Microsoft Engage 2021

Scholarly Design Document

By
Narayan Pai
Final year undergraduate
B. Tech (Computer Science)
National Institute of Technology, Karnataka

Mentored by
Amarendra Singh
Engineering Manager
Microsoft

Project overview

Scholarly is a web application for teachers and students to interact online. It tries to inculcate different processes involved in daily activities in a school such as subject tests, lectures, etc.

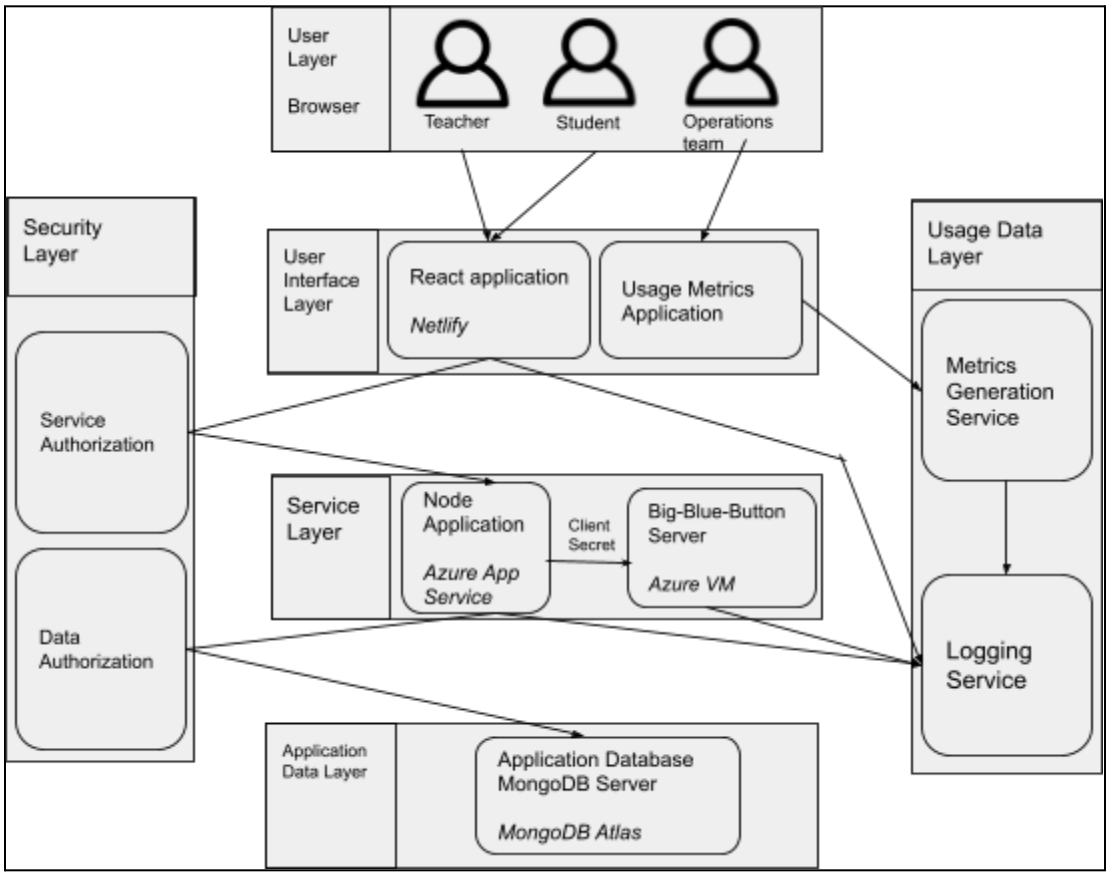
The website acts as a platform where teachers can create a course, publish it, organize and evaluate tests, conduct online classes, etc and students can enroll in any of the courses to stay updated about it.

