Bilkent University

Department of Computer Engineering

Senior Design Project

Tutoryum

Project Analysis and Requirements Report

Barış Ogün Yörük

Halil Özgür Demir

Mustafa Çağrı Durgut

Oğuzhan Özçelik

Yusuf Miraç Uyar

Academic Supervisor: Can Alkan

Industry Expert: Mehmet Gök

Jury Members: Erhan Dolak, Tağmaç Topal

Project Analysis and Requirements Report
Oct 17, 2022
This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2

1. Introduction

Have you ever thought that the most efficient way to achieve anything is to communicate directly with someone who has accomplished it before? Well, We did. Tutoryum is a platform where you can arrange one-on-one video interviews with people who have excelled in an exam you want to take.

An examination is an inevitable part of the education system, and whatever happens to the education system, it will require some form of testing. One of the biggest problems with traditional exam preparation methods is that the direct applicability of the education given in such methods to exams needs to be tested. In the orthodox method, more than one instructor specializing in only one subject successfully gives detailed information about their field, and they help students to have a deep understanding of these fields. However, what students desire is to familiarize themselves with these fields just to let them pass the exams. However, Tutoryum's instructors who have excelled in exams and successfully registered on Tutoryum have both expertise in the fields and a deeper understanding of the exam and its methods than any other traditional instructor.

So, Einstein once said that if you can't explain it simply, you haven't understood it deeply. Like this, Tutoryum says that if you understand it deeply enough to excel in an exam, you can explain it more simply than any other instructor. Suppose you think so and want to be included in the system as an instructor or student. In that case, Tutoryum will soon be your indispensable platform with its fast and scalable servers and great user experience!

2. Current System

One of the most significant shortcomings of today's education system is that people's motivations to continue their education need to match the purpose of education, while the education given is trying to make these people intellectual in a particular subject. Many people work to pass exams or get a degree. Traditional teaching aid methods also copy this method of this education system. Because of this difference in motivation, many students find their education process boring and unproductive.

Some products like Zoom [6] and Google Meet [5] are video conferencing applications. Also, applications like Udemy [7] provide educational courses via video streaming. However, both of those applications give a private lecture and schedule-oriented system.

Our application differs from Zoom and Google Meet in two main aspects. First, our application is more oriented toward lecture giving. Meaning it will include features like an interactive whiteboard, slide sharing, optimized whiteboard recording and storage, and automated slide creation from the whiteboard at the end of the session.

Secondly, we are planning to conduct video streams as a peer for the sake of scalability. We might use an API for video streaming purposes and try to add special functions for educational purposes, but our group intends not to do that.

Compared with applications like Udemy, our application differs as it focuses more on private lecturing and automated verification of tutors for their profession. Our application will provide live streaming and live interaction with the lecturer and the student. Most of the features will be focused on this live interaction fundamental.

3. Proposed System

3.1 Overview

Tutoryum is a platform where users can arrange one-on-one video conferences with people who have excelled in an area and have this achievement registered in the system. Tutoryum provides a platform for students and tutors to find what they want quickly. Tutors who have registered their achievements in the Tutoryum system share their availability schedule on their page. Anyone interested in having a one-on-one video interview with that instructor books an available time slot, and a meeting settles. Tutoryum provides an all-in-one experience where users don't have to register to any other platform to join the meeting. All of the video meetings are conducted inside the platform.

Tutoryum also provides easy-to-use class materials to improve education. Tutors can create virtual 'courses by uploading their materials to the system; then, they can teach their classes. Built-in whiteboard and slideshow facilities will make tutors' jobs easier.

Tutoryum is a web application where we will need a server, a P2P video conferencing infrastructure, and user-friendly interfaces. We will use WebRTC technology to set up the video conferencing and interactive whiteboard features. Relational databases such as MySQL will be used to store data. Modern cache techniques will also be utilized with tools similar to Redis. A RESTful API or a GraphQL API can communicate the back end with the front end. To provide an excellent and smooth user experience, Tutoryum will be built with progressive web application principles in mind. That is, we will use service workers, manifests, and other web-platform features in combination with progressive enhancement to give users an experience on par with native apps.

3.2 Functional Requirements

- Users can register to the system.
- Users can send documents to the system to tutor a specific subject.
- The system will automate the verification of the documents.
- Students can see the expertise of an instructor from their profile.
- Students can reserve times from instructors.
- Students and tutors can meet at peer-to-peer video conferences.
- Users can use educational tools interactively.
- Educational tools such as interactive whiteboards should be saved and used later by the tutor.
- An automated system will create a slide show from the changing status of the whiteboard throughout the session.
- Tutors can make quizzes after classes using the materials they use in lessons.
- Students can pay for the courses.
- Tutors can get their pay from the payment.
- Tutors can create courses.
- Users can preview a limited part of a course material
- Tutors can add course materials to their courses.

3.3 Non-Functional Requirements

- **Security:** Tutors' documents should be protected. Payments should also be safe.
- **Scalability:** The system should be able to handle 500 concurrent meetings.
- **Extensibility:** The system should be suitable for new features to be added. Document automation will be added for each subject type separately. It should support the addition of new document types for new professions.
- **Usability:** The application should be simple enough to access broad communities.

3.4 Pseudo Requirements

3.4.1 Implementation Constraints

- The project will be implemented as a website on the Internet.
- Git will be used for version control of the codebase. [1]
- GitHub will be used for cooperatively working on the project’s repository. [2]
- CircleCI will be used for CI/CD pipeline so that codebase will be controlled and tested with each commit. [3]
- Asana will be used for task management. [4]
- Google Meet will be used for the team’s video meetings. [5]
- The project will use a third-party library to implement video conferencing logic.

3.4.2 Economic Constraints

- Since many users will use the application, multiple servers will be needed to scale the application. For development purposes, a free or relatively cheap server will be enough. In the case of production, using one of the Cloud Providers might be needed. In that case, the price will be related to the number of users.
- For all the development tools, free versions will be used. In case of an unexpected price, another new tool will be used instead. (i.e., development tools [Asana, CircleCI, Google Meet] mentioned in the implementation constraints are subject to change.)
- We will buy a domain name to publish our project.

3.4.3 Ethical Constraints

- Tutors' information should be viewed without leaking too much personal information.
- Documents of tutors are confidential, so they should be securely stored.
- Contents of the meetings should be encrypted so that adversaries cannot watch them.
- The payment system should be safe, so we don't have a leaking problem.

3.4.4 Sustainability Constraints

- The application will be reviewed at regular intervals once it finishes. The purpose of it is to keep technologies and tools used in the project updated, enhance performance, and fix unwanted behaviors and bugs.
- New features regarding education tools might be added after the project finishes.
- In case of a surprising number of users, the project will have a scalability logic so that the application will keep running without intervention.

3.4.5 Social Constraints

- The project's initial implementation will be in Turkish since our target market is Turkey. We will first consider the exams of Turkey, such as YKS, ALES, and KPSS. The addition of other countries will be done after the project.
- There should be a request & complaint email for solving disputes between students and teachers. For example, teachers can be different people from how they introduced them to the system.

3.4.6 Technological Constraints

- Users will need a microphone and a camera.
- Users need a sufficient internet connection for the best experience since it is necessary for both interactive real-time education tools and video conferences.

3.4.7 Time Constraints

- All the basic functionalities of the project must be finished before the deadline, such as login & register logic of students and teachers, verifying exam results, viewing teachers, setting up & canceling meetings, and uploading documents.
- At least video conferencing and interactive whiteboard features must be finished before the deadline.

3.4.8 Ethical Constraints

- Content publication of tutors should be checked to verify the published content belongs to them.
- Users should be allowed to report any inappropriate content published by the tutors.
- The payment system included in the application must follow each online payment-related law of the countries in which the payment system is implemented.
- Users should be informed and ask for permission for each user's collected and stored data.
- Any personal information of the users that are stored should be stored securely.
- Users should have an option to report others they are involved with in a lecture session.

3.5 System Models

3.5.1 Scenarios

John Doe is a system user, a student.

Scenario 1	
Use Case Name	Register
Participating Actors	John Doe, Tutor
Entry Conditions	● Actor opens up the web app ● Actor is not registered in the system ● Actor has an email account
Exit Conditions	Actor registers into system or actor cancels the registration process

Scenario 2	
Use Case Name	Login
Participating Actors	John Doe, Tutor
Entry Conditions	● Actor opens up the web app ● Actor has a registered account ● Actor is not logged in system
Exit Conditions	● Actor's credentials are correct.
Main Flow Events	● Actor goes to tutoryum.com ● Actor clicks "Login" ● Actor fills out the required areas. ● Actor logs in

Scenario 3	
Use Case Name	Logout
Participating Actors	John Doe, Tutor
Entry Conditions	● Actor is logged in
Exit Conditions	● -
Main Flow Events	● Actor clicks "Logout"

Scenario 4	
Use Case Name	List Tutors
Participating Actors	John Doe
Entry Conditions	• Actor is logged in system (though there may be tutors that can be seeable without logging)
Exit Conditions	• -
Main Flow Events	• Actor goes to tutoryum.com • Actor logs in • Actor clicks list tutors button

Scenario 5	
Use Case Name	View a Tutor’s page
Participating Actors	John Doe
Entry Conditions	• Actor is logged in the system
Exit Conditions	• -
Main Flow Events	• Actor logs in tutoryum.com • Actor list tutors and clicks one of them, or actor follows a specified URL to Tutor’s page

Scenario 6	
Use Case Name	Book a meeting
Participating Actors	John Doe
Entry Conditions	• Actor is logged in the system • Tutor is authorized and able to have meeting
Exit Conditions	• -
Main Flow Events	• Actor logs in tutoryum.com • Actor opens up a specific tutor page • Actor books an available time,

Scenario 7	
Use Case Name	Join a meeting
Participating Actors	John Doe, Tutor
Entry Conditions	• Actor is logged in the system • Actor has a meeting registered in the system
Exit Conditions	• Meeting ends, (30-45 minutes)
Main Flow Events	• Actor logs in tutoryum.com • Actor clicks the meeting link

Scenario 8	
Use Case Name	Use Online board
Participating Actors	John Doe, Tutor
Entry Conditions	• Actor is in a meeting
Exit Conditions	• Meeting ends
Main Flow Events	• Actor is in a meeting • Actor clicks online board button • Actor uses the board simultaneously with their meeting partner

Scenario 9	
Use Case Name	See Meeting Notes
Participating Actors	John Doe
Entry Conditions	• Actor is logged in the system • Actor had a meeting in the system
Exit Conditions	• -
Main Flow Events	• Actor logs in tutoryum.com • Actor goes to archive and lists their past meetings • Actor clicks meeting material for one of those meetings

Scenario 10	
Use Case Name	Comment & Rate a Tutor
Participating Actors	John Doe
Entry Conditions	• Actor is logged in the system • Actor had a meeting in the past
Exit Conditions	• Actor submits their comment and rating
Main Flow Events	• Actor logs in tutoryum.com • Actor goes to archive and find the meeting they want to rate • Actor comments and rates the tutor

Scenario 11	
Use Case Name	Verify Tutorship
Participating Actors	Tutor
Entry Conditions	• Actor is logged in the system
Exit Conditions	• Actor is verified in the system as a Tutor
Main Flow Events	• Actor logs in tutoryum.com • Actor submits their exam qualification

Scenario 12	
Use Case Name	Submit availability
Participating Actors	Tutor
Entry Conditions	• Actor is logged in the system • Actor is verified as a Tutor
Exit Conditions	• -
Main Flow Events	• Actor logs in tutoryum.com • Actor goes into profile page and updates their availability status

Scenario 13	
Use Case Name	Upload Course material
Participating Actors	Tutor
Entry Conditions	• Actor is logged in the system • Actor is verified as a Tutor
Exit Conditions	• -
Main Flow Events	• Actor logs in tutoryum.com • Actor goes into profile and submits their course material

Scenario 14	
Use Case Name	View a Tutor’s page
Participating Actors	John Doe
Entry Conditions	● Actor is logged in the system
Exit Conditions	● -
Main Flow Events	● Actor logs in tutoryum.com ● Actor list tutors and clicks one of them, or actor follows a specified URL to Tutor’s page

Scenario 15	
Use Case Name	Make Payment
Participating Actors	John Doe
Entry Conditions	● Actor is logged in the system
Exit Conditions	● Actor makes the payment
Main Flow Events	● Actor logs in tutoryum.com ● Actor makes payment for their purchases in the system ● Payment System API handles the rest

Scenario 16	
Use Case Name	Receive Funds
Participating Actors	Tutor
Entry Conditions	• Actor is logged in the system • Actor has withdrawable amount of money in their wallet
Exit Conditions	• Actor receives the funds and
Main Flow Events	• Actor logs in tutoryum.com • Actor clicks the button to withdraw the funds • Payment System API handles the rest

