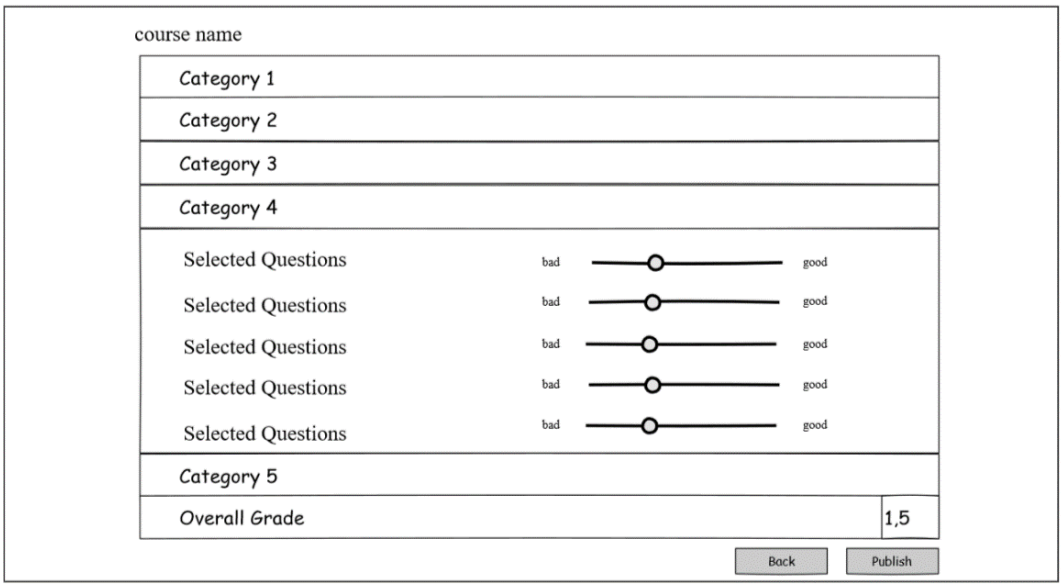
s_review



s_fill_self_eval



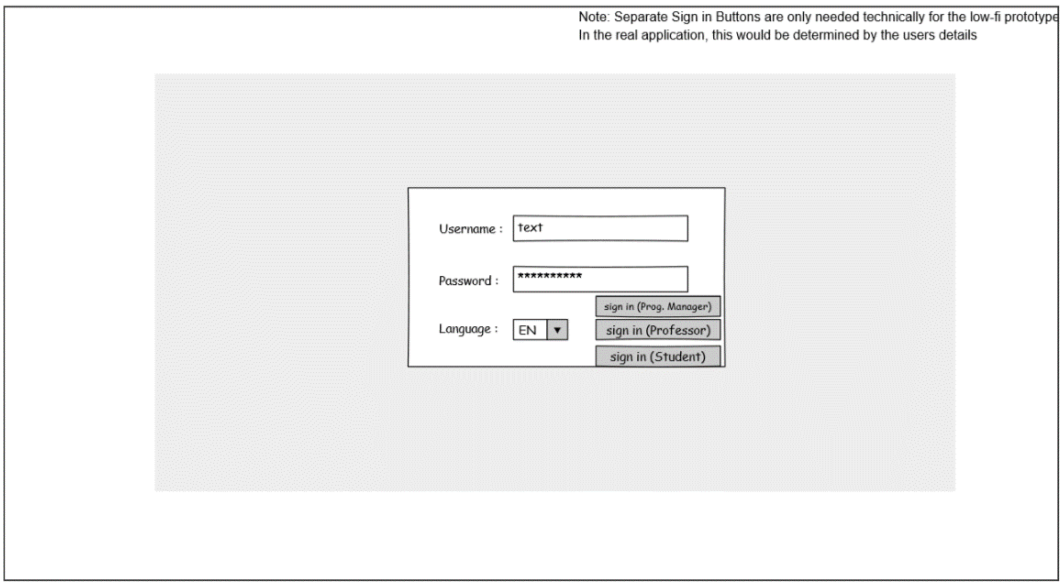
s_review_self_eval



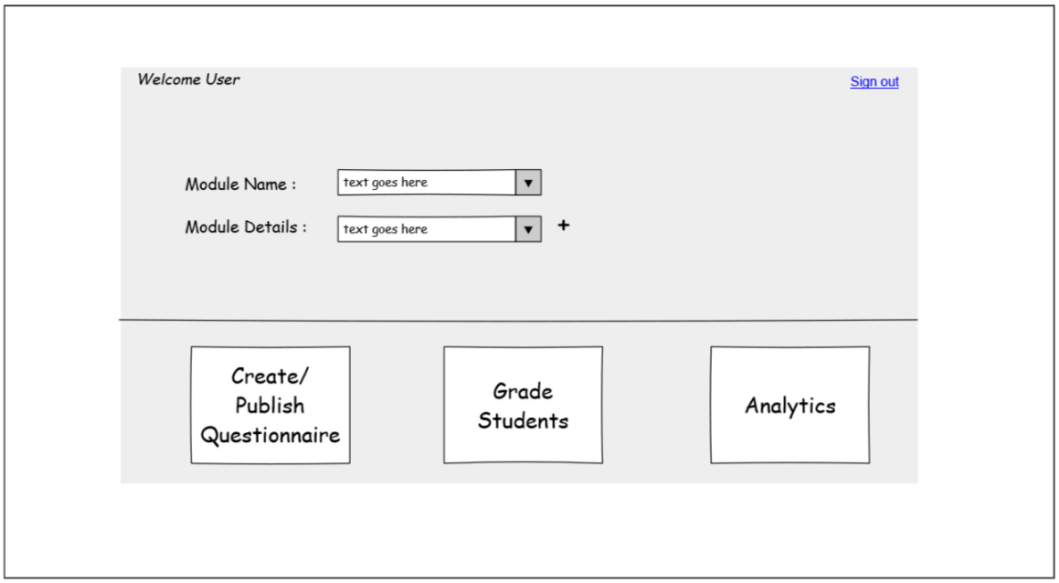
s_analytics

8.5 Annexure E – Mid Fidelity Iteration 4

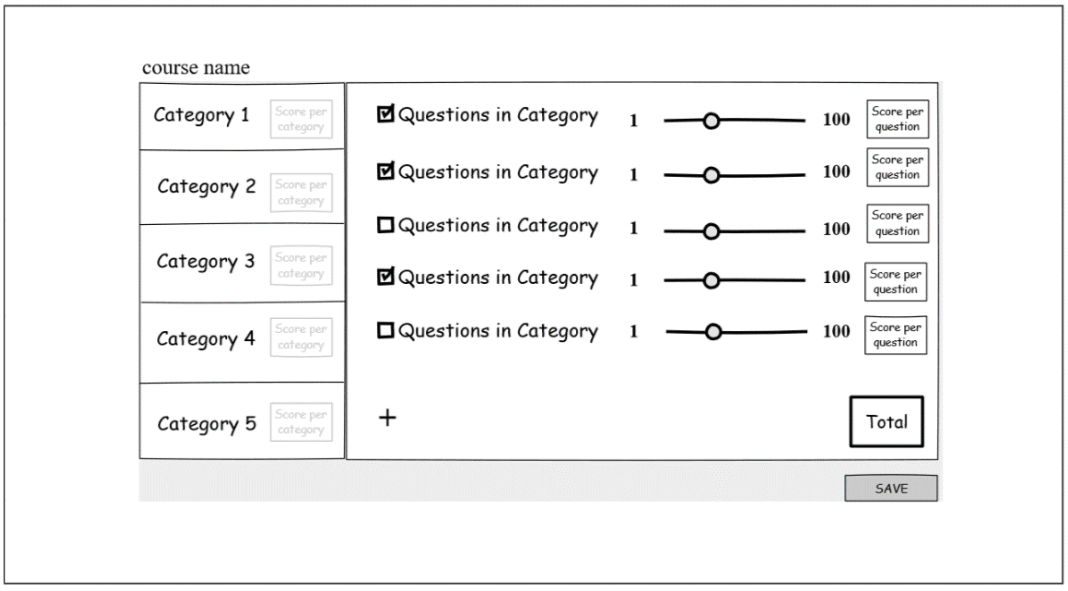