System overview

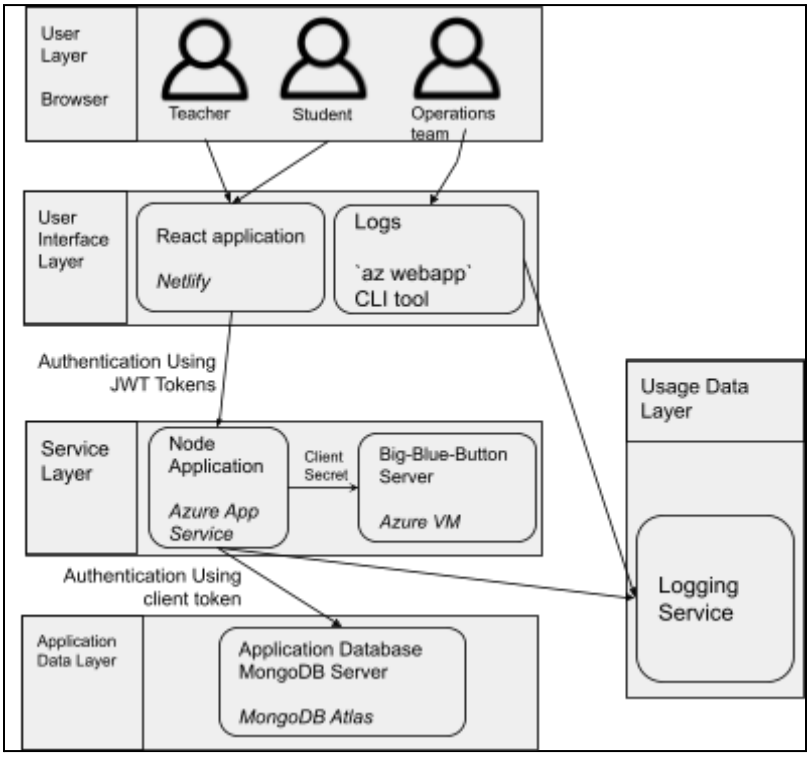
Technologies Used
By product: NodeJS, Express, MongoDB, React, Material-UI, BigBlueButton
For development purposes: Postman, ESLint

Platforms Used
Azure VM, Azure App Service, Netlify, MongoDB Atlas

Complete Application Architecture



Developed Application Architecture



The different services involved in the application are as follows.

1. **React Application:** A react application served from Netlify. We configure it to rebuild when any changes are done in the 'deploy' branch of the Github repository.
   Uses 'create-react-app' flavor of React and Material-UI components to build a single-page application. It connects to the Backend server through REST APIs for any operation.

2. **Node Application:** The backend application running an Express server. It is hosted in Azure App Service. We configure it to rebuild when any changes are done in the 'deploy' branch of the Github repository.
   The application has different REST APIs that can be used by the frontend. The APIs use JSON Web Tokens(JWT) based authentication for clients.
   It connects to the database server to retrieve any data. The data is accessed and altered with the help of the Mongoose ORM library.

3. **Big-Blue-Button:** A virtual machine hosted in Azure that has Big Blue Button installed and running on it. It exposes different APIs to manage meetings.

4. **Application Database:** MongoDB, a collection-based NoSQL database hosted in Atlas.