3.5.2 Use-Case Model

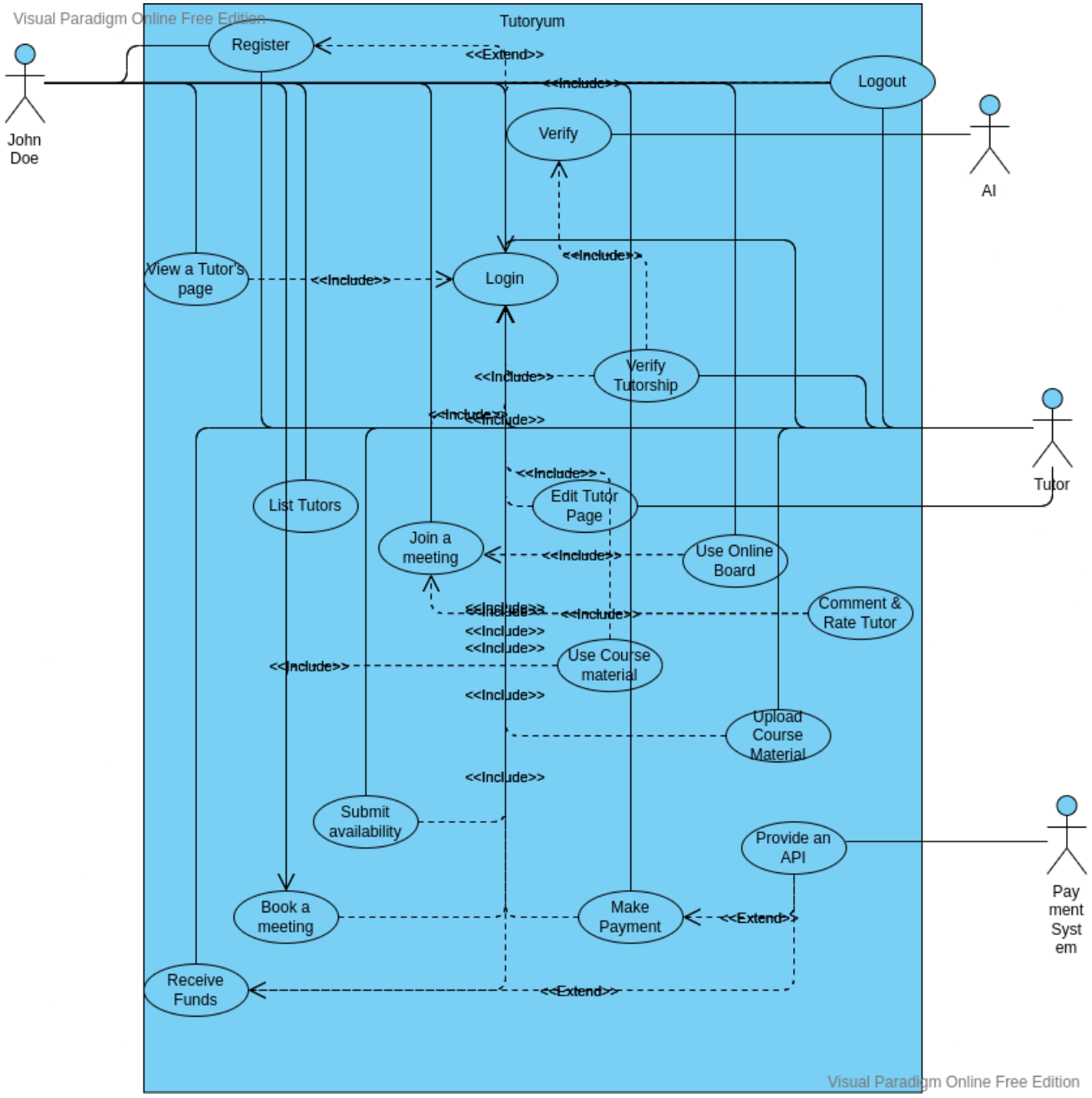

Figure 1: Use Case Diagram

3.5.3 Object and Class Model

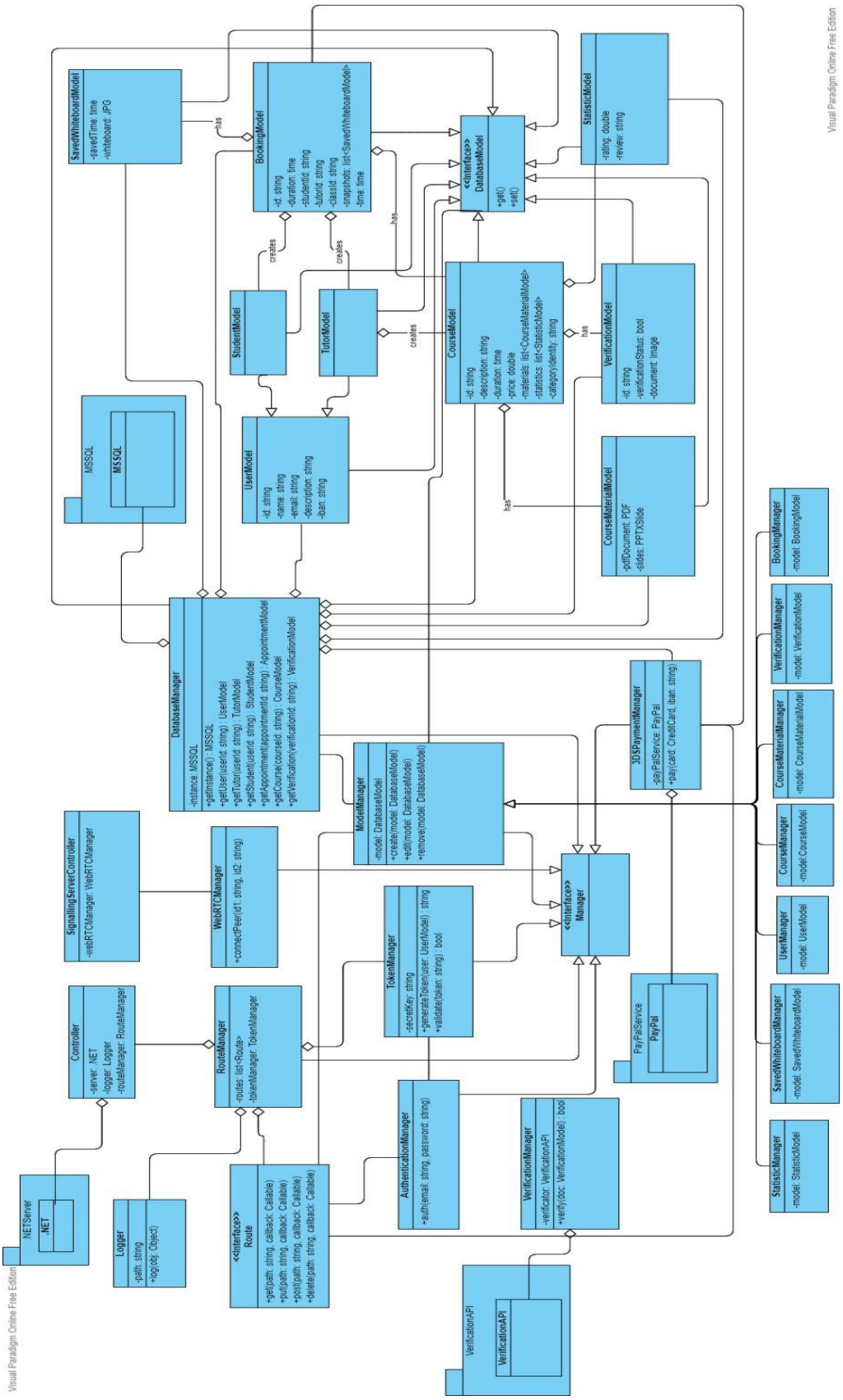

Figure 2: Object and Class Model for Server Side

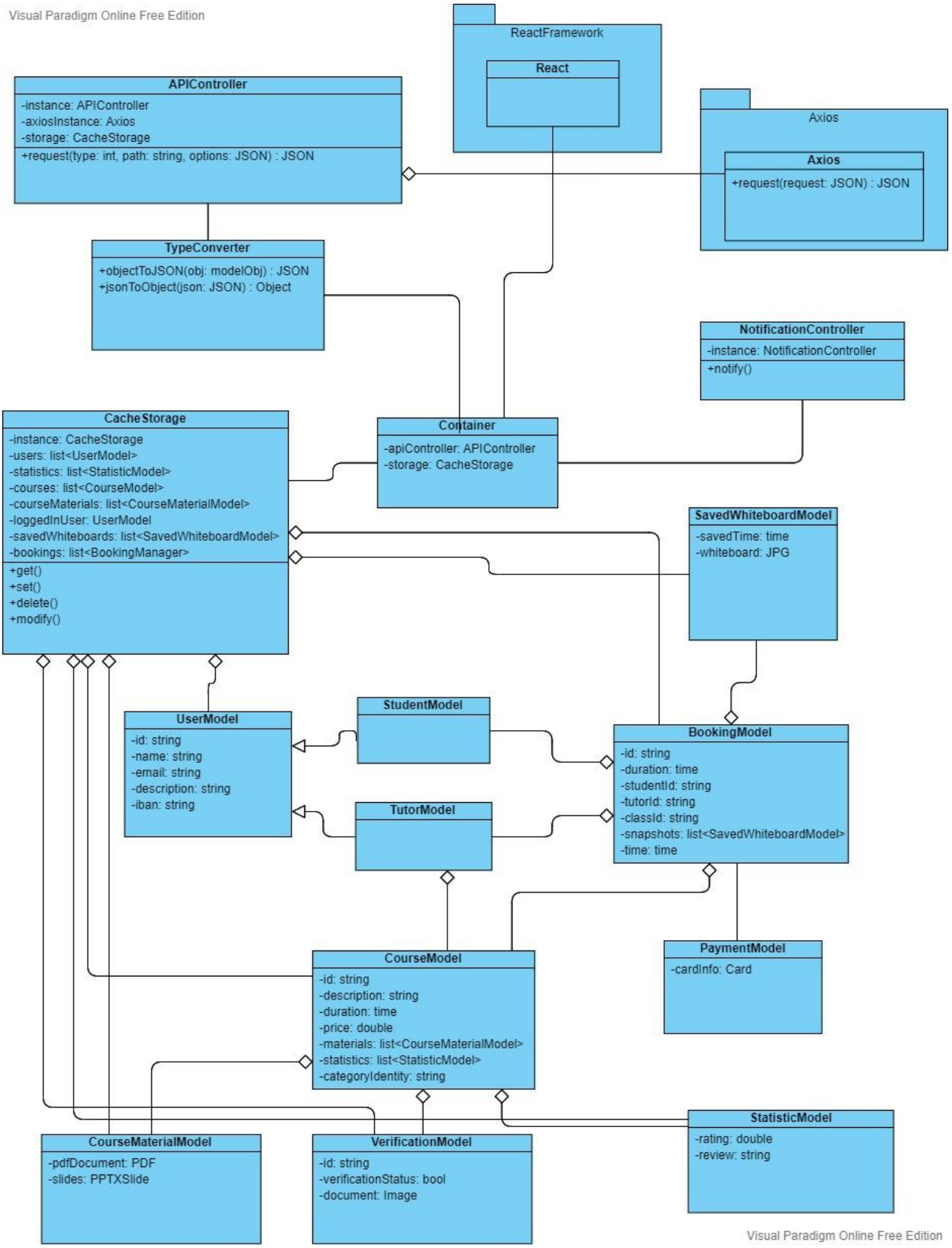

Figure 3: Object and Class Model for Client Side

3.5.4 Dynamic Models

3.5.4.1 Sequence Diagrams

- Register System Workflow

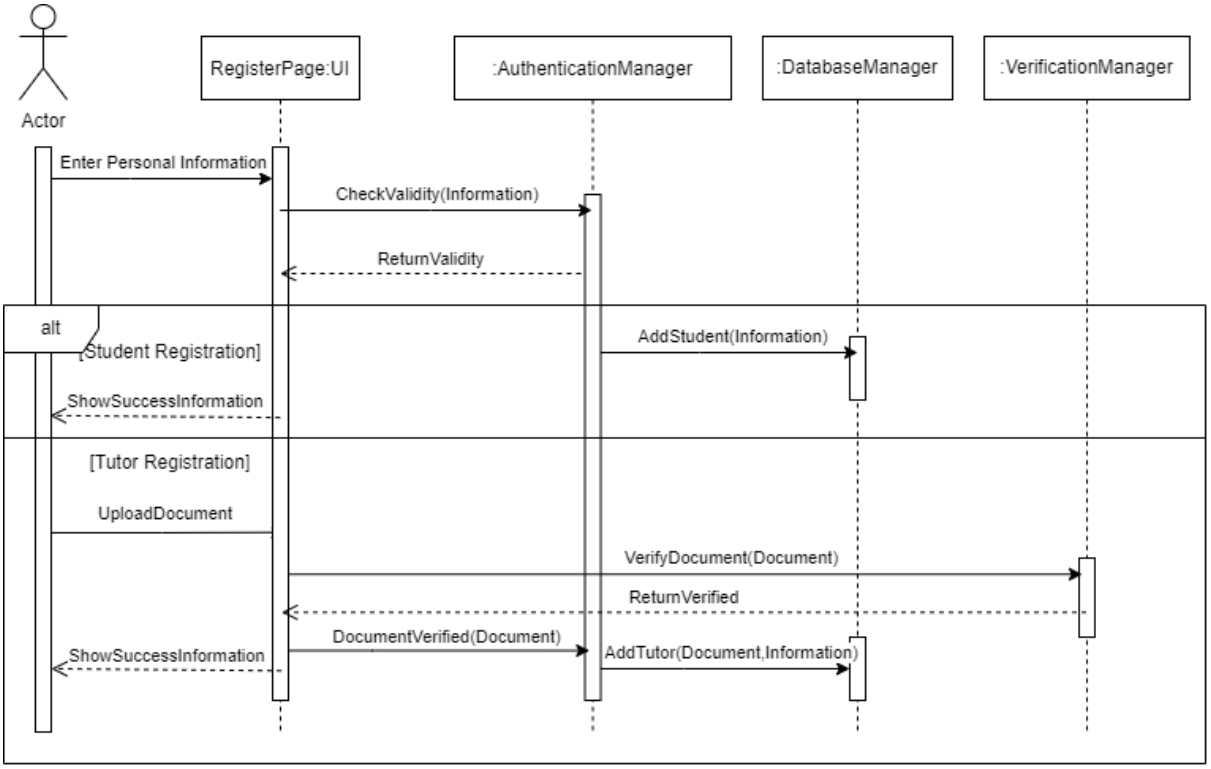

Figure 4: Register System Workflow

Scenario: An actor wants to join the system as a user

To register the system, an actor must be on the register page of the website. On this page, there are spaces for users' login information. The user also enters whether s/he wants to be a tutor in the system or a student. If s/he chooses to register as a student, AuthenticationManager checks whether the information s/he entered is valid by getting the required information from the database. The student will be added to the system if all the information is valid. Suppose an actor wants to register the system as a tutor after finishing the validity check for information. In that case, s/he uploads the required document to be a tutor, and VerificationManager controls whether this document is valid. If it is valid, then the user will be added to the system as a tutor.

- Login and Logout Workflow

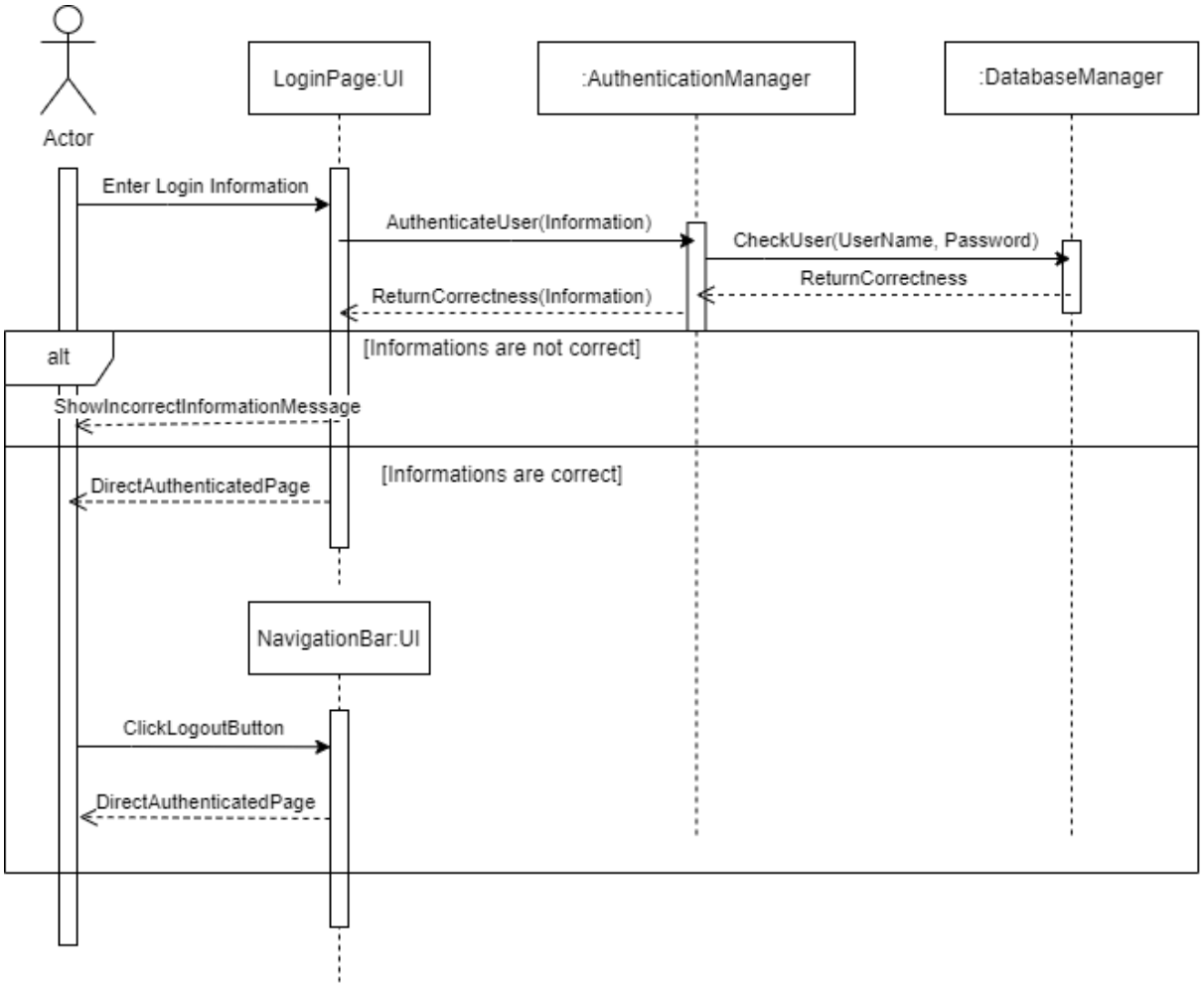

Figure 5: Login and Logout Workflow

Scenario: A user wants to log in to the system and then log out of the system.

To log in to the system, an actor must be on the website's login page. On this page, there are spaces for users' login information. The user enters his/her login information there then AuthenticationManager checks whether the information entered is correct by asking DatabaseManager. If the given information is incorrect, the user will see an incorrect message in UI. The user will be directed to a home page with authentication if the information is correct. If a user has authentication while navigating through Tutoryum, there is a logout button on the NavigationBar. If s/he clicks that button, now she will be directed to an unauthenticated web page.

- Listing Tutors Workflow

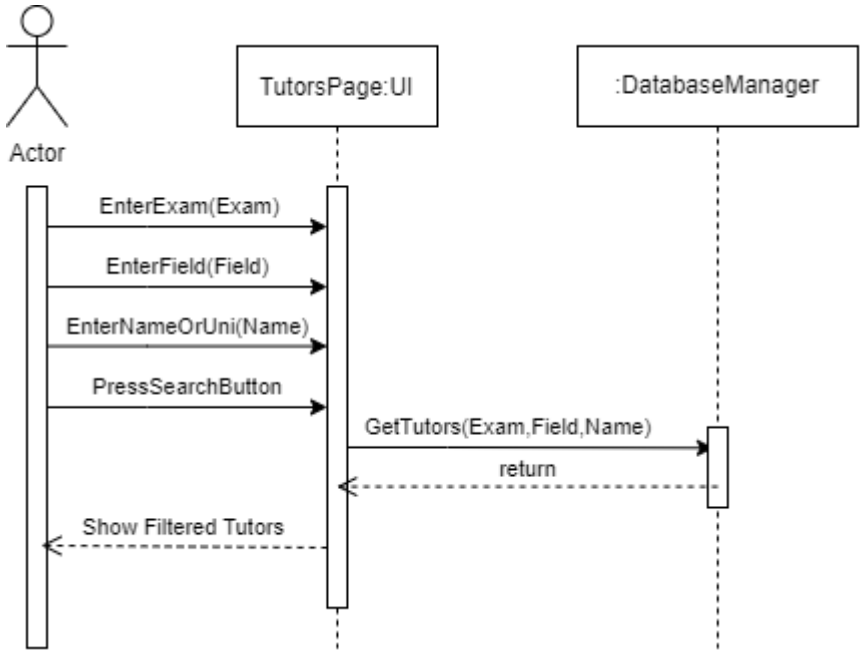

Figure 6: Listing Tutors Workflow

Scenario: A user wants to list tutors according to his/her preferences.

To search for tutors, a user must be on TutorsPage. On this page, there are spaces for detailed filtering for tutors. The user can enter the name of an exam for which the tutor uploads a document, a field of courses a tutor gave, and the tutor's name or university. After entering these pieces of information, the user can press the search button to filter all the possible tutors. When clicking that button, UI requests fitting tutors from DatabaseManager and then reorganizes the page concerning the information returned.