Login_



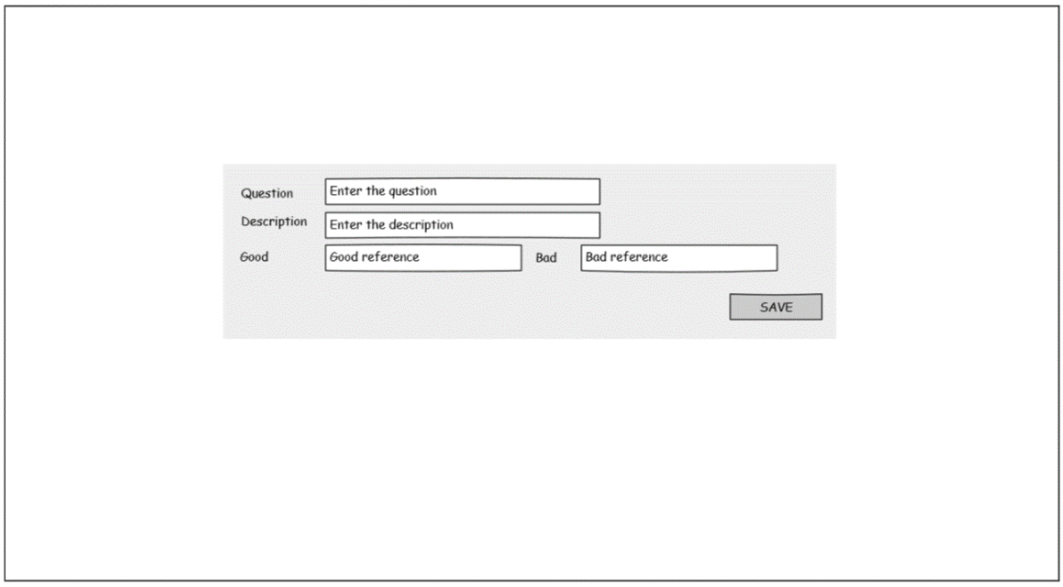
P_home



P_create_course

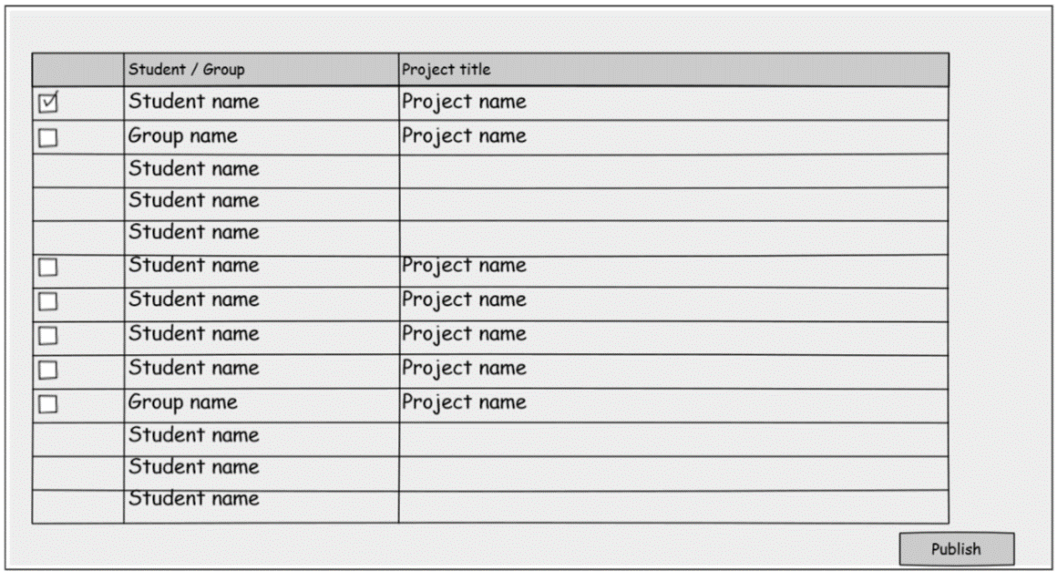


p_create_course_new_question_popup

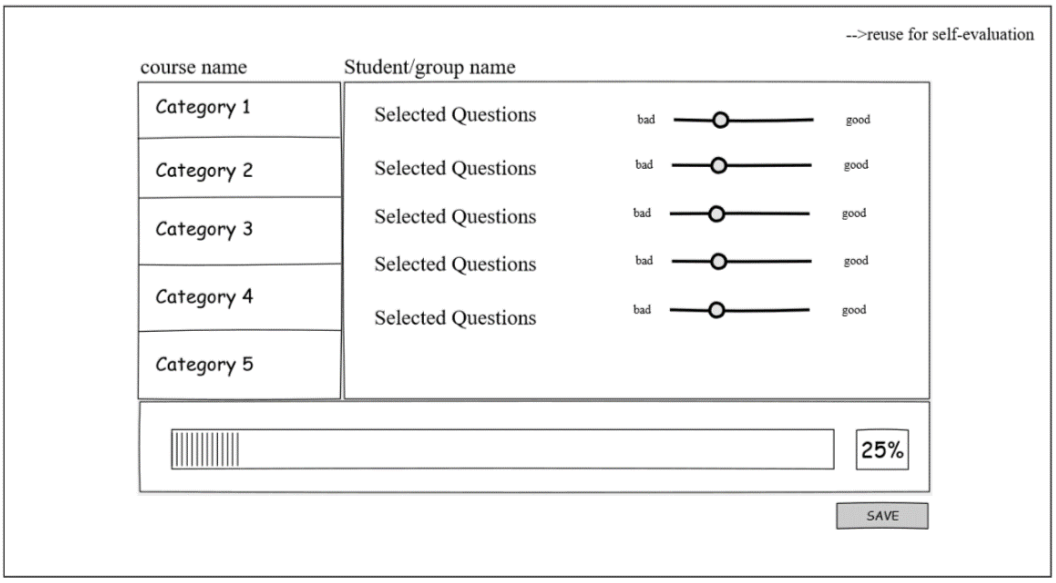


P_analytics

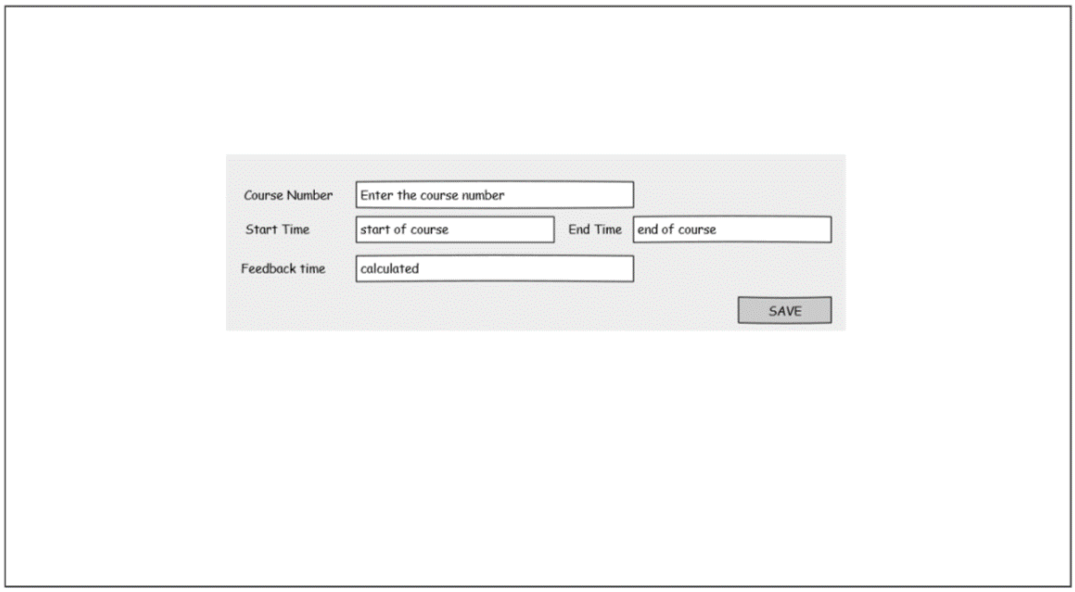