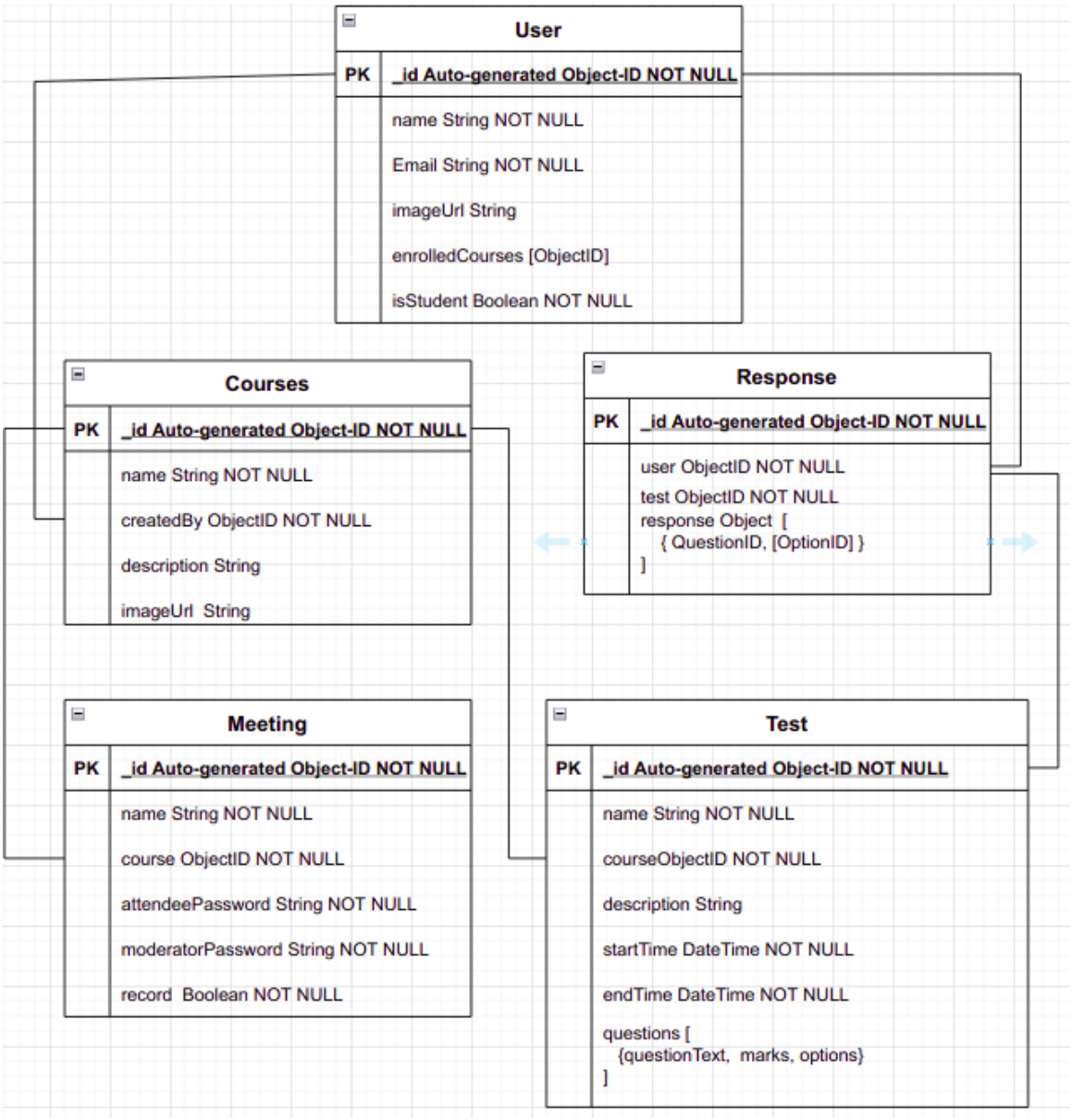
Miscellaneous details

1. The website has an adaptive user interface.
2. API testing will be done with the help of the Postman app and Postman CLI tool Newman.
3. These tests, including linting tests, will be run on each push to the Github repository.
4. Higher-level logging is currently done in the backend server itself. Due to the time constraint, these logs are currently just console-outputs that can be monitored with the help of the Azure CLI tool.

Data Overview

The figure below shows the Entity-Relationship diagram of the database designed. Though there are many relationships in the data design and using a SQL database would have been more scalable, a NoSQL database was chosen due to the time constraints. It establishes the following benefits which are crucial for a prototype app.

1. It is highly flexible since the table in itself doesn't enforce a schema.
2. The MERN(MongoDB-Express-React-Node) web-stack is very popular. Hence, a lot of documentation and references are available online.