- Book Meeting Workflow

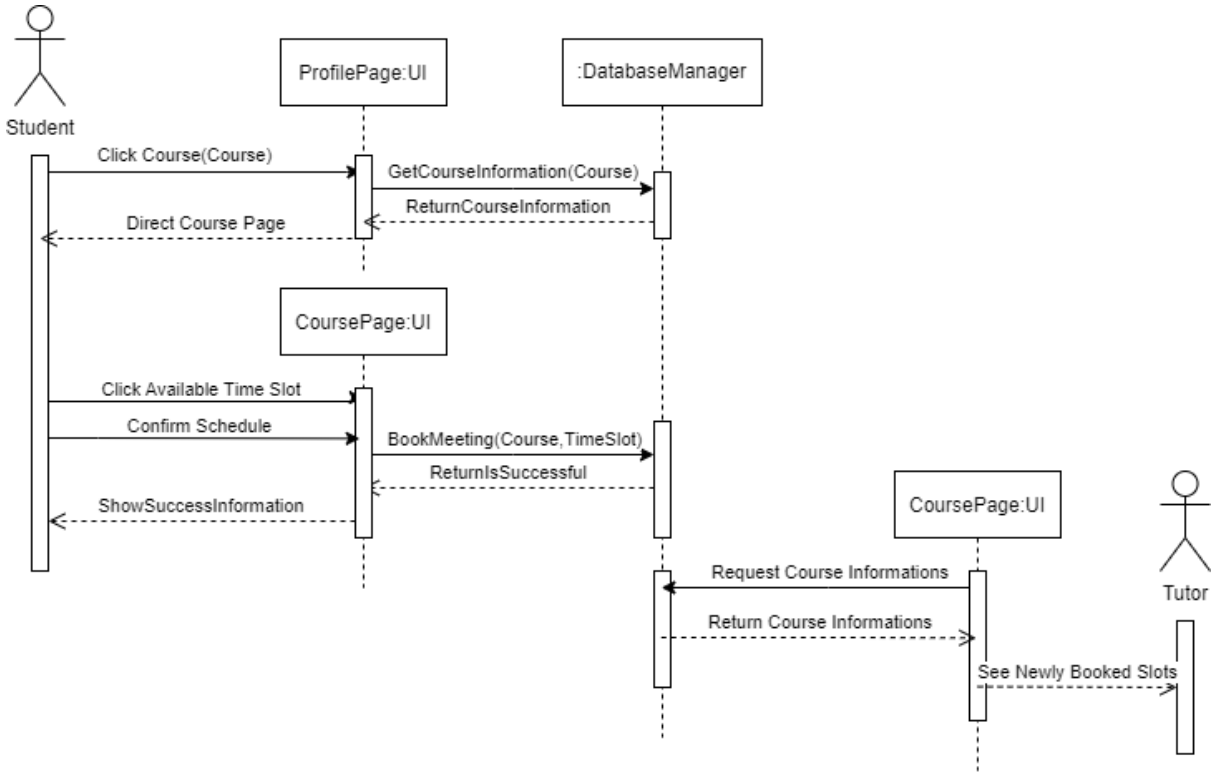

Figure 7: Book Meeting Workflow

Scenario: A student wants to book a meeting with a tutor.

First, a student has to visit the profile page of a tutor who s/he wants to book a meeting. On this page, all the courses that tutor was giving were listed. By clicking one of these courses, UI requests the contents of that course from DatabaseManager and then directs the student to the corresponding courses page with the requested information. On this page, there is a space for available meeting times of the tutor. By clicking one of these time slots, students will be asked to confirm if they want to book this slot. If s/he confirms this, then DatabaseManager adds a new meeting to the system. After this, when the tutor of this course enters this course's page, s/he can see available places with updated information.

- Join Meeting Workflow

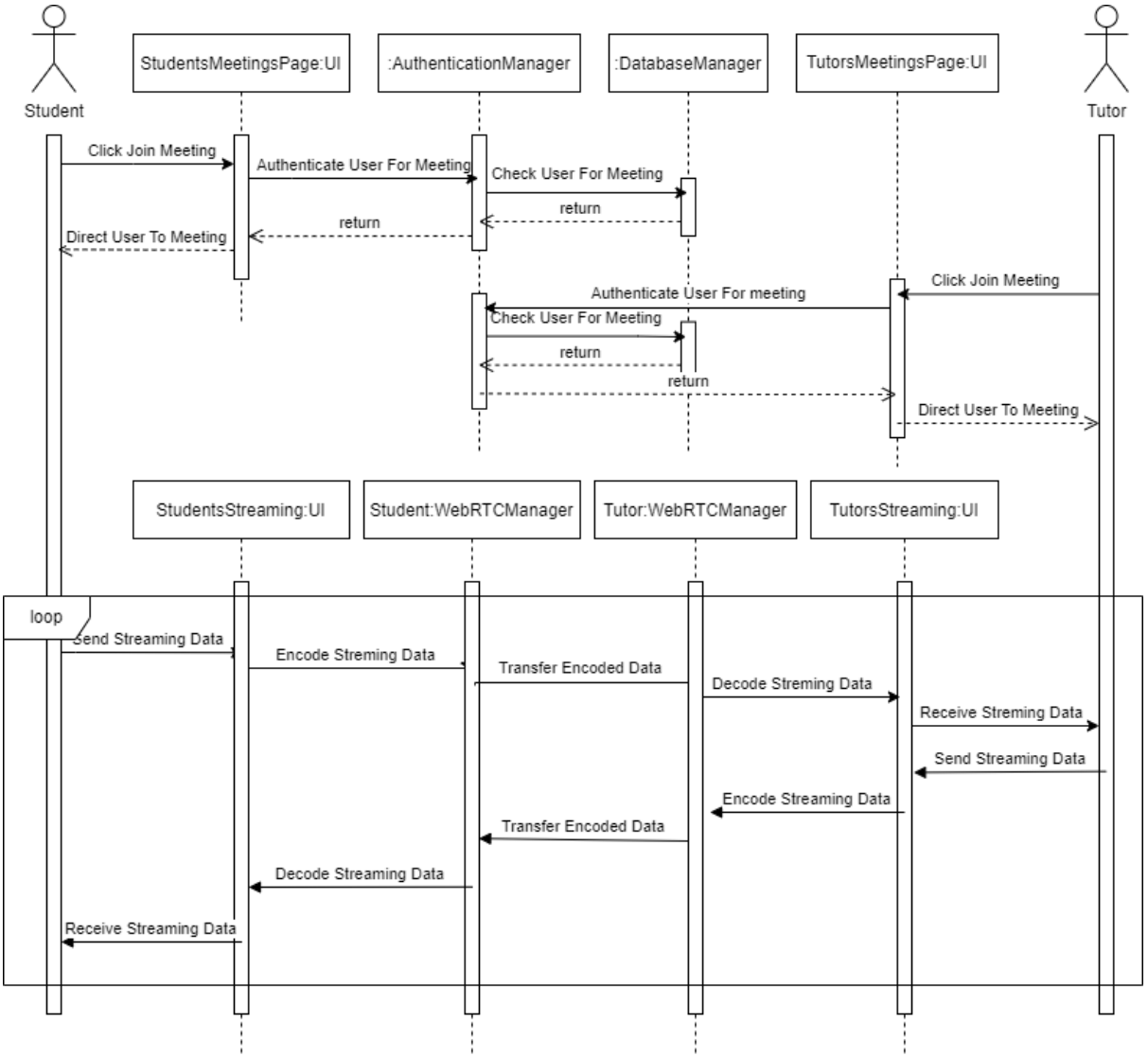

Figure 8: Join Meeting Workflow

Scenario: A student and a tutor want to join an already booked meeting.

Both the student and the tutor can see upcoming meetings from their MeetingsPage. By clicking the join meeting button under the upcoming meeting, they will be directed to the Streaming page after the authentication check for both of the users using AuthenticationManager. As the connection between them will be peer-to-peer, WebRTCManagers will handle data transformation between each side of the network. Therefore until the end of the meeting, data transformation will be handled by WebRTCManagers.

- Commenting and Rating Tutors Workflow

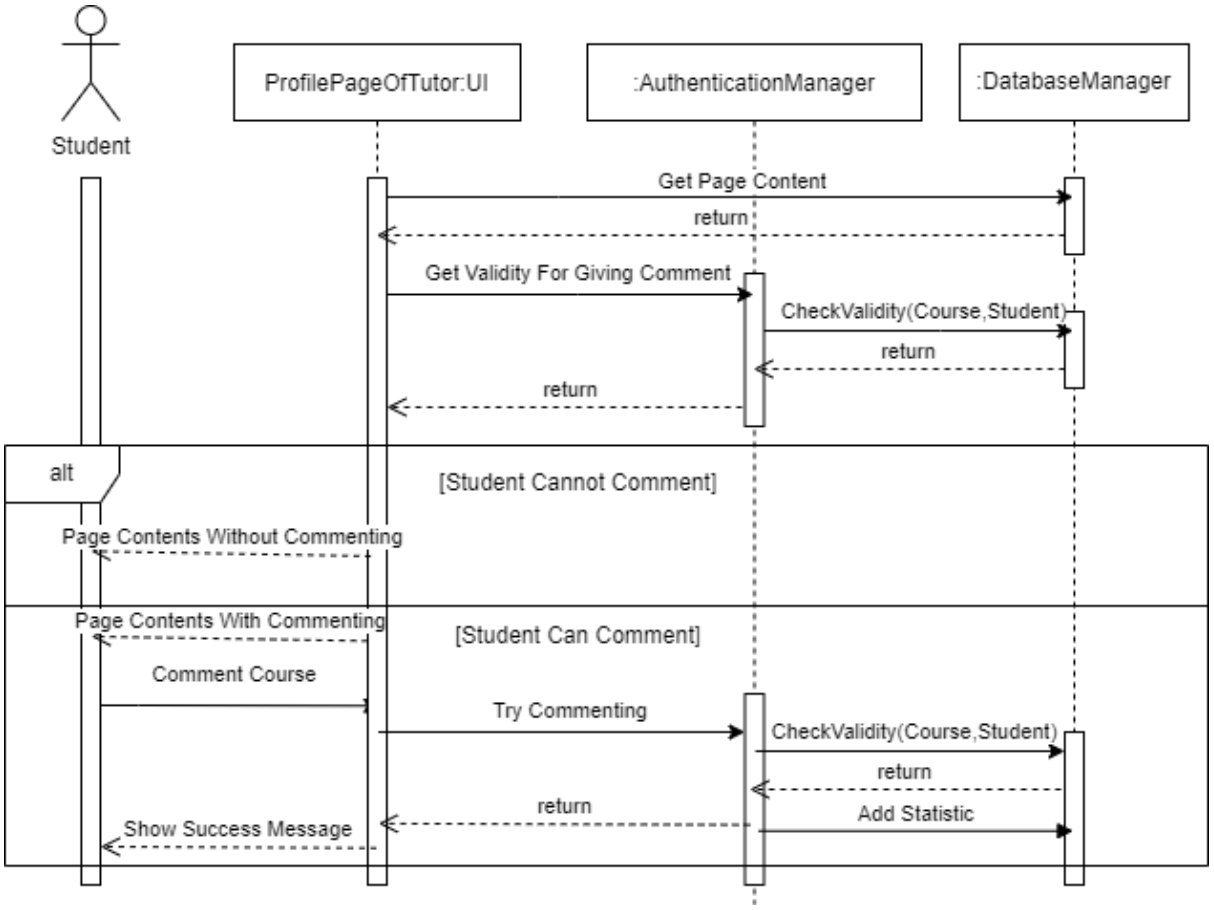

Figure 9: Commenting and Rating Tutors Workflow

Scenario: A student wants to comment on some tutor's course.

To comment on the course, first, the Student must enter the Profile Page of the tutor who gave that course. While the page is trying to get contents from DatabaseManager, it also asks AuthenticationManager whether this Student has taken a course from that particular tutor. If the Student didn't take a course from this tutor, commenting button does not show up on UI. Otherwise, it shows a button for commenting. After the user writes a comment on that course, UI tries commenting using AuthenticationManager, which checks whether that Student is eligible to comment on that course. If s/he passes this check again, DatabaseManager adds a new statistic to the system and UI shows a success message to the Student.

- Uploading Course Material Workflow

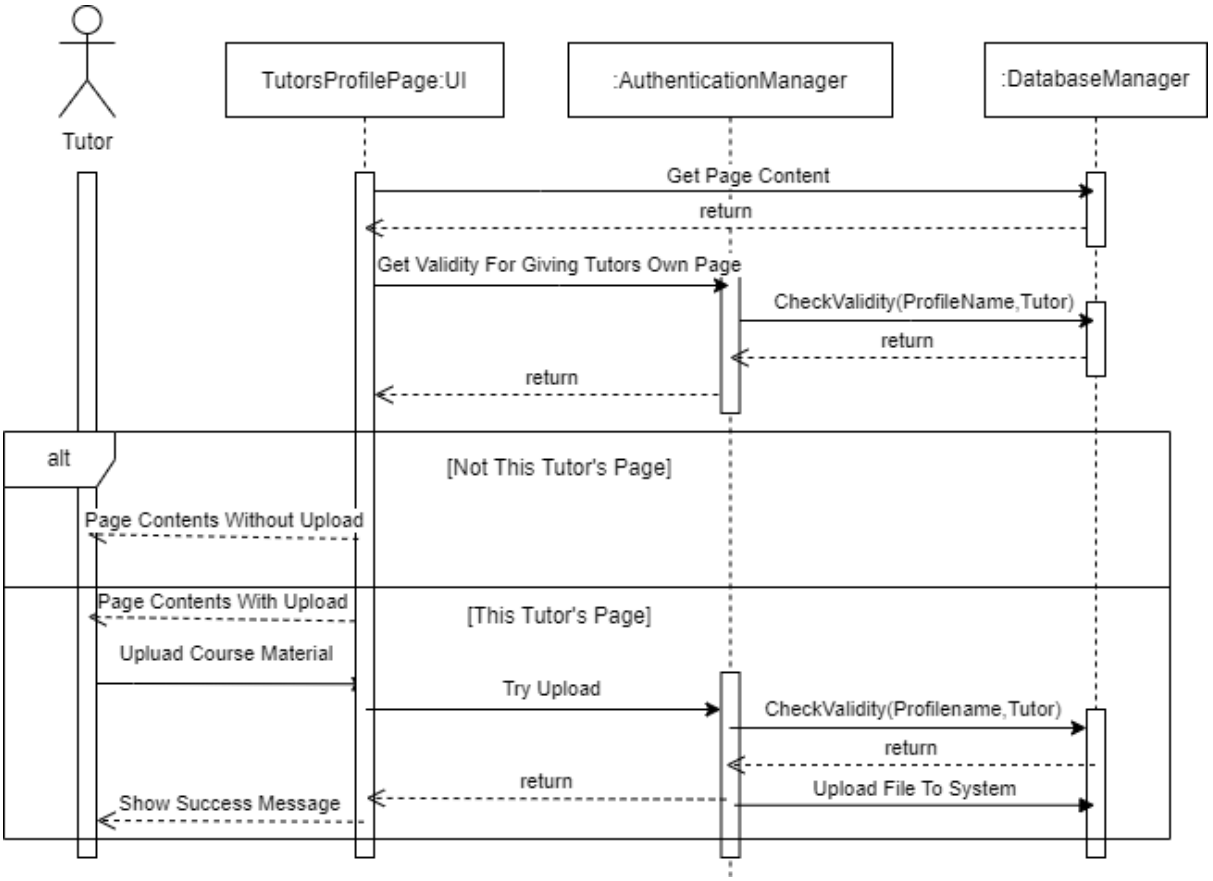

Figure 10: Uploading Course Material Workflow

Scenario: A tutor wants to upload new material to the course.

To upload material for a course, first, the Tutor must enter his/her Profile Page. While the page is trying to get contents from DatabaseManager, it also asks AuthenticationManager whether this page belongs to the Tutor who is trying to access it. If it is not the Tutor's Profile Page, the upload button won't appear on the UI. Otherwise, it shows a button for uploading. After the Tutor tries to upload a file to this course, UI tries uploading using AuthenticationManager, which again checks whether that Tutor is eligible to upload material to that course. If s/he passes this check again, DatabaseManager adds a new file to the system, and UI shows a success message to the Tutor.

3.5.4.2 Activity Diagrams

In this part, activity diagrams to show the main usages of the application will be given and discussed in detail. They will be used to demonstrate three main flows

of the application from students' and tutors' perspectives. In each one, users will perform certain actions, resulting in other actions on specific conditions.