P_fill_student_list



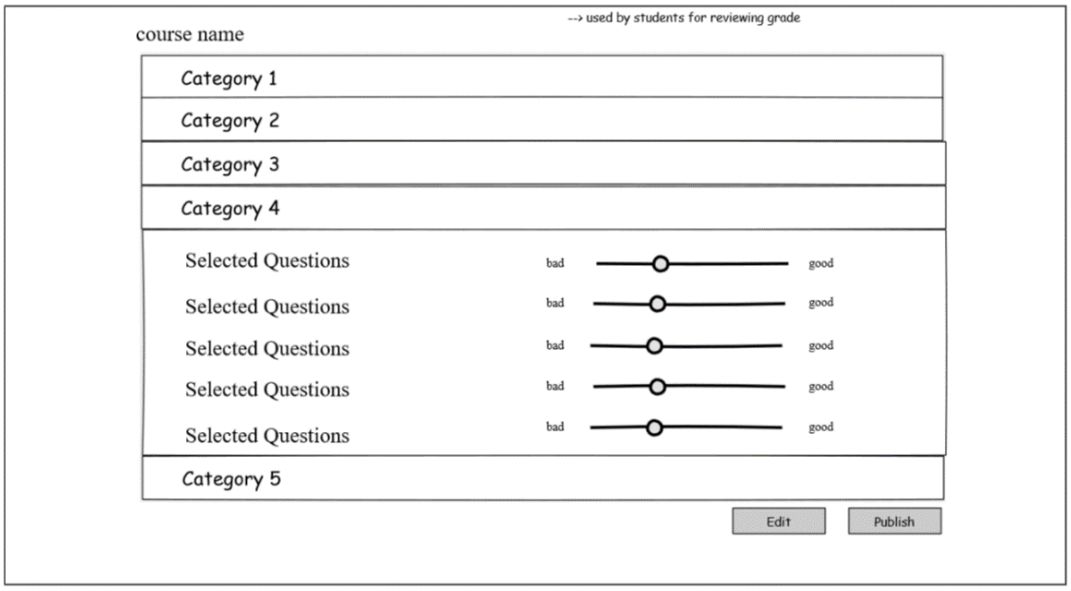
P_fill



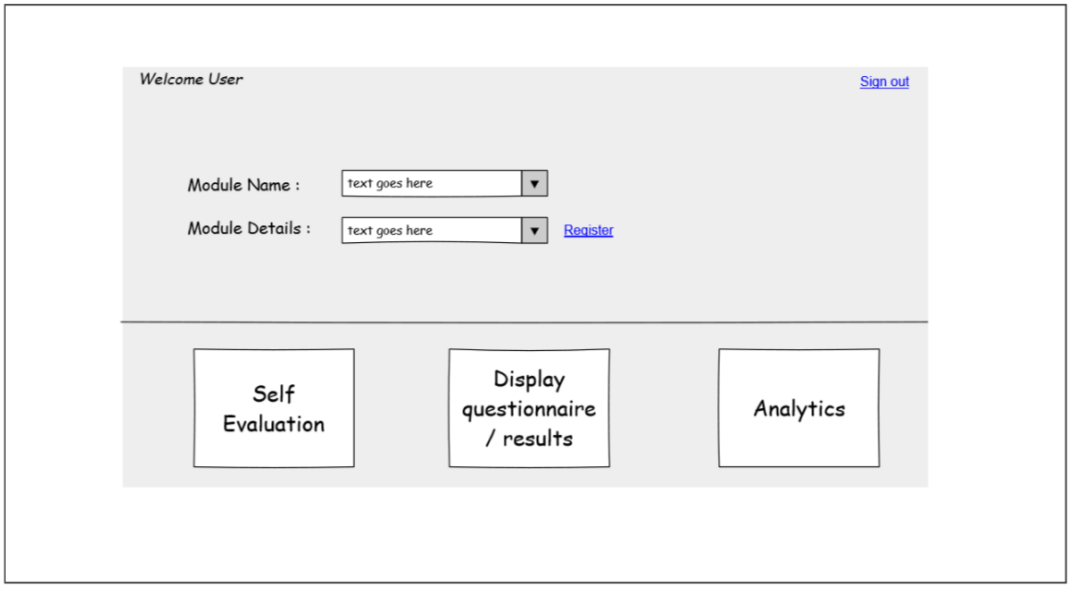
p_create_module_module_detail_popup



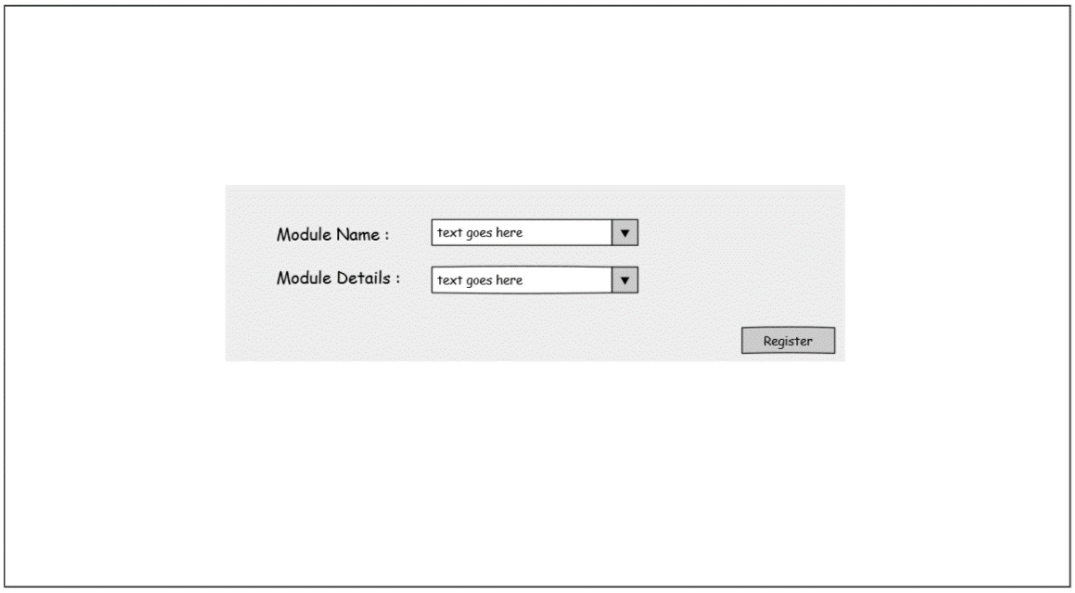
p_at_a_glance



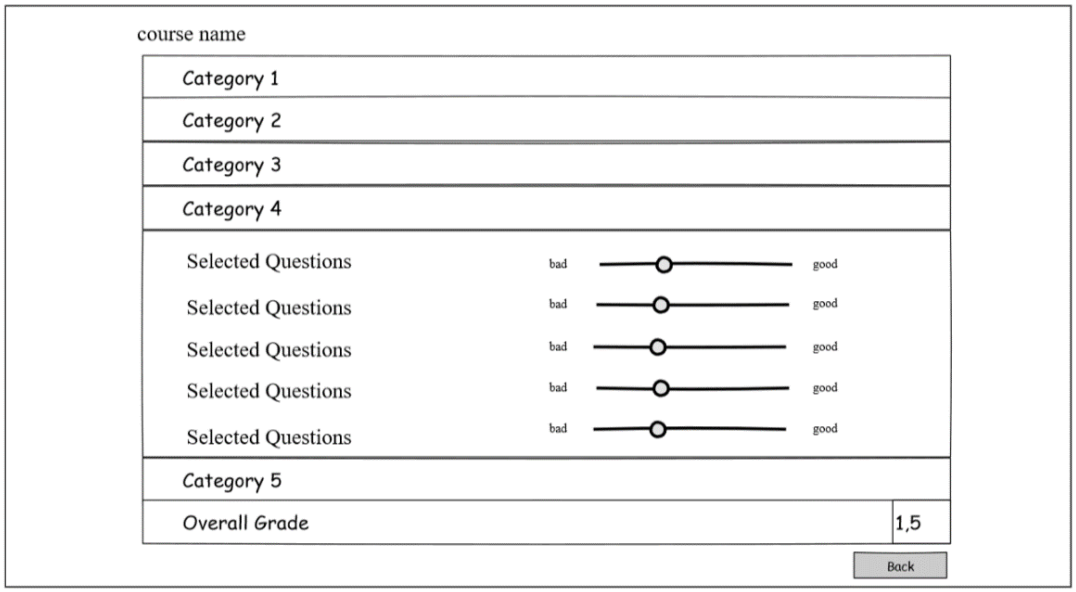