The following are the miscellaneous details about these collections(tables).

1. User: The password stored is the hash of the password provided by the user. The *isStudent* property indicates if the student(*true)* or a teacher*(false)*. The *enrolledCourses* is an array of courses the user is enrolled in(valid only for a *student*).
2. Meeting: Contains two passwords: *attendeePassword* and the *moderatorPassword*. The former will be used while generating links for all the students while the latter will be used while generating the meet-link for the course instructor. The *record* value indicates if the meeting should be recorded or not. This value will be used while creating a meeting link with the configuration.
3. Test: The *questions* property holds an array nested object containing the details of each of the questions in the test.
4. The database uses nested objects to store properties when the data access of the parent collection and the property occurs simultaneously frequently (eg, *Questions* and *Test*). Else, different collections are used for the same (eg, *Meetings* and *Courses*).

Product design

There are two kinds of users in Scholarly; students and teachers. The teachers have permission to create courses and manage them whereas students can enroll in the courses and stay updated about them.

The app has two main features; Course Tests and Virtual Classes. The detailed information about these features is listed below.

Course Tests

1. Course tests are a way to assess the knowledge of the enrolled students of a course.
2. Scholarly allows teachers to create multiple tests with different topics.
3. The teachers must set a start time and end time for each test. This has the following advantages.
    a. A short span of time to attempt a course test will allow teachers to simulate an actual offline test. The students should be on time and submit within the allotted time. This will also allow that the students do not have room for any sort of malpractice.
    b. A large span of time (possibly days) to attempt a course will simulate an assignment. The students can learn the concepts involved, discuss the problems among peers and then come submit the assignment.
4. The tests currently support multiple-choice-multiple(or single)-answer questions. To be precise, each question can have any number of answers and any number of options(maximum five).

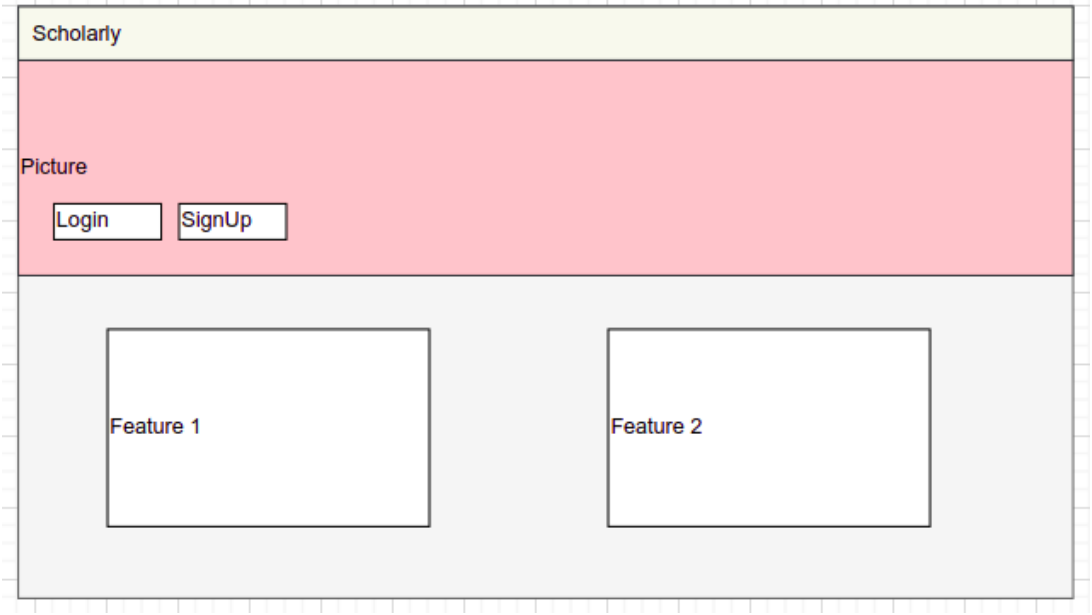
5. Each question carries marks and these marks are variable in nature and can be set by the teacher.
6. The web portal eases the process of creating/managing a test for teachers. The teachers can reorder the questions, delete a question, include images in questions and options, clone a question as a template for the next question, and so on.
7. The teachers also get to see all the responses of the students in real-time. Scholarly also assesses the responses and automatically grades each of the students.
8. The students can start the test any time after the start time but have to submit on or before time. If a student does not submit the test while attempting it, Scholarly automatically submits it when the time is up.
9. Once attempted, a student can go back to the test any time and review his/her performance. This includes all the questions and their responses along with the expected responses.