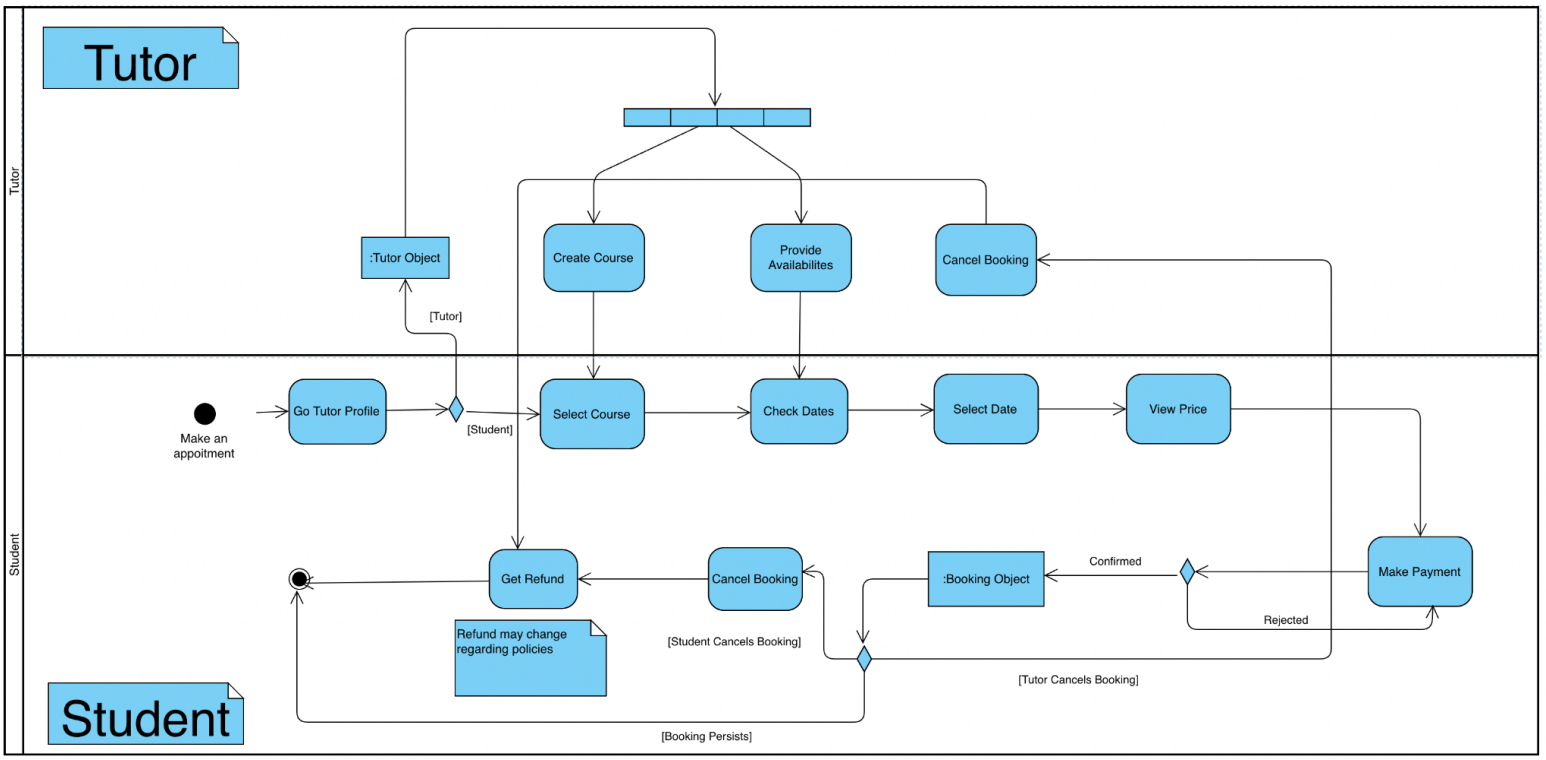

Figure 11: Activity Diagram for Booking Objects' Lifetime

In this diagram, the flow of the actions for making a booking is shown. Let's have a look at students' actions to begin with. First, students need to 'Go the Tutor Profile' from whom they want to take a lecture. Then, they need to 'Select a Course' from the tutor's profile, e.g., YKS. Second, they need to 'Check Dates' and see the tutor's availability. Moreover, they need to 'Select a Date' that fits into their schedule so that a booking object can be created on that date. Third, students can 'View the Price.' Then, they need to 'Make a Payment. It is important to note that if their payment is rejected, they need to 'Make a Payment' again. There is a loop here that indicates that the payment object will not be created until a successful payment is made. If the payment is confirmed, the booking object is created for students and the tutor of their choice. Note that after the creation, there are further actions that both students and tutors can take. They can 'Cancel the Booking,' which results in a 'Get Refund' action. It is essential to know the refund amount or whether there will even be a refund, depending on the policies. If users perform the cancel operation, the

booking object is destroyed, and the activity diagram finishes. If the booking goes as planned, then the activity diagram ends simply without destroying the booking.

Now let's check out what tutors' actions are for this diagram. As we discussed, they can also 'Cancel the Booking,' which would result in the same flow as in the students. On the other hand, tutors have two other significant actions in this diagram: ' Create Course' and 'Provide Availabilities.' If students want to book with a tutor, that tutor needs to 'Create the Course' beforehand. Note that you can find another activity diagram for course creation. The other action, 'Provide Availabilities', again needs to be executed by tutors before students can make a booking from them. Without any availability, students will not be able to select a date.

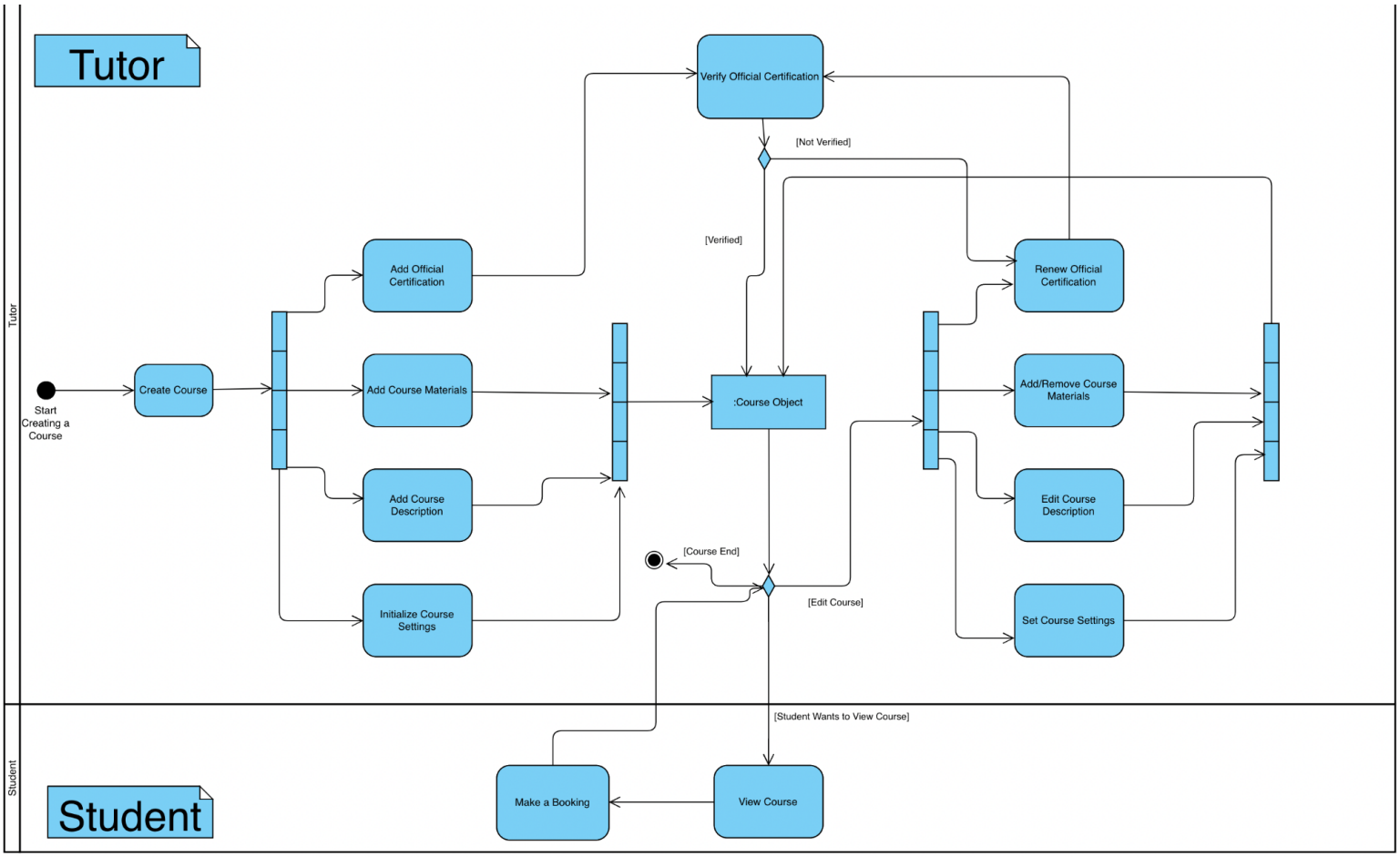

Figure 12: Activity Diagram for Course Objects' Lifetime

In this diagram, the flow of the actions for creating a course as well as managing it is shown. Let's have a look at tutors' actions to begin with. First, they need to 'Create a Course.' On the creation of the course, they can apply four

different actions. 'Initialize Course Settings' is for adjusting the course settings. In this action, tutors can select the price of the new course as well as the duration of the lectures. 'Add Course Description' is for adding a course description. In this action, tutors can put a brief description to promote their course, especially if it is exotic. 'Add Course Materials' is for uploading course documents so students can benefit. These can include pdf as well as slide shows. 'Add Official Certification' is for uploading a certification document that verifies that the tutor is competent in the course. Note that this action results in another action, ' Verify Official Certification.' The system will execute this action automatically to confirm that provided document is correct. Once four actions from the tutor side finish, the course object will be created. If the 'Verify Official Certification' action is verified, this will be added to the course object and returned to the tutor as a shiny badge. However, if it fails, the tutor can retry to add official documentation with the 'Renew Official Documentation' actions, as shown in the figure. Once the course is created, tutors can edit the course object with the same four actions described above: ' Set Course Settings,' 'Edit Course Description,' 'Add/Remove Course Materials, and 'Renew Official Certification.' Note that the certification documents may have an expiration date. After this date, the course object will return to the unverified states. To verify again, tutors must take the 'Renew Official Certification' action. Once tutors remove the course, this activity diagram will end.

Now let's check out what students' actions are for this diagram. These actions were further covered in the previous activity diagram, called 'Activity Diagram for Making a Booking.' The students will be able to 'View Course' and then 'Make a Booking' regarding the course object, as shown in the diagram.

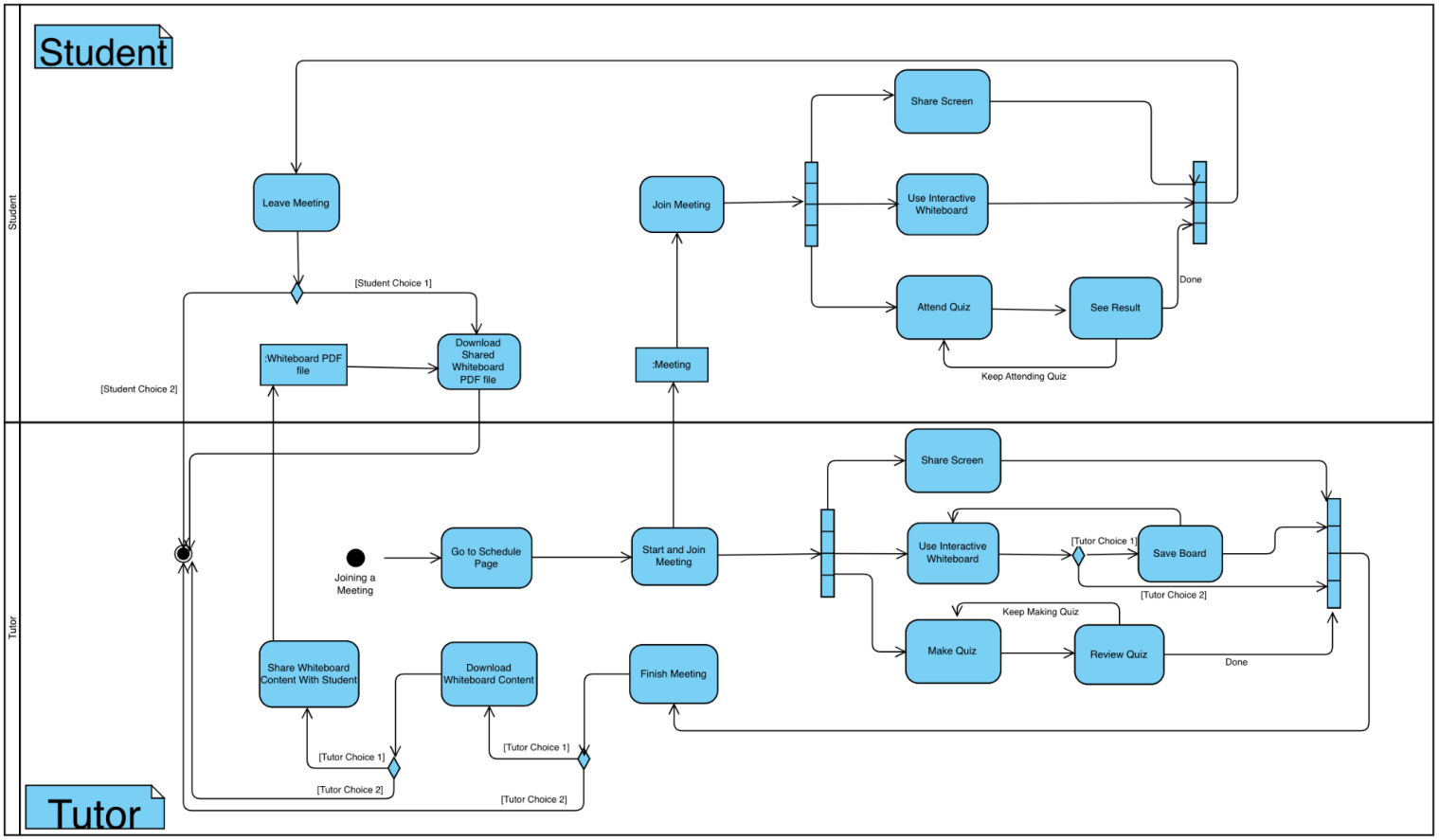

Figure 13: Activity Diagram for the Meeting Component

In this diagram, the flow of the actions for joining a meeting and the lifetime of the meeting component will be shown. Now, first, have a look at the tutors' actions. First, tutors will 'Go to the Schedule Page' that belongs to them. There, they will be able to see their current and upcoming meetings. With the help of a button, they will "Start and Join the Meeting." This action will create the meeting component, and all the logic will begin. While in the meeting, they can apply several actions in parallel. These are 'Share Screen,' 'Use Interactive White,' and 'Make Quiz.' With the 'Share Screen' action, tutors can share their screens with students. Students can observe how tutors solve particular problems thanks to this. With the 'Use Interactive White' action, tutors can draw to a whiteboard that will be visible to students in real-time. This can be used for solving mathematical equations as well as designing projects. While using the whiteboard, tutors can save its content whenever they want. Note that this is optional and shown in the diagram with Decision Node, which looks like a

diamond. The tutor can create and test their students with the' Make Quiz' action. Students will answer these questions in real time. After the answering phase of a student, tutors will be able to review the quiz and give the student a grade. This will help them to understand how well they are in their job and how well students attend the meeting. After the meeting gets close to its end, tutors will be able to finish the meeting. After the end of the meeting, tutors can take several actions or skip them. They can choose to 'Download White Board' content which consists of the saved screenshot from the meetings (see 'Save Board' action). After this action, tutors may or may not share this file with the respective student who attended the meeting. All of these choices are shown with Decision Node in the figure.

Now let's check out what students' actions are for this diagram. First, once the meeting component is created, they can perform a 'Join Meeting' action. With this, they will enter the meeting. Like tutors, they can perform 'Share Screen' and 'Use Interactivate Whiteboard' actions. These actions are explained in the previous paragraph. However, unlike the tutors, students cannot 'Save Whiteboard.' Another difference is the 'Attend Quiz' action. With these actions, students can participate in the quiz released by the tutor. After they answer the question and the tutors is reviewed them, the student will be able to 'See the Result' of the quiz. After the meeting gets close to its end, students will be able to leave the meeting. If tutors share the whiteboard content with students, they will have a chance to apply the 'Download Shared Whiteboard PDF File,' from which they can see the saved content of the whiteboard as prepared by the tutor.

3.5.4.3 State Diagram

In this part, state diagrams for the entities will be given and discussed in detail. State diagrams will be used to show the exciting behaviors of the objects. They will mainly focus on how objects react to external stimulants at certain states.