s_home



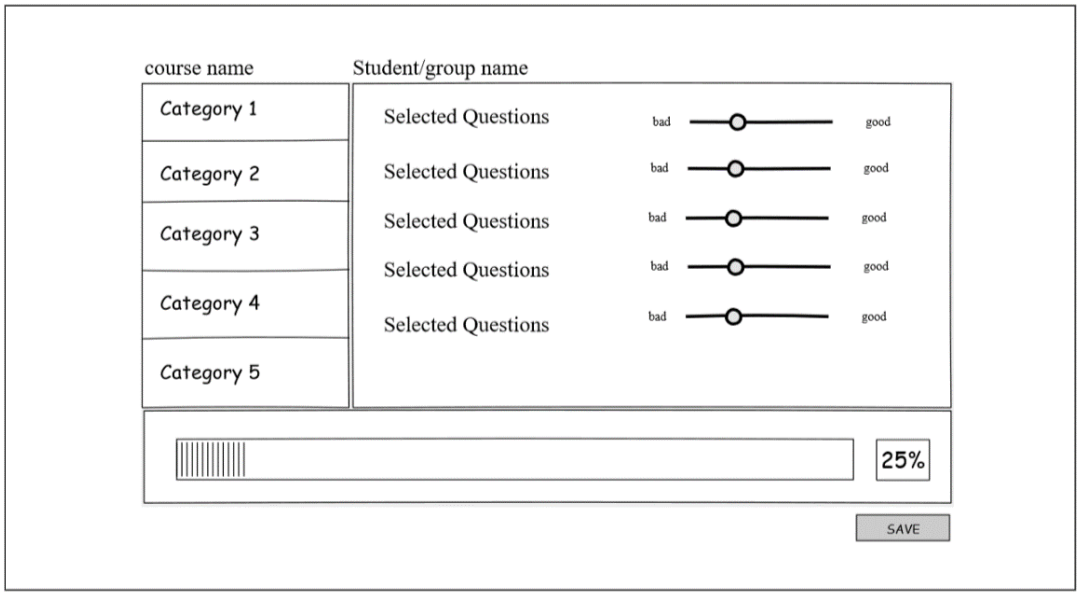
s_register_popup



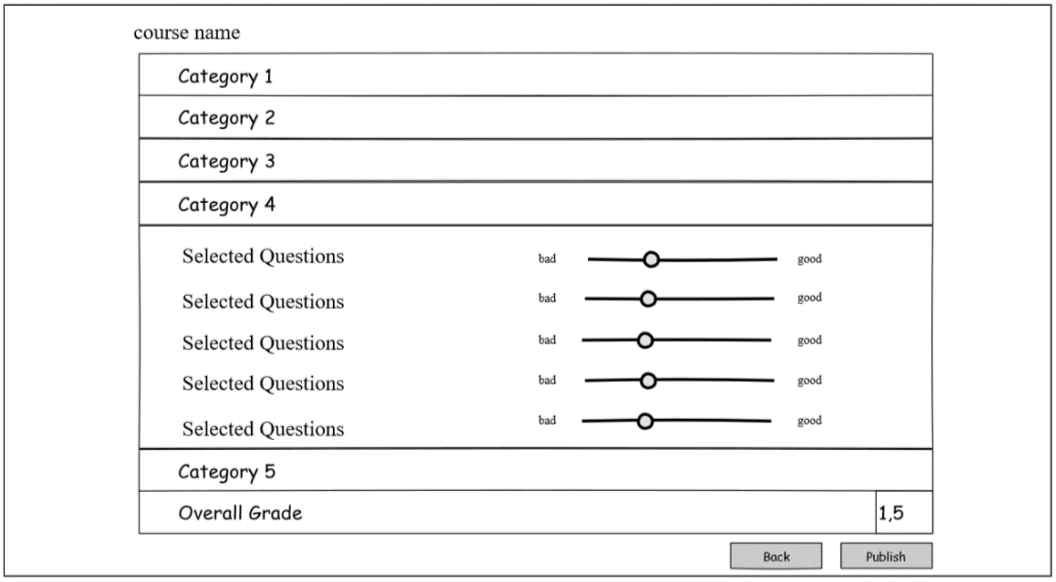
s_review



s_fill_self_eval



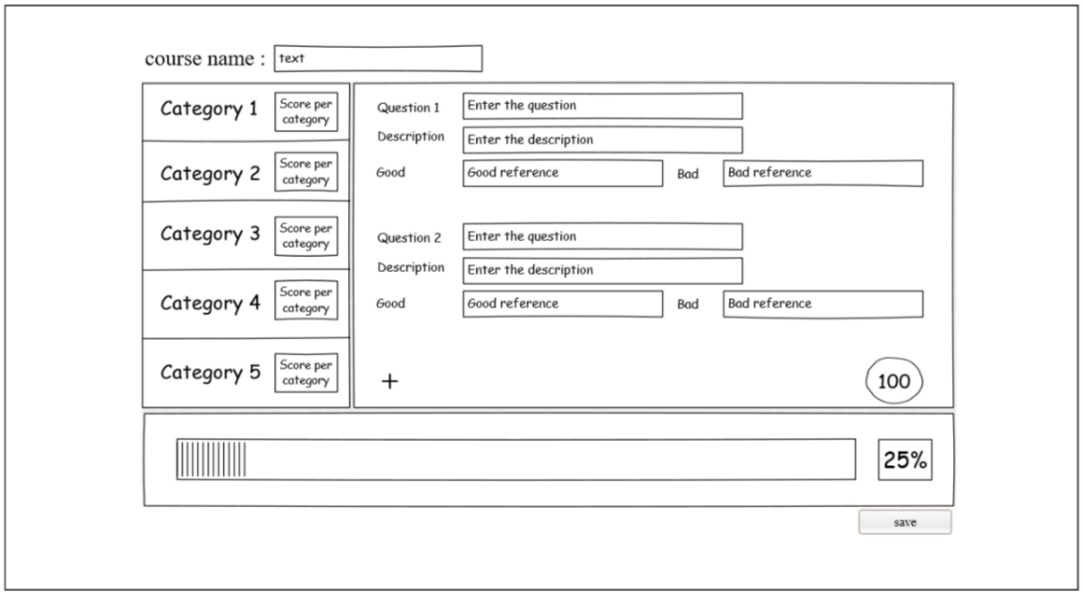
s_review_self_eval



s_analytics

PM_home

PM_create_module



8.6 Annexure F – Meeting Minutes Professor Kuehn

Meeting 11.04.2017

Montag, 10. April 2017
13:51

Meeting topic / purpose:

Interview with Professor Kühn regarding the persona "Professor"

Participants:

Martin Vogel
Professor Kühn

Agenda Points

1. Overview Presentation
2. Questions:
    a. Key responsibility of the professor

b. Main Goals when using the Application

c. In one Sentence:
   What does he hope for when he thinks about the Application

3. Key Features discussion:

   a. Questionnaire (Templates, weighting, etc.)