Virtual Classes

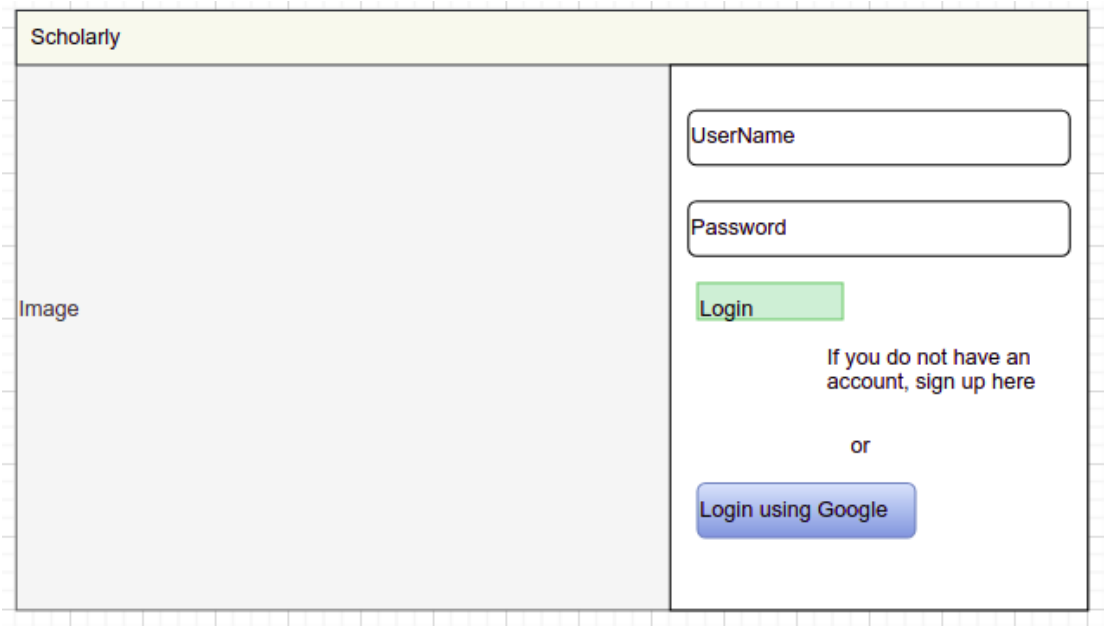
1. A teacher can start a class under the created courses at any time.
2. At the time of creating a class, the teacher also gets an option to record the meeting.
3. Once created, the teacher will be redirected to a BigBlueButton meeting where he/she is set as a moderator.
4. All the students enrolled in the course will be able to see the meet link on the course page and they should be able to join the meet.
5. If the meet is no more active and the meeting was recorded, the student can watch the recording of the lecture at any time.

The following are the low-level designs of each page that were planned in the early stages.