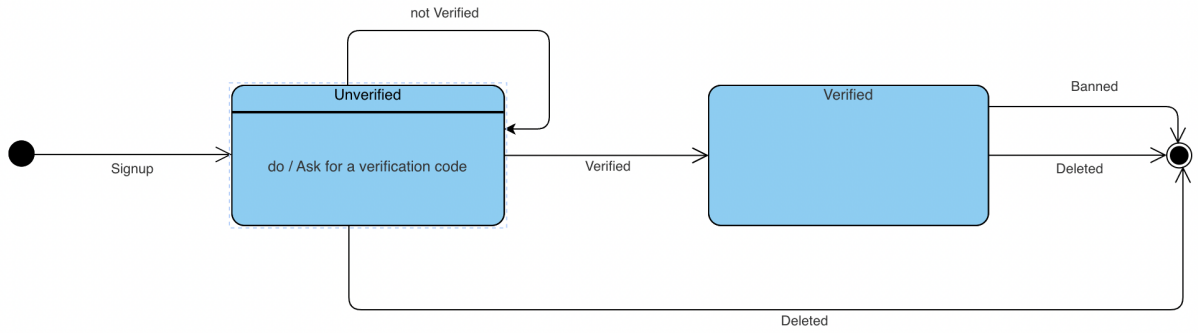

Figure 14: State Machine Diagram for a User objects

This diagram shows how each user's account will live through their lifetime. First, the user will create their accounts as either an instructor or a student. They will input their names, email addresses, phone numbers, etc. Afterward, the user object will initially be in the unverified state. The user will be required to verify their accounts via email and phone number. Once their verification is confirmed, their account will be in the verified state. Before creating a course (for instructors) or reserving a timeslot (for students), user status will be enforced to be verified. Note that there are three situations where user accounts can be erased. The first one is that if the user does not verify their accounts, their information will be deleted after a determined time. The second one is if the user does not obey the rules of the website, they will be banned, and their account will be erased. The third one is if the user wants to be forgotten, their account will be deleted. Note that this step is vital to conform with GDPR's law, mainly the 'Right to Be Forgotten' [8].

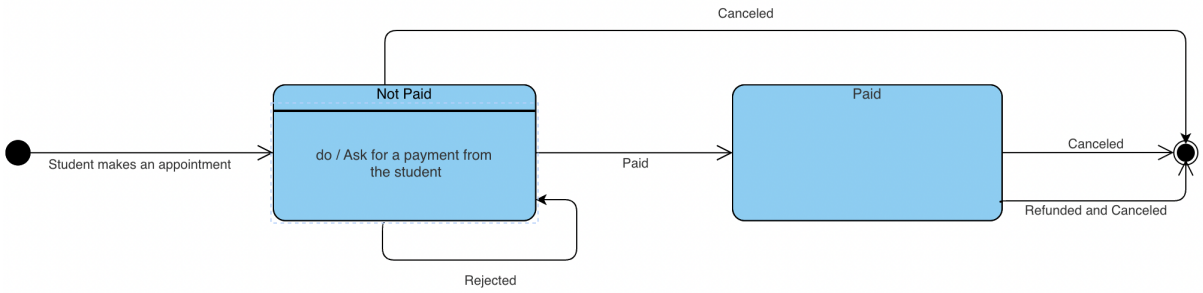

Figure 15: State Machine Diagram for the payment logic of the booking object.

As the final stage of making a booking, it will be mandatory to make payment. Let's elaborate on this further. First, students will choose a time slot from a tutor to make a booking. Afterward, they will be required to make a payment to confirm and create their bookings. The user will be in the 'Not Paid' state in this step. Note that they do not lose the slot chosen immediately if their payment is rejected. They will have a chance to make a payment again. It is important that they will stick to the 'Not Paid' state until they make a successful payment. They will proceed to 'Paid State' if the payment is successful. This will result in confirmation of the booking, and students will be able to learn their favorite topics. Note that two possibilities may destroy the booking object and, therefore, result in a refund. First, students may cancel their booking request in the 'Not Paid' state. This is straightforward that there will be no payment or changes in the database, and the payment logic will be halted immediately. Second, students/tutors may cancel their booking after it is confirmed. In this case, there are two possible outcomes. The students may be entitled to get a refund or not (depending on the policy). In either case, booking objects will be destroyed.

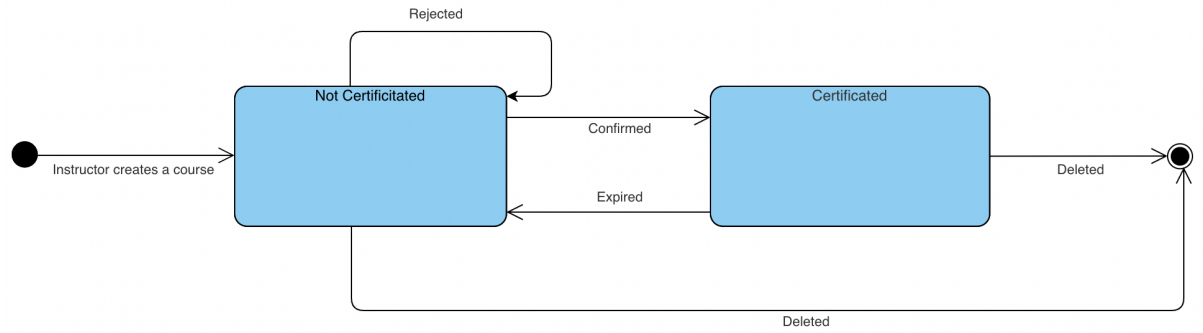

Figure 16: State Machine Diagram for the verification logic of the course object.

Some courses will be verifiable. For example, the university entrance exam of Turkey (YKS) will be one of those verifiable courses. The tutor giving that course will be able to upload certification to prove that they are indeed having success in that course. In the system, each course object will have two states indicating whether tutors have certifications on the course they are giving. When the instructor creates a new course, the course they created will be in the ‘Not Certificated’ state. The instructor may provide the necessary documents to certify their courses. If the verification request fails, the course will stay in the ‘Not Certificated’ state. The course object will move into the ‘Certificated’ state if the request is confirmed. On the other hand, if the given documents have expiration days, then the course objects may go back to the ‘Not Certificated’ state. Tutors will be able to provide further documents in this step to get their verification from the system one more time. Note that in any one of the states, tutors might choose to delete the courses they are giving. This action will destroy the course objectives.

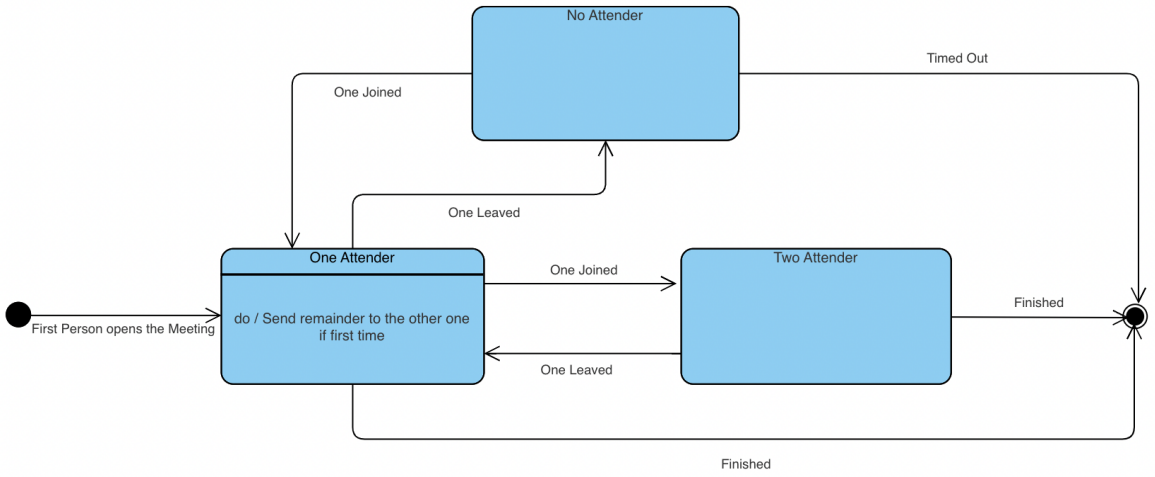

Figure 17: State Machine Diagram for the Meeting Component.

The meeting component can be in different states regarding the number of attendees. The meeting component is created when the first person opens the meeting and moves into the ‘One Attender’ state. In this step, the other person will be notified so that s/he can know that the meeting has started. Once the other person joins the meeting, the meeting will be in the state of 'Two Attenders.’ In this state, the lecture can begin, and both tutor and student can enjoy the features of Tutoryum, such as interactive WhiteBoard or real-time quizzes. If, at any moment, one of the attendees leaves the meeting, the meeting goes back to the ‘One

Attender' state. Note that this time the leaver will not be notified. If the only remaining person also leaves the meetings, then the meeting component moves into the 'No Attender' state, where it stays idle. After a while, if both participants do not come back, time out will occur, and the meeting component will be destroyed. Another way to destroy the meetings component is for the tutor to finish it. In this case, the meeting component will also be destroyed.

4. User Interface Design

4.1 Navigation Bar

Figure 18: Navigation Bar

The left part of the navigation bar contains the logo with three different navigation tools that various direct parts of the site, which will be discussed later. This part might expand in the feature corresponding to the feature developments. The right part of the navigation bar will include buttons that will direct the user to either the sign-up page or the login page. If the user is already logged in to the site, the user's name will be displayed instead of those buttons. Upon clicking the name, the user will be guided to his profile page.

Users will be asked to enter their name, surname, email, and password on the sign-up page to create their accounts. Upon signing up, a verification email will be sent to the user's email address. Upon verification, users will be able to log in to their accounts.

4.2 Home Page

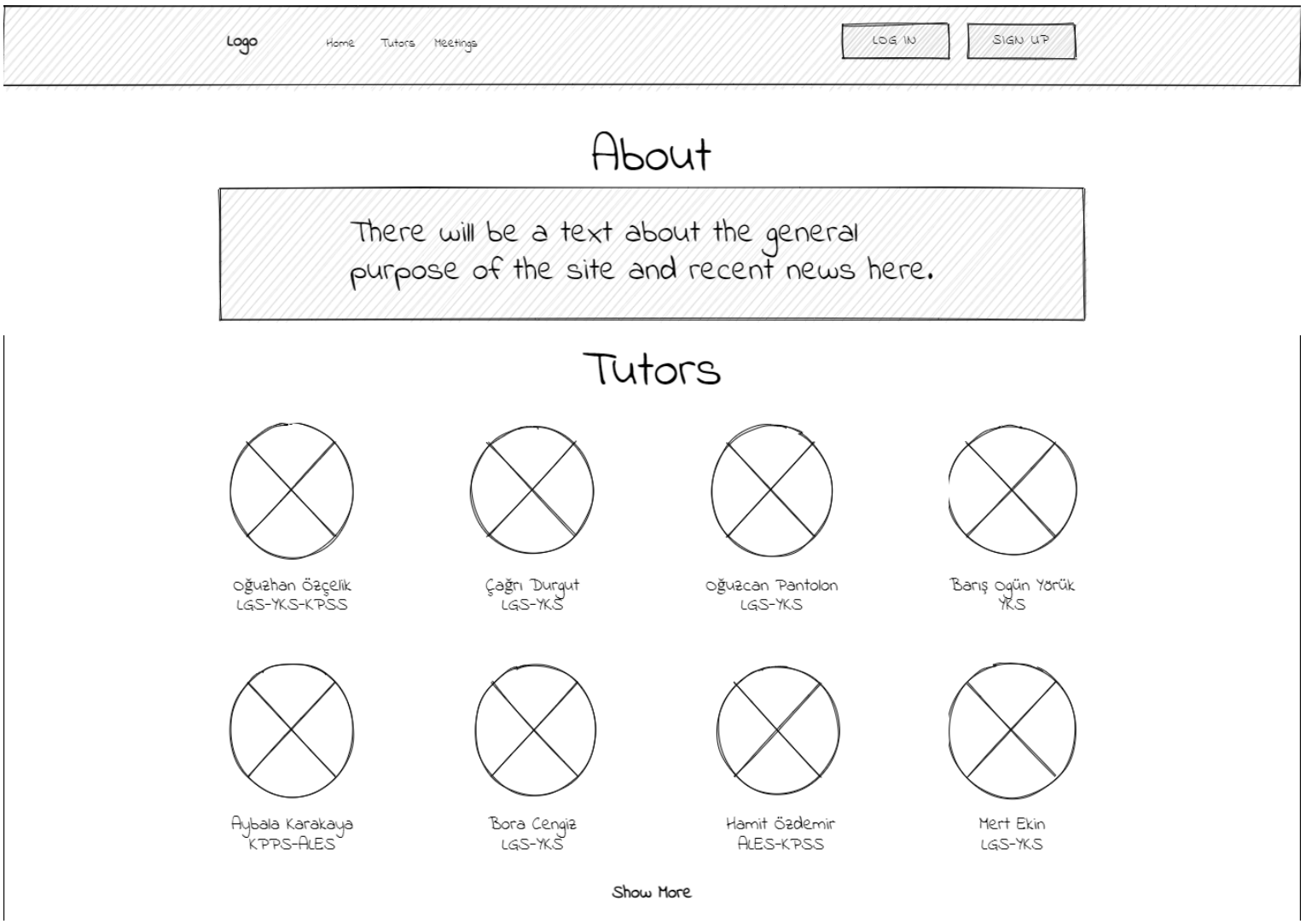

Figure 19: Home Page

The home page can be accessed by clicking “Home” on the navigation bar. The home page will consist of two parts. In the upper part, users will be informed about the general purpose of the site and recent news, like upcoming updates or changes made to the policies.

On the lower part of the home page, eight of the active tutors that use the site will be showcased as recommended tutors. Tutors’ image, name, and their most popular course names will be displayed. At the bottom part, users will be able to go to the Tutors Page by clicking the “Show More” button.

4.3 Tutors Page

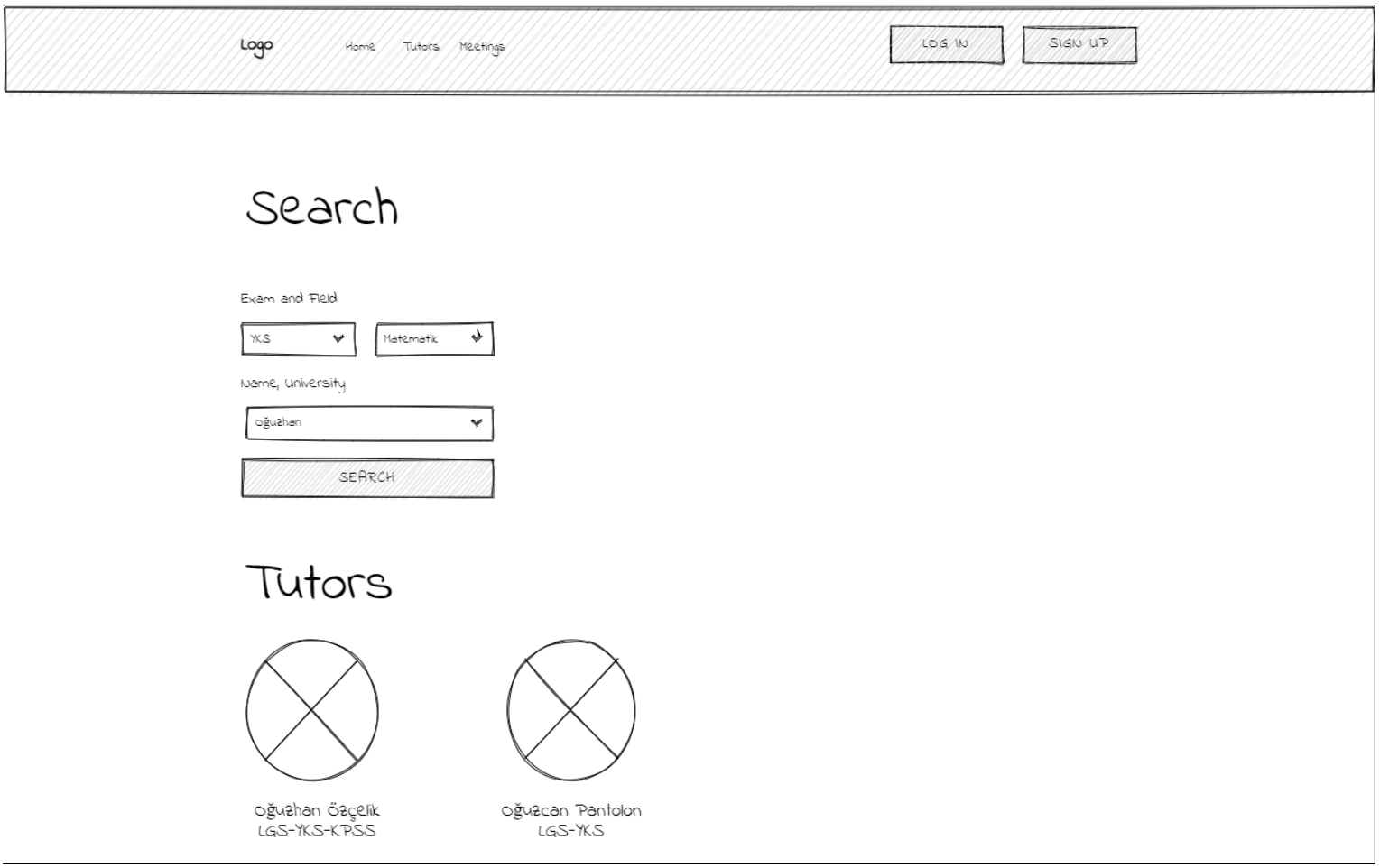

Figure 20: Tutors Page

Users will be guided to this page upon clicking “Tutors” on the navigation bar. Initially, all the users will be displayed as a list, familiar with the showcase part on the home page. Users will be given a search tool to filter the tutors. They will be able to filter users by selecting a field and name. The tool can search for any empty choices made by the user. An example in the above image is that the filter displays two tutors with the desired filter choices.