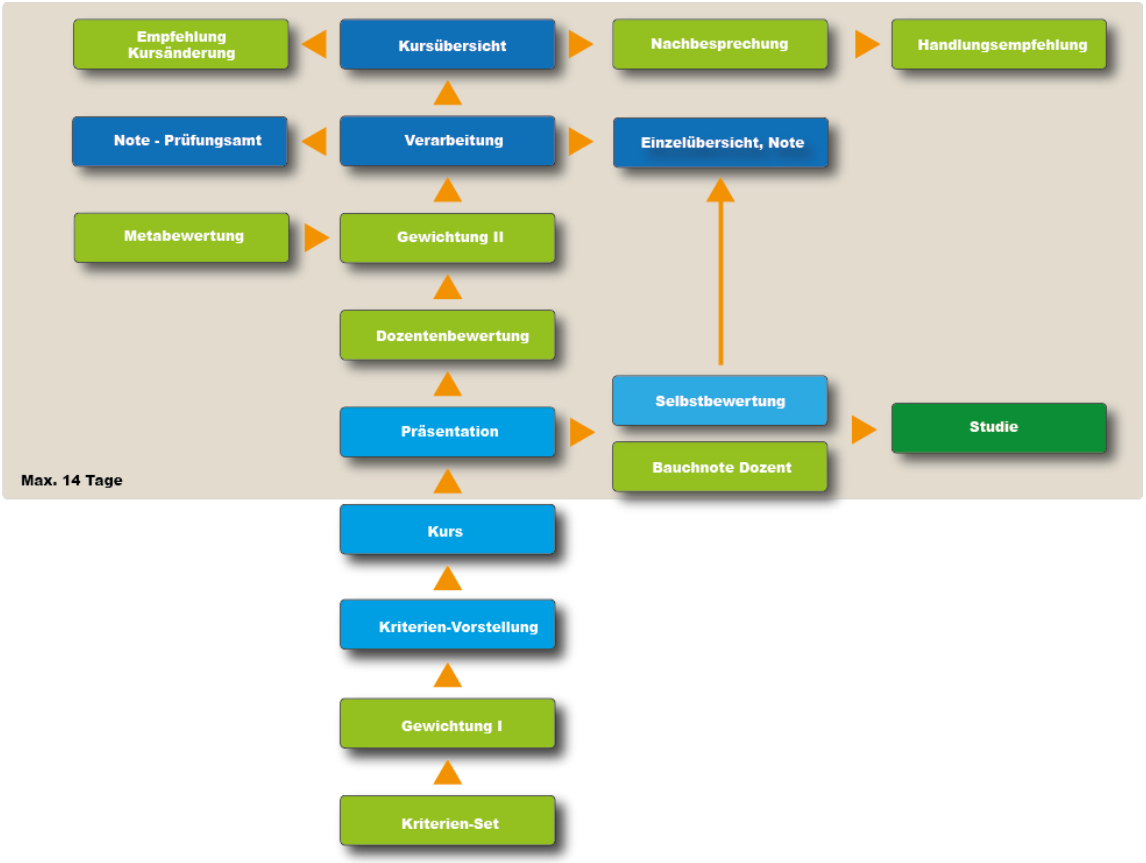
   b. Analytics (Statistics, Bell curves, etc.)

   c. Export / Import (File Format, Formatting)

4. In case we still have time: User Experience Map

Meeting minutes:

When Starting to discuss the topic, Professor Kühn immediately showed me a Flow-Chart as the result of their research project:



In addition he showed me an example questionnaire. This should be part of the paper that Christoph Hahn will send to us.
The questionnaire was split in 3 big groups with a defined weighting.
Each group had a set of questions to be answered.

As of the Flowchart (sorry for it being in German), the following applies:

1. The Professor defines a set of criteria to be used.

    1. Discussion point:

       How freely can he pick & choose from existing criteria?

       Possibility: Include some plausibility check with a message that the professor can just click away.

2. The Professor defines the weighting of the criteria

3. The criteria get published for the course and the course takes place.

4. After the Presentation, the following steps apply (this is just for statistical research purposes. NO influence on the grade):
    1. The Student needs to give a self-evaluation according to the defined criteria
    2. The Professor needs to give a grade according to his "gut-feeling"
5. The Professor fills the questionnaire
6. The Professor reviews the class performance and can possibly adjust the weightings slightly
    1. Discussion point: What exactly can the professor adjust?
       -->Only the detail weightings within the different blocks. Not the weighting between the blocks.
7. The questionnaire gets published. With this step, the system calculates the grades for each student.
8. Export to the official Tool.
9. Analytics on the grades (For Students, professors, deans, etc.)