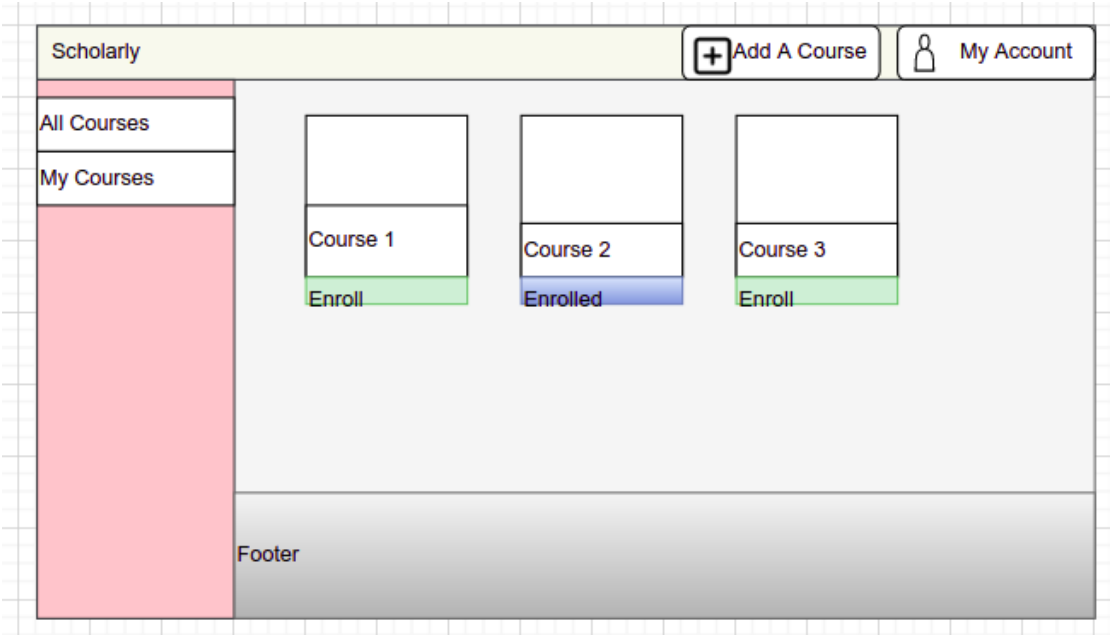
1. Landing Page.



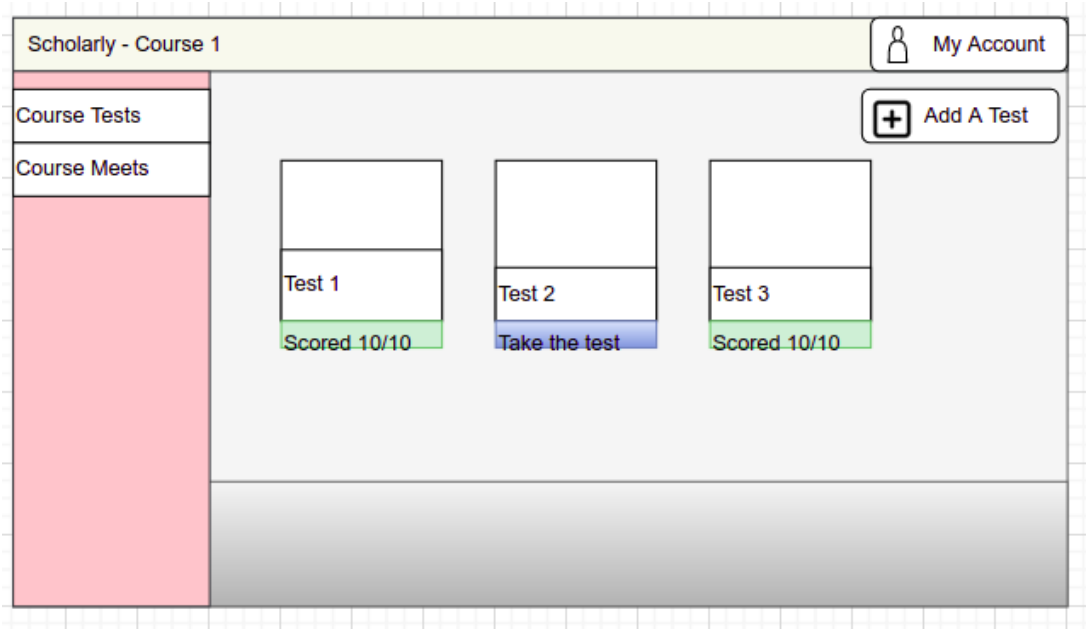
2. Log in/Sign up Page.