4.4 Profile Page

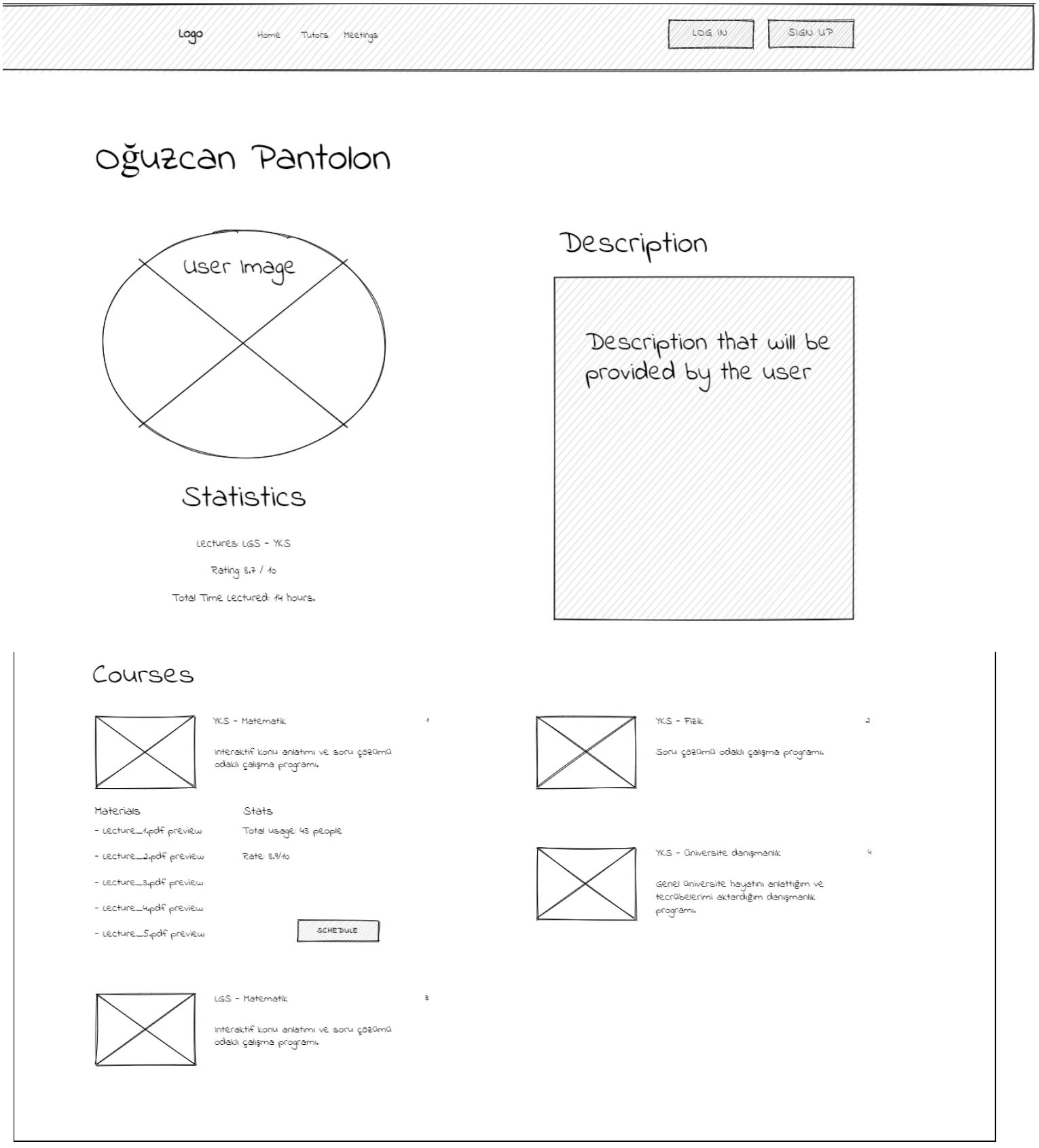

Figure 21: Profile Page

There will be a unique profile page for each user. Clicking on a user's name will navigate the corresponding user's page. This page will consist of three parts: the upper left part, the upper right part, and the bottom part.

The username and profile image provided by the user will be displayed on the upper left part. Initially, a pseudo image will be displayed as an image. Upon clicking the image, users can change their profile picture. There will be a statistics part below the profile picture. The statistics part will show the field of the courses given by the user, overall ratings of the courses given by the user, and the user's total spend hours as a lecturer.

The upper right part will include a brief description of the users. The users can edit those descriptions upon clicking on them.

At the bottom part, a list of courses given by the user will be displayed. Clicking on course items will make the item expand to give further information. Only the field and description of the courses will be given by default. On the expanded version, users can see statistics of the course and course materials used on the course. Users will be able to preview limited parts of any of the given course materials. If interested, users can schedule with the tutor by clicking the schedule button. Tutors will be able to add course materials from their profile pages. Upon expanding the material, they will be given the option to upload their slides or pdf files.

4.4.1 Schedule Page

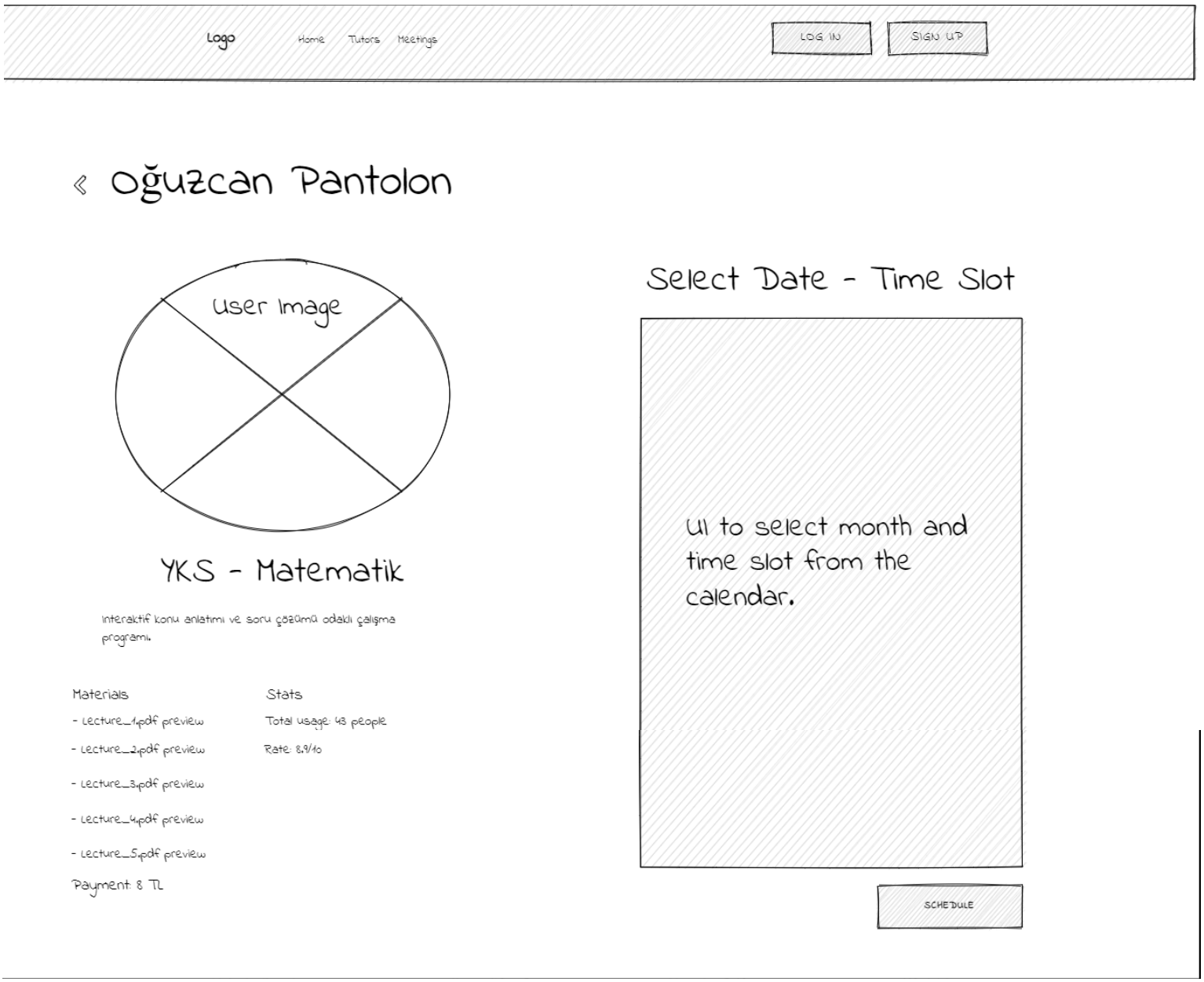

Figure 22: Schedule Page

Users will be guided to the schedule page when clicking the schedule buttons on the course items on the tutors' profile page. This page will consist of two parts. On the left part, the image of the tutors and detailed information about the course will be displayed. On the right side, users can schedule the course on a calendar. Users can only select times that tutor provided as free time.

4.4.2 User Settings Page

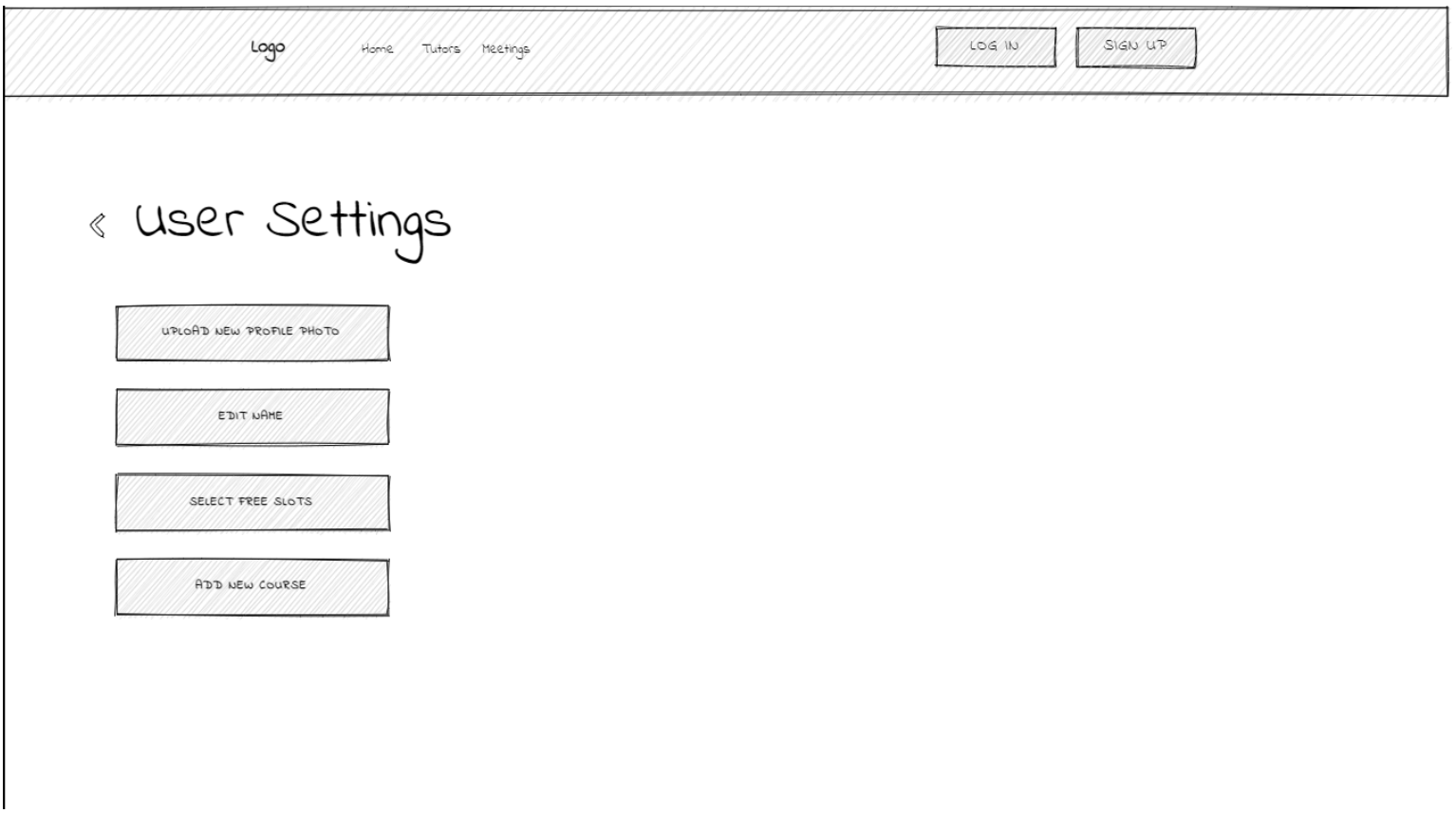

Figure 23: User Settings

Users will be able to access the settings page from their profile page. On this page, they can change their name, upload a new profile image, select their free time to give courses, and add a new course to their given courses. They will be given a calendar to select their free time to lecture. To add a new course to their courses, they will provide the necessary information to add a course addition page.

4.4.3 Course Addition Page

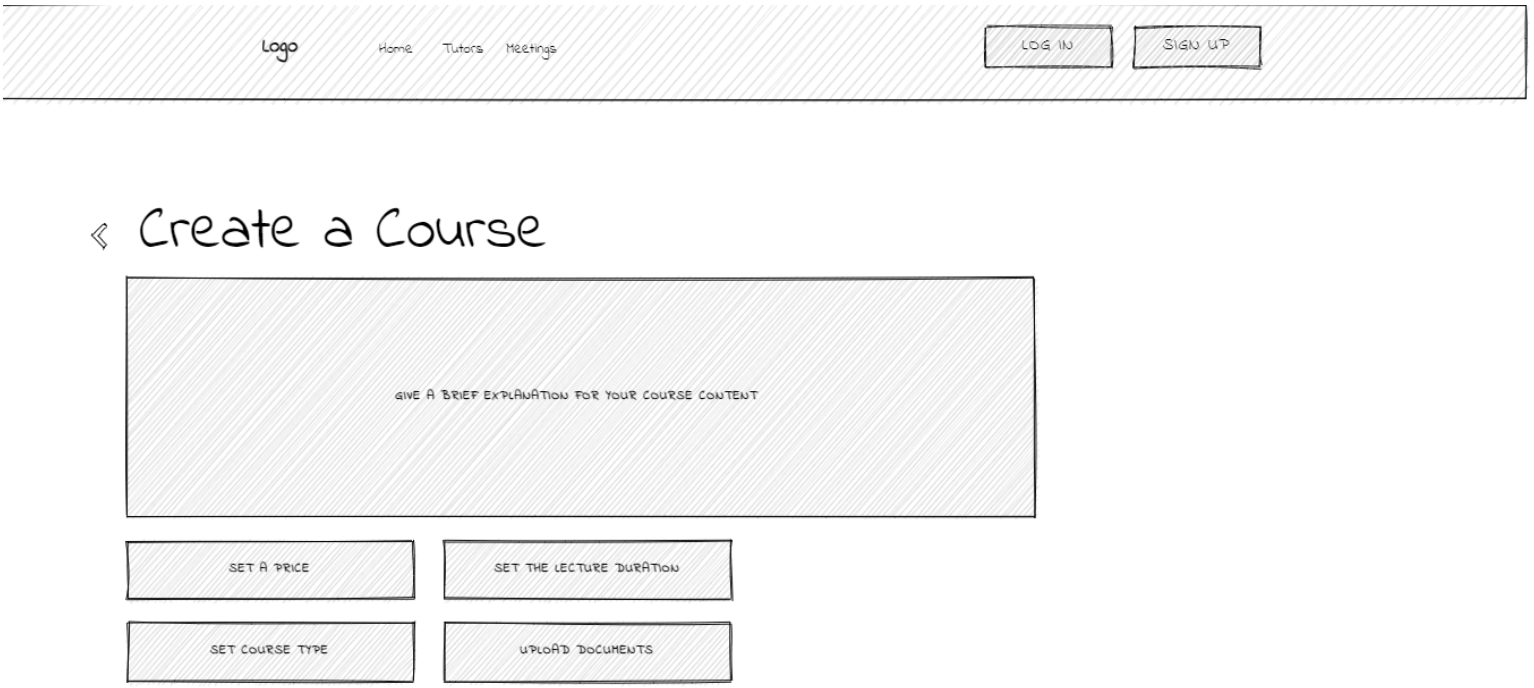

Figure 24: Course Addition Page

Users on this page will be asked to give information to create their course. They will be asked to give a description of their course, set a price and duration, and a type. The type will be either supported fields like YKS or LGS by the system or a custom field. The system automatically verifies the user's success on the uploaded document if the given field is not custom. Users must upload a document and verify their success to create a course on the non-custom field. By verifying, there will be a sign on the course elements that verify the tutor's proficiency in the field. If the field is selected as custom, uploading a document will not be necessary.