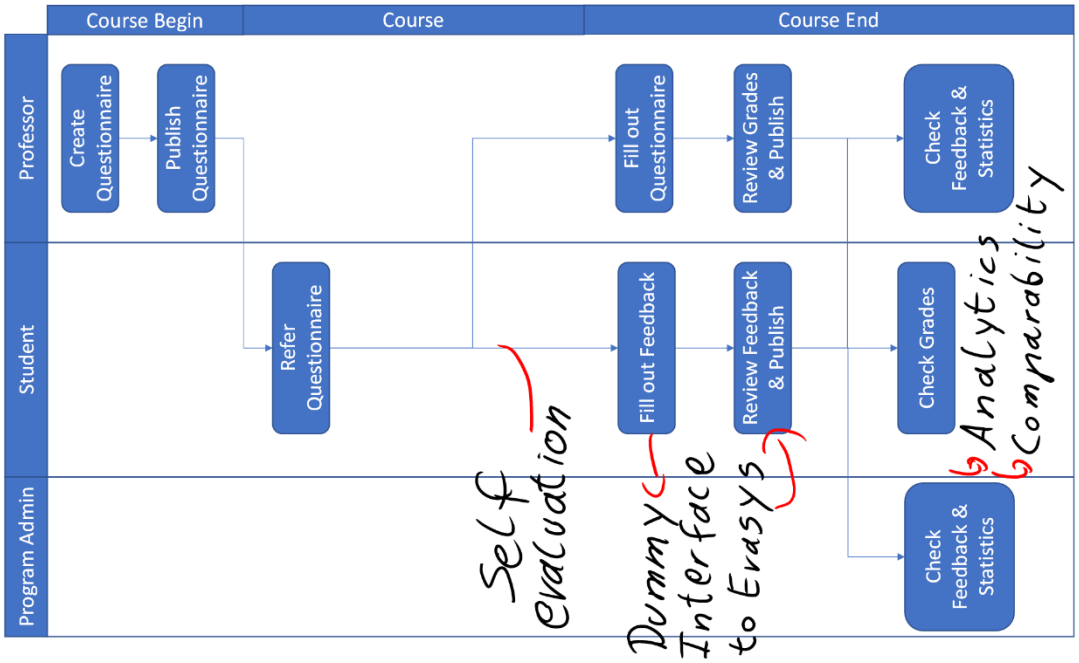
In Regards to the questions we defined Yesterday, he gave the following answers:

1. Main Goals when using the Application
    1. Ease of use, ease of access
    2. Multi-platform (incl. Paper-based, etc.)
    3. Mobile, etc.
    4. Publish Button
2. In 1 Sentence:
   What does he hope for when he thinks about the Application

    1. Transparency and efficiency of giving grades

3. Key Features discussion
    1. Questionnaire (Templates, weighting, etc.)
        a. Templates: Matrix, how freely professors can decide on criteria --> Legal considerations?
        b. Multi-Language Support
        c. Weighting
        d. Weighting adjustment (In details, not overall blocks)
        e. Self-evaluation of the student
        f. Group Grades are OK, if all Students of a group give their permission to it
4. Analytics (Statistics, Bell curves, etc.)
    1. Analytics for students (see experience map)
5. Export / Import (File Format, Formatting)
    1. Export --> for importing to the official "Prüfungsamt" --->Input Professor Kühn
    2. Printing
    3. Export as pdf
    4. Import from Scan
    5. Student Import (from campus net, excel, or similar)

I showed him the User Experience Map / Application flow. His comments have been that he would see the following adjustments:


Course Begin
Course
Course End
Professor
Student
Program Admin
Create Questionnaire
Publish Questionnaire
Refer Questionnaire
Fill out Questionnaire
Review Grades & Publish
Check Feedback & Statistics
Fill out Feedback
Review Feedback & Publish
Check Grades
Check Feedback & Statistics
Self evaluation
Dummy Interface to Evasys
Analytics
Comparability

8.7 Annexure G – Email Christoph Hahn

Von: Hahn, Christoph (SRH Hochschule Heidelberg)[SMTP:CHRISTOPH.HAHN@HOCHSCHULE-HEIDELBERG.DE]
Gesendet: Mittwoch, 3. Mai 2017 12:01:43
An: Vogel, Martin (SRH Hochschule Heidelberg Student)
Cc: Nadkarni, Navata (SRH Hochschule Heidelberg Student); Dogra, Aalisha (SRH Hochschule Heidelberg Student); Lee, Jo-Wei (SRH Hochschule Heidelberg Student)
Betreff: AW: Low-Fi Prototype - SAD Group 5 - Student Grading System
Diese Nachricht wurde automatisch von einer Regel weitergeleitet.

Hello,

thank you for submitting your prototype.

I think if we're talking about standardisized questionnaires it is mandatory, that not every professor can edit the single questions and questionnaires. So an additional role is needed here.

As professor you can pick questionnaires depending on the needed category. Maybe it will be needed that additional questions (pre-defined or not) can be added.

There will be questionnaires that target individual grades, others will target group grades. Background is group only grades are not possible due to legal aspects as we already discussed.

Most important step for you is to validate the low-fi prototype. This means to validate functional requirements like my input above but also validate user user acceptance by process usability tests etc. Most importantly for the low-fi prototype is the navigation and interaction of the single elements.

Regards,
C. Hahn

Von: Vogel, Martin (SRH Hochschule Heidelberg Student)
Gesendet: Dienstag, 2. Mai 2017 16:10:02
An: Hahn, Christoph (SRH Hochschule Heidelberg)
Cc: Nadkarni, Navata (SRH Hochschule Heidelberg Student); Dogra, Aalisha (SRH Hochschule Heidelberg Student); Lee, Jo-Wei (SRH Hochschule Heidelberg Student)
Betreff: Low-Fi Prototype - SAD Group 5 - Student Grading System

Dear Mr. Hahn,

attached you can find our Low-Fidelity Prototype for the Student Grading System. There is a .pdf file and the .zip file contains a html-version of it (with clickable buttons and links).

The Prototype was created using pencil: http://pencil.evolus.vn/
If you want us to, we can also send you the pencil file.

Could you give us Feedback from the following two perspectives?:
1. As the docent of our course.
2. As a future user of the Tool in the "Professor" Role.

Once we have Feedback from your side, we would also run this through Professor Kühn, who already offered us to give us some Feedback.

Please let us know in case we should have a quick meeting this week to go through it. We are flexible from a timings perspective.

Thank You,
Group 5