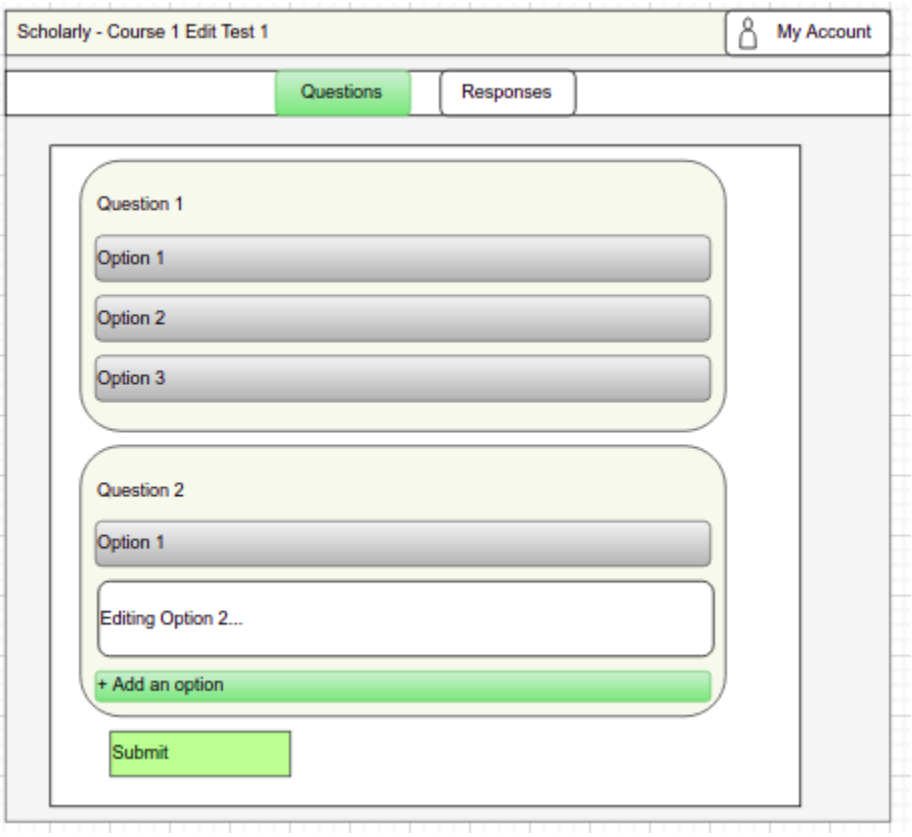
3. Dashboard
   Here, the ‘Add a Course’ button is only visible to a user of type ‘teacher’.