4.5 Meetings Page

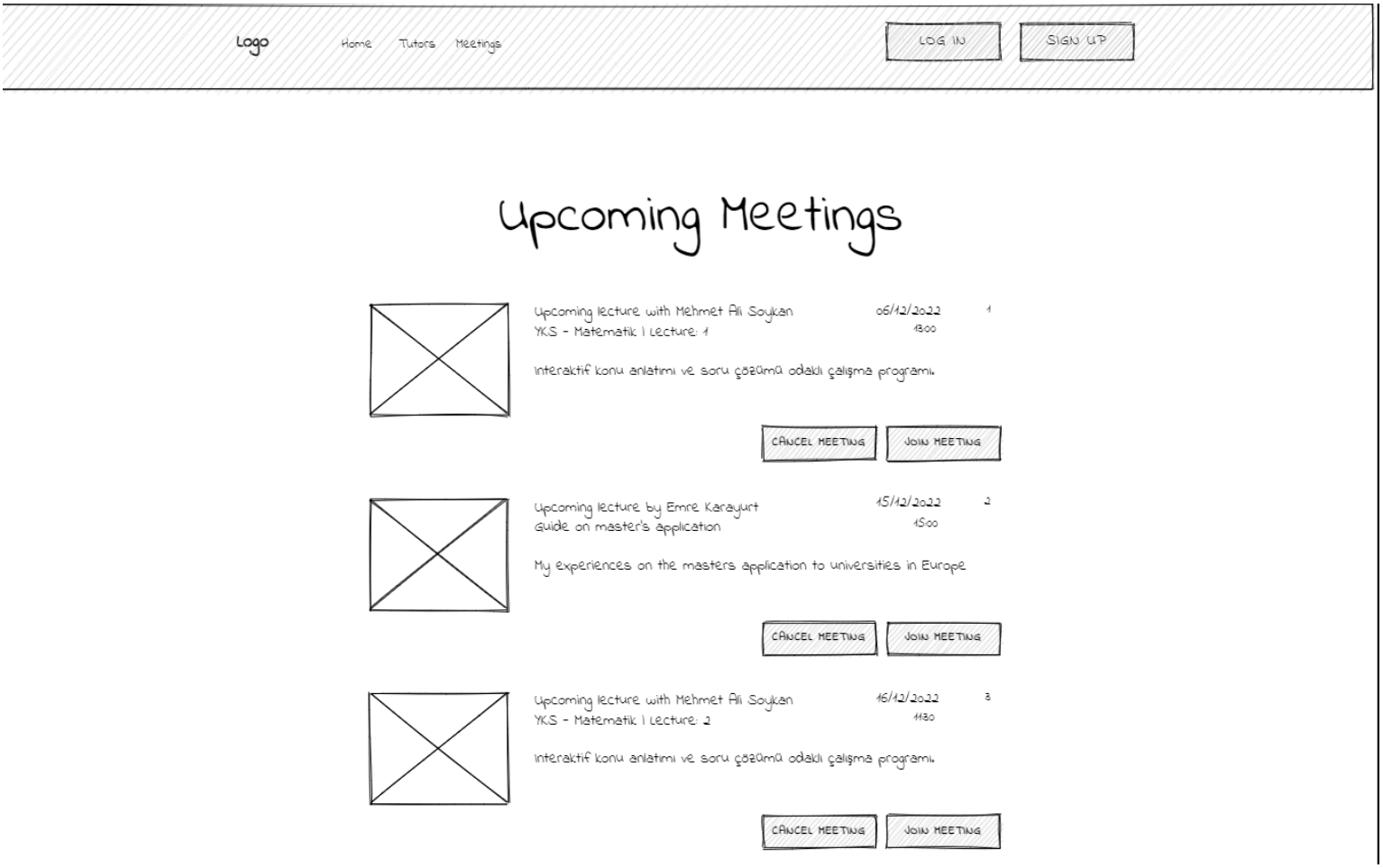

Figure 25: Meetings Page

Users can access this page by clicking “Meetings” on the navigation bar. This page will show the upcoming meetings of the user. Lectures given and taken by the user will be displayed here. Those will be differentiated from each other with colors. Users can see the partner’s name, date, time, and course description in each item. Users will be able to cancel or join the meetings using this panel.

4.6 Streaming Interface

4.6.1 Initial Mode

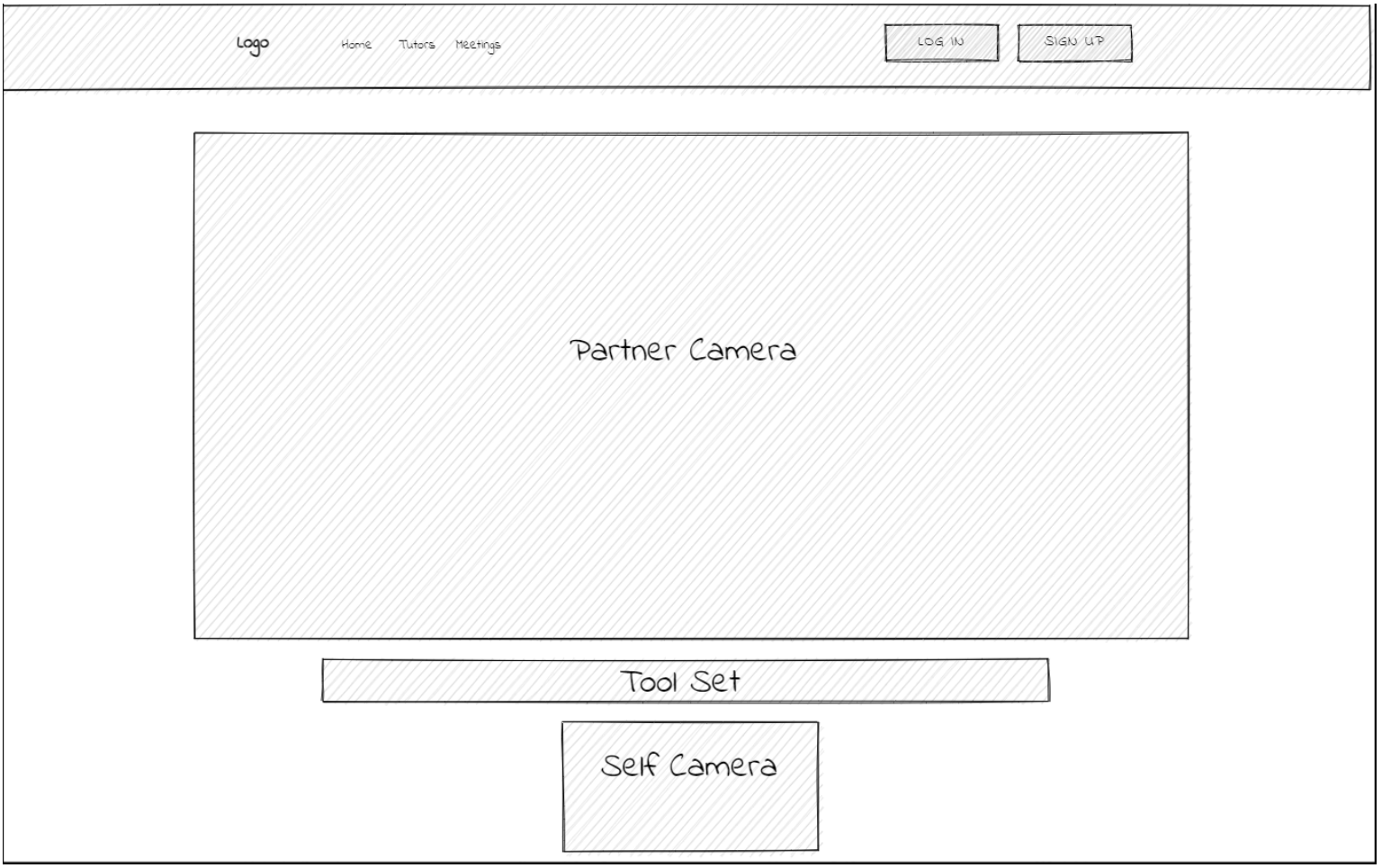

Figure 26: Streaming Interface

Users will be directed to the streaming page when clicking the join meeting button on the meetings page. Initially, users will see their camera on the full screen. In the above example, the display UI is provided for two users. From the tool's menu, users can disconnect, mute, and close their cameras. In addition, tutors can share their screens and use an interactive whiteboard. The tools menu can be expanded to support other functionalities that will be implemented later on.

4.6.2 One Screen Mode

Figure 27: One Screen Mode

If the tutor decides to share their screen or use an interactive board, users' cameras will be displayed at the bottom of the screen. The shared screen or the interactive board will be displayed on the upper part.

4.6.3 Two Screen Mode

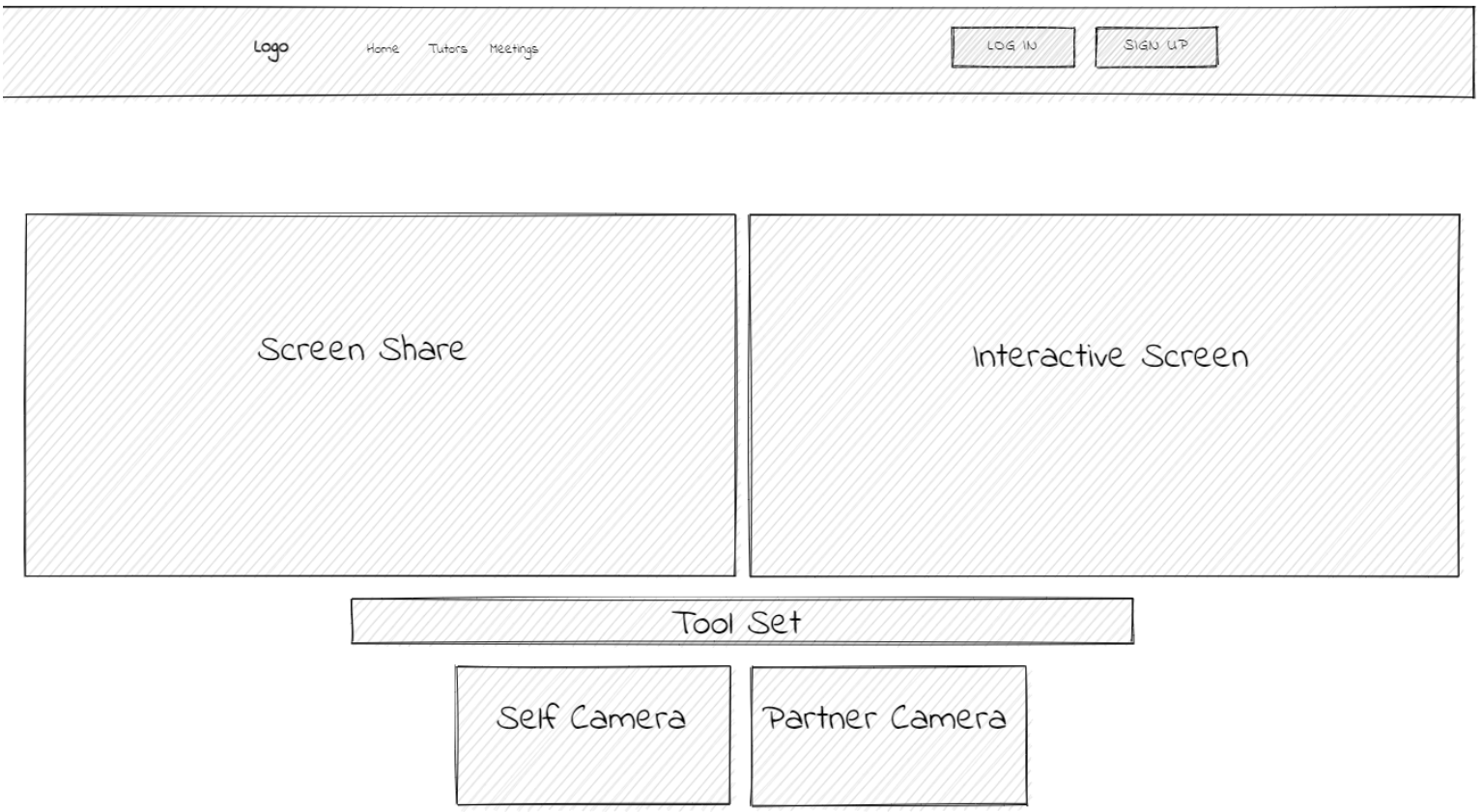

Figure 28: Two Screen Mode

Tutors can decide to use both the share screen and interactive board features at the same time. This will divide the upper part of the screen into two, which will display both features.

5. Other Analysis Elements

5.1 Consideration of Various Factors in Engineering Design

While designing the product, we made some judgments about different factors.

5.1.1 Economic Factors

While designing this product, it is intended to be a commercial one. By using this product, the ones with significant success can share their experiences with those in need and earn money. Besides, the owners of this product, us in this case, must get some proportion from this trade to maintain the product. Therefore, the design of the product should support this intention.

One of the main concerns was that even though tutors in this system had proven their successes, users may need more convincing to pay them to get their knowledge. So, we gave them a chance to convince users by giving them free courses. By doing this, they can increase their positive comments and easily convince other users for charged courses.

Another concern is that all the product users will trust us for their payment information and security. This leads us to two design concerns: securing all the users' payment information safely and ensuring that transactions between users happen as neatly as possible. For the security of payment information, information like credit card numbers shouldn't be accessible by others. For the integrity of money transactions between users, we have to decide what to do about situations like; the tutor does not come to the conference, the user/tutor wants to cancel the appointment, the conference can't be held because of technical issues, etc.

5.1.2 Social Factors

This product aims to be a bridge between tutors and the ones who want to gather knowledge from tutors’ experiences. This makes this product a social product, leading us to think about social factors. By forming this bridge, both sides benefited. Tutors earn money from his/her success and help others to benefit from his/her previous experiences. However, for both sides to benefit from this product optimally, we decided to add comments to the tutors. By doing this, users can easily evaluate

tutors according to their comments which makes the unpredictable side of the product more predictable.

5.1.3 Environmental Factors

Maintaining a server consumes energy which has a bad effect on the environment. Using peer-to-peer video streaming, Tutoryum solves this problem in addition to scalability and cost. Therefore, we only need to use the server to track which conferences are currently and previously held, information about the users, and course materials. This will decrease the server requirement by a noticeable amount.

5.2 Risks and Alternatives

5.2.1 Video Streaming

For the product, we are planning to make conferences peer-to-peer for scalability. However, making video streaming scalable is a general problem that was hard to solve. Therefore we can face unprecedented problems while we are trying to design the system.

We will design the system to work with available video conferencing applications. By doing this, we guaranteed that even though we couldn’t solve the scalability problem using peer-to-peer video streaming, we could use all other features we proposed with some available video streaming systems.

5.2.2 Saving Course Materials

We want tutors to be able to save their actions as course material after the course. We have two options here, the first one is saving them on the server, and the second one is on the tutor's PC. If we save it on the PC of the tutor, users cannot access that material anytime they want, so our main aim is to save it on the server. However, this emerges a new problem with scalability as the server needs to hold all the data, which may increase the server's cost. So, we are planning to use small-sized data such as the pencil's movements on the whiteboard to save the data instead of using files like png or jpg.

In the case of not successfully implementing this, we can always rely on the other two options mentioned: using the tutor's PC or saving it on a server with larger files.

5.3 Project Plan

In the first two weeks, Yusuf Miraç Uyar and Halil Özgür Demir will be working on the used version of the front end. They will finish the implementation of HTML and CSS. Barış Ogün Yörük and Oğuzhan Özçelik will be working on the streaming APIs. They will try to implement and test the viability of the APIs and discuss our options for implementing the streaming feature. Meanwhile, Çağrı Durgut will initialize the backend and database part. Between weeks two and six, depending on the assessed difficulty of implementing the streaming feature, people will be assigned to the frontend, backend, and implementation of the streaming feature. A prototype of the streaming feature and general course functionalities will be ready to use until week eight. After that, one week will be assigned for general testing and bug fixing. In addition, one week of work time will be reserved for people to work on their school assignments and work. Thus, it is planned to complete the project prototype in ten weeks.

5.4 Ensuring Proper Teamwork

Since Tutoryum is a rather big and complex project, proper teamwork should be ensured. Ideally, each person has responsibilities and specific tasks. We are a team of five people, and if each person contributes to the project equally, the project can be finished in approximately six months. Each person must complete their tasks without delay since delay might cause others to be delayed. Another important topic is how tasks should be divided; we discussed this in section 5.3

To assign tasks to five people, we will first assign each person to a specific part of the project. The project has four main components: Backend, Frontend, UI/UX design, and DevOps. According to each person's choice, we will divide according to the following structure:

Backend: Mustafa Çağrı Durgut, Barış Ogün Yörük, Yusuf Miraç Uyar

Frontend: Oğuzhan Özçelik, Halil Özgür Demir, Yusuf Miraç Uyar

Infrastructure: Barış Ogün Yörük

To ensure that each part of the project is done correctly and goes synchronously, underlined people will be responsible for reporting to the team about progress of each part. Now, tools that will be used to ensure proper teamwork will be discussed.