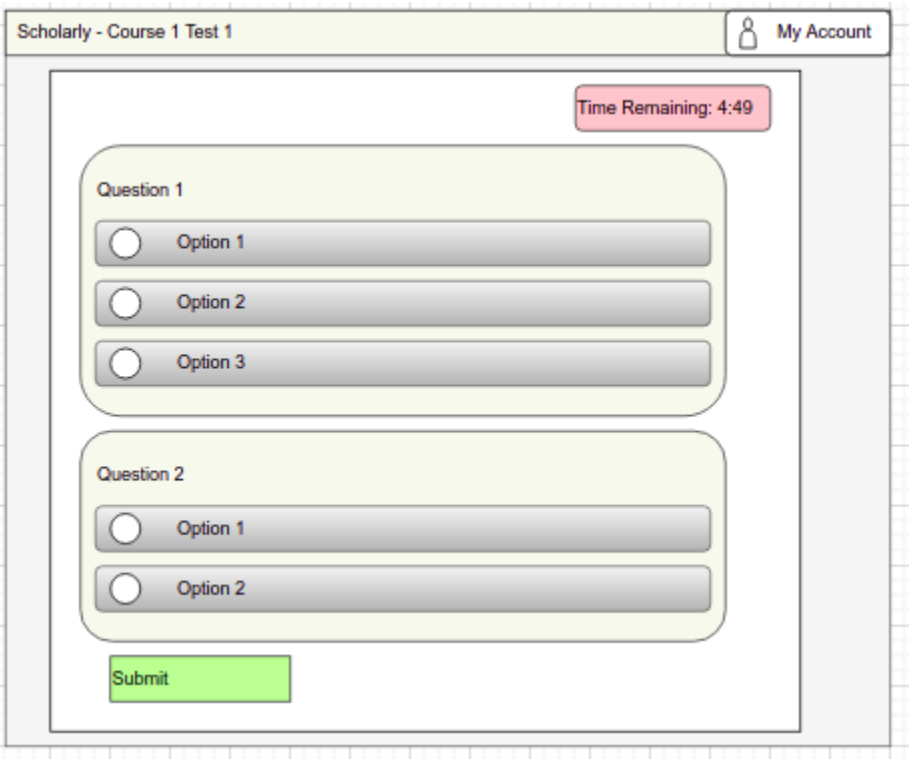
4. Course view
   Here the ‘Add a Test’ is only visible to the instructor of the course.



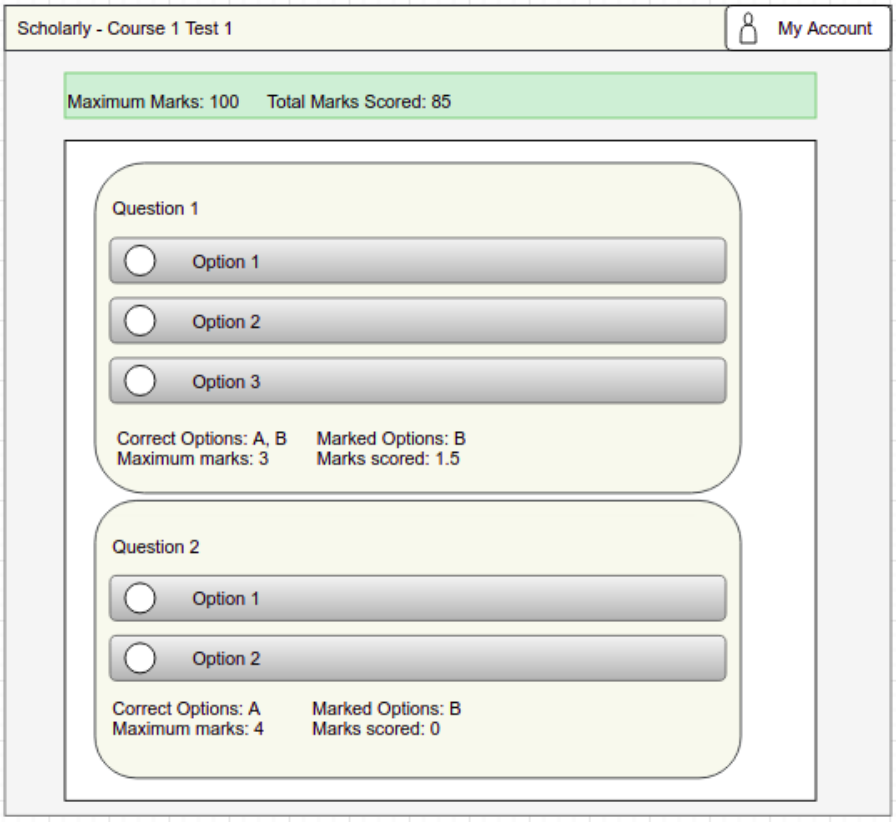
5. View for creating a test



6. View for attempting a test



7. Test view after attempting the test



8. Test summary view for the instructor.