5.4.1 Asana

As a task management application, we will use Asana [4]. We will use the following features of Asana:

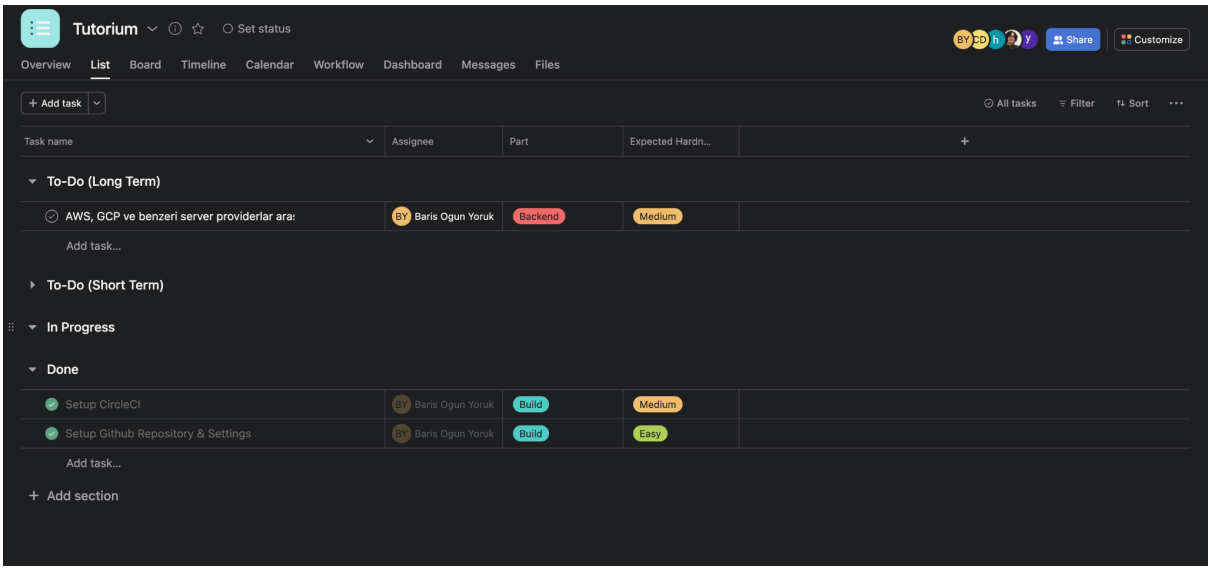

Figure 29: Asana Tasks as a List Page

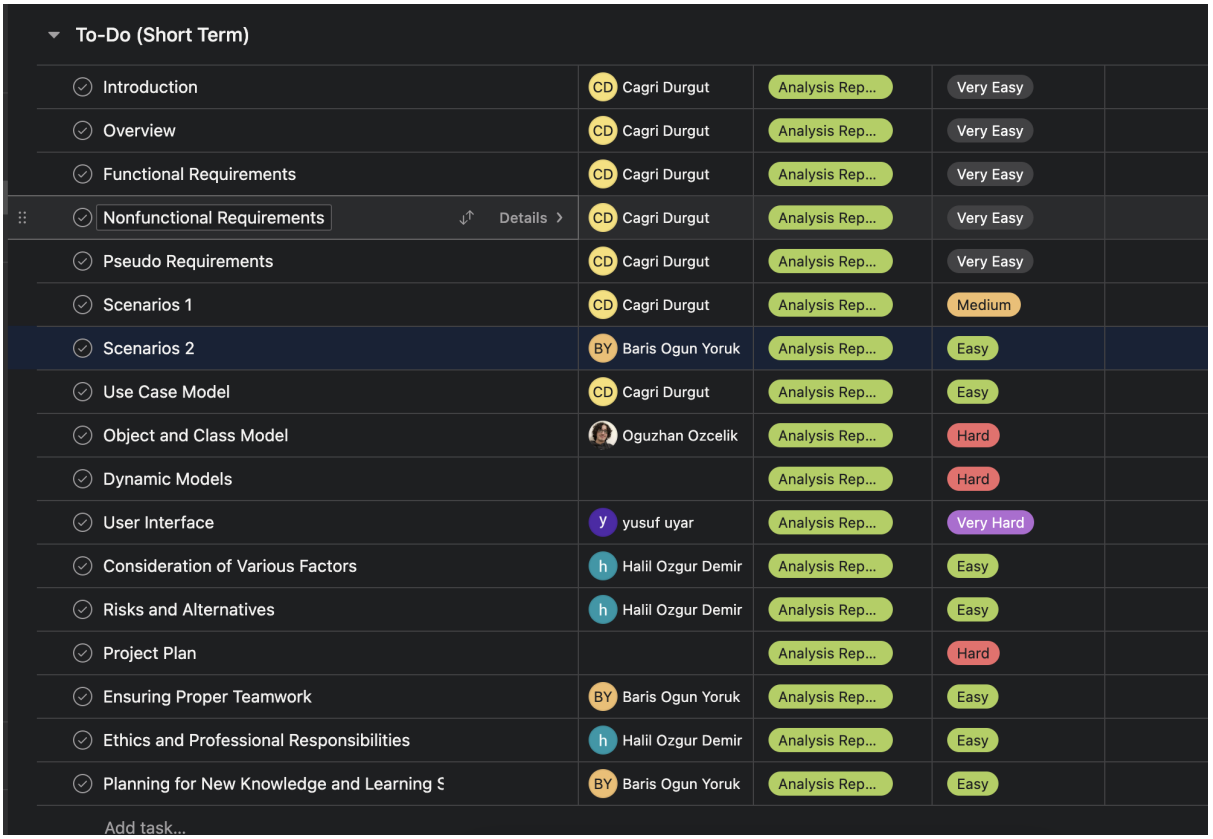

Figure 30: Asana Tasks in To-Do (Short Term) List

- We will have four sections to divide tasks.
    - To-Do (Long Term): In here, we will list all the tasks that should be done before finishing the project. But these tasks will not have a priority. Think of this section as idle tasks.
    - To-Do (Short Term): Same as Long Terms, but these tasks should be started immediately after finishing those in the section 'In Progress. These tasks have priority over Long Terms and should be considered when someone is taskless.
    - In Progress: These tasks are currently being implemented. Each person is responsible for having a task in Progress so that all team members can see which tasks are currently being implemented by a teammate.
    - Done: This section is for auditing reasons. With this section, it will be possible to keep track of each teammate's contribution and see the project's development.

- Additionally, we will have three different tags to differentiate between tasks for each task.
    - Assignee: This is the person who is responsible for a particular task.
    - Part: This indicates which part of the project is tasks belong to. Possible values are: Backend, Frontend, Build, and Report.
    - Expected Hardness: This tag will indicate the anticipated hardness of a particular task. It will be helpful to divide tasks between team members so that each person has the same effort. Note that this is expected hardness, meaning it is an estimation, not exact.

Thanks to Asana, we will have a SaaS that enables us to divide tasks and organize the project's development.

5.4.2 Discord

We will use Discord to communicate with the team and organize documents and artifacts [9].

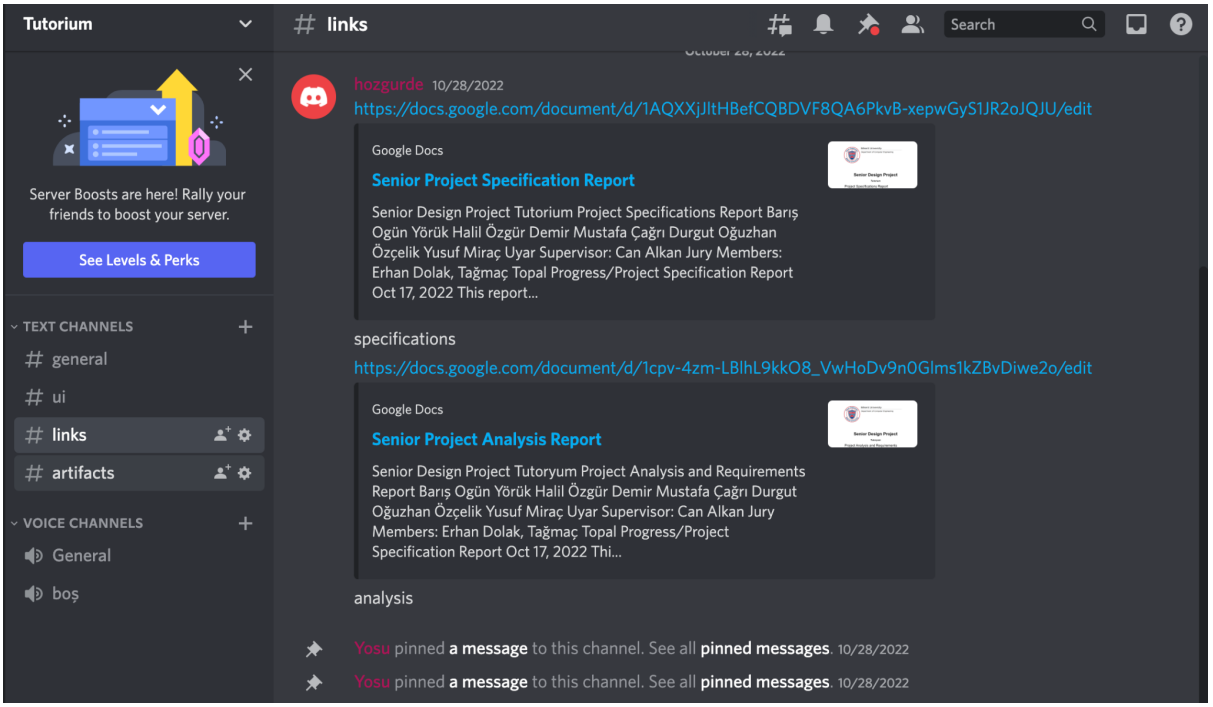

Figure 31: Discord Chat

- Voice Channels: We will use this feature for communicating in an online meeting with each other.

- Text Channels: We will use this feature for communicating via texting with each other. Additionally in these channels, we will upload some artifacts such as UI design ideas or links such as the URL of the analysis report.

Thanks to Discord, we plan to have transparent and open conversations with each other so that ambiguities in the project or different proposals will be resolved as soon as possible. Additionally, having dedicated artifacts and links channels will save us time.

5.4.2 GitHub

We will use GitHub as a remote repository service [2]. It will enable us to share code and manage our project's codebase. We will use several features of GitHub and Git to manage our codebase properly with contemporary techniques. These features will be utilized in the following way:

- Branching: Each person will create a separate branch for their tasks. For example, tasks of updating the readme.md will be in the following way: ‘update-readme.md.’ Note that there will be no strict requirement for branch naming. However, it is advised to use a name related to the task.
- Commit Messages: Each person will be strictly required to follow a commit message convention so that different people will understand what each commit is about. For example, a commit message for adding a user login page will be the following: ‘feat(frontend)/add-user-login-page.’ Here ‘feat’ indicates the reason for the commit, ‘frontend’ suggests the scope of the commit, and ‘add-user-login-page’ shows what the commit is about. Note that this convection will be checked in the CI/CD pipeline.

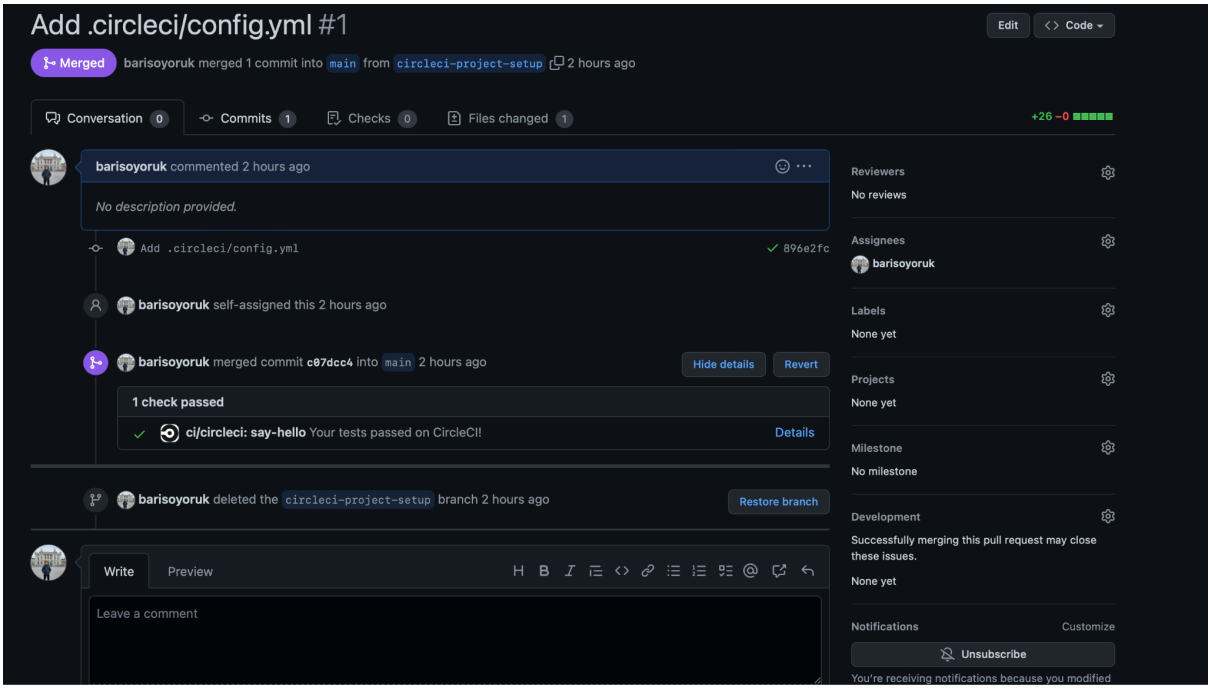

Figure 32: GitHub Pull Request

- **Pull Requests:** First, it will be forbidden to push the main branch directly. Each commit will need to be first pushed in a non-default branch. Then it will be required to create Pull Requests over GitHub to have a code review on each branch. This will ensure that each code merged to the Master will have a second look. Additionally, each commit will go over CI/CD pipeline to ensure that merging on the main branch will not cause bugs/errors.

Finally, GitHub will also be used for auditing purposes. It will be visible how many commits each person created, how many pull requests each person opened, and how many lines of code each person added. However, it should be noted that each task has varying difficulties. Some will require more research than others, and some may include many lines, but they are just boilerplate statements. These indicators should not be treated as the only indicator for ensuring equal teamwork.

5.5 Ethics and Professional Responsibilities

Tutoryum will be a bridge between people, and these people may have different ethnicities and sexualities. While accepting tutors to the system, it does not consider any of these characteristics of people. With this property, the Tutoryum will

be an inclusive environment. If one student who attends the tutor's conferences insults one of these tutor characteristics or vice versa, Tutoryum will have a report section. Any of these users may report these insults, and after a proper inspection, these users will be banned from the system. Moreover, this reporting system can also be used to resolve payment issues between tutors and users.

Another responsibility is the security of the personal information of tutors. As tutors will be uploading their formal exam documents to the system, we need to securely store them and delete them when they are no longer needed.

5.6 Planning for New Knowledge and Learning Strategies

Coverage of the Tutoryum project is extensive and requires knowledge about several tools and technologies. These tools and technologies are as follows:

- HTML
- CSS
- TypeScript
- Material UI
- React
- C#
- .NET
- Entity Core Framework
- PostgreSQL
- Docker
- CircleCI
- GCP
- WebRTC
- Computer Networks
- Software Project Management

Since each team member is assigned different parts of the project, they are required to know each tool used in their parts. From previous experiences, each team member knows some tools. While dividing team members into each part, prior knowledge was considered. However, of course, there are some missing parts, and

each team member is required to learn these missing parts before starting the implementation.

Some team members have known Computer Networks and Software Project Management since they took these courses in previous semesters. These members will be responsible explicitly for these concepts. However, if it is required for another team member who does not know these to learn them, people knowing these will help them out.

The rest of the tools and technologies can be learned online. To learn these, the following approaches will be used:

- Official Documentations
    - Tutorials: These help boost new learners. They usually tend to be precise and direct to the point. They help to give learners an overview of the tool. However, most of the time, they are insufficient.
    - Reference Pages: These include all the information regarding functions, classes, etc., inside the new tool. Of course, it is impractical to sit and read all these pages, but new learners can utilize them when they are stuck on a specific part of the tool.

- Online Courses: These resources help you learn new technologies in depth. Usually, these courses include all the required knowledge to develop a specific tool. Moreover, they often have exercises in coding along while learning, which is extremely useful to reinforce what is learned. Each team member who does not know a tool will be suggested to follow one of the online courses so that before starting the project, they will have enough information about the tool to develop with it.

6. Glossary

- API - Application Programming Interface
- CI/CD - Continuous Integration / Continuous Deployment
- P2P - Peer-to-peer
- PC - Personal Computer
- UI - User Interface
- URL - Uniform Resource Locator

7. References

[1] “About”. https://git-scm.com/about [Accessed: Oct 12, 2022]

[2] “Let’s Build from Here”. https://github.com/about [Accessed: Oct 12, 2022]

[3] “Winning Businesses Are Powered by Effective, Fast-Moving Software Teams”. https://circleci.com/about/ [Accessed: Oct 15, 2022]

[4] “We're in Business to Help You thrive”. https://asana.com/company [Accessed: Oct 15, 2022]

[5] “Secure Video Conferencing for Everyone”. https://meet.google.com/ [Accessed: Oct 12, 2022]

[6] “We Deliver Happiness”. https://explore.zoom.us/en/about/ [Accessed: Oct 14, 2022]

[7] “We Share Knowledge with The World”. https://about.udemy.com/ [Accessed: Oct 14, 2022]

[8] “Everything You Need to Know About the ‘Right to be Forgettent’”. https://gdpr.eu/right-to-be-forgotten/#:~:text=In%20Article%2017%2C%20the%20GDPR,originally%20collected%20or%20processed%20it. [Accessed: Nov 1, 2022]

[9] “Create Space for Everyoner to Find Belonging”. https://discord.com/company#:~:text=Discord%20is%20a%20voice%2C%20video,with%20their%20friends%20and%20communities. [Accessed: Oct 26, 2022]