1 Executive Summary:

TutorSwipe is an advanced platform designed to revolutionize how students connect with tutors, creating an engaging and well-organized learning environment. With the increasing demand for personalized education and the shift towards online learning, there is an essential need for a centralized platform that simplifies the process of finding, booking, and engaging with tutors. TutorSwipe meets this need by offering a user-friendly interface and comprehensive features, ensuring a safe and secure space for educational purposes.

Functions and Services:

TutorSwipe provides best-in-class features designed to meet the diverse needs of both students and tutors. Users can easily register, create detailed profiles, and browse through a list of tutors based on their expertise, teaching style, and experience level. The platform ensures that only approved tutors and verified content are accessible to users through strict content moderation. Once connected, students can book tutoring sessions, communicate with tutors via a built-in messaging system, and track their payment history. Students can also securely pay for tutoring sessions through the platform. Additionally, students have the flexibility to organize group bookings, promoting collaborative learning opportunities. Tutors can manage their availability, upload multimedia content to enhance their profiles, and receive student ratings and feedback.

Unique Features:

TutorSwipe stands out with its emphasis on security, quality, and user engagement. The platform introduces an innovative video-swiping functionality, allowing students to browse through tutor profiles effortlessly. This intuitive feature enhances the user experience by enabling quick and engaging interactions, allowing students to easily find and connect with tutors who best match their learning needs and preferences. Additionally, TutorSwipe incorporates gamification elements, rewarding students for their commitment and achievements.

About Our Team:

TutorSwipe is the brainchild of dedicated students from Fulda University of Applied Sciences. With our expertise and passion for innovation, we are committed to delivering an educational platform that empowers both students and tutors.

2 Personae and main Use Cases:

2.1 Personae

Sophia Bergmann (Visitor)

- **General characteristics:** Sophia is an individual who accesses our web application without creating an account or logging in. She is seeking information about the services offered.
- **Goals:** Sophia aims to gather enough information to decide whether to use the service and registering as a student or a tutor. She wants to understand the features and benefits of the platform.
- **Skills:** Basic internet navigation skills are a requirement for Sophia to explore the website. She should be able to browse through different pages and access information.
- **Pain points:** Sophia may be frustrated if she cannot find the information she needs easily and fast.

David Wagner (Student)

- **General characteristics:** David is a individual who searches for tutoring services for himself. Needing academic support.
- **Goals:** David aims to find tutors who meet his specific learning needs. He wants to easily search, compare, and connect with them.
- **Skills:** David should possess the same basic internet skills as a visitor to navigate the web application. In addition, he also needs to communicate effectively to convey his learning goals and connect with tutors.
- **Pain points:** David may struggle to find tutors who meet his scheduling preferences, subject requirements, or budget. He also might face problems in judging the quality and reliability of tutors based on limited information.

Felix Richter (Tutor)

- **General characteristics:** Felix is a individual offering their knowledge and teaching services through the web application. He comes from a computer science background (bachelor) and possess high levels of experience and is currently a master student.
- **Goals:** Felix aims to attract students by showcasing his qualifications, expanding his student base, improving his teaching skills, and earning money.
- **Skills:** Felix should have the same basic internet skills as the student and the visitor, with additional expertise in his respective subjects. He also needs to communicate effectively, manage his time schedule, and be able to teach different students.

- **Pain points:** Felix may face challenges in marketing himself to attract students and managing administrative tasks like scheduling sessions and tracking payments.

Emma Berger (Administrator)

- **General characteristics:** Emma is responsible for overseeing the operation and maintenance of the service. She enforces policies, manages user accounts, and handles any technical issues.
- **Goals:** Emma aims to maintain the integrity and security of the platform while providing support to users. She needs to verify the credentials of tutors and all uploaded media to address any violations of the platform's terms of service that may occur.
- **Skills:** Emma requires technical proficiency to navigate the administrative part of the web application. She should also possess strong organizational and communication skills to interact with users effectively.
- **Pain points:** Emma may find it challenging to handle a large volume of user activities, verifies the authenticity of documents submitted by tutors, and enforces platform policies consistently.

2.2 Use Cases

- **UC-01:** Sophia explores the Tutoring Service as a Visitor
  Persona: Visitor
  Short description: Sophia, as a visitor, navigates the tutoring platform to gather information and evaluate its features. She explores various sections to understand the platform's offerings and functionality.
  Main Scenario:

  1. Sophia uses the Visitor Access to explore the platform.
  2. She explores the Privacy Policy, About Page, and Imprint to get additional information about the platform.
  3. Sophia uses the video swiping feature (FR-04) and the search and filter function (FR-13) to search for potential tutors.
  4. Sophia decides to register as a student (FR-01) to take advantage of the tutoring services, or as a tutor to offer tutoring services herself.

  Outcome: Sophia has a good understanding of the platform's features and decides to register as a student.

- **UC-02:** David registers on the platform
  Persona: Student
  Short description: David registers on the platform for the first time and creates a profile.
  Main Scenario:

  1. David successfully registers on the platform (FR-01, FR-02), and receives a short onboarding tour (FR-06) to familiarize himself with the different areas and functions of the platform.
  2. He then navigates to his profile to enter further information about himself or to make further settings. (FR-03)

  Outcome: David has an active account and a complete profile, ready to use the platform's tutoring services.

- **UC-03:** David searches for a Tutor and books a Lesson
  Persona: Student
  Short description: David searches for a tutor using the platform's features and books a tutoring session that fits his needs.
  Main Scenario:

  1. First, he uses the video swiping feature (FR-04) to quickly and easily search for tutors.
  2. David swipes through the introductory videos of various tutors.
  3. The first three tutors don't appeal to him. So he swipes them to the left.
  4. David likes the fourth suggested tutor and wants to book a tutoring session.
  5. David clicks on "Book Class" and selects a suitable time slot.
  6. After successfully booking a tutoring session, David can export his meetings as a calendar (FR-11).
  7. David receives rewards and experience after successfully completing tutoring sessions (FR-17). With the experience collected, he rises in level and can compete with other students on the leaderboard (FR-18).

  Alternative scenario:

  1.1 David can also search for a specific tutor. To accomplish this, he uses the search and filter function (FR-13) to define his preferences for tutoring, e.g. subject area, skill level, price, and location.

  3.1 David can also book classrooms with other students from the suggestion list (FR-20).

Outcome: David has successfully booked a tutoring session, earning rewards and experience points. His progress is tracked through the gamification system.

- **UC-04:** David rates a Tutor
  Persona: Student
  Short description: David rates the tutor based on his experience to provide feedback and help other students make informed decisions.
  Main Scenario:

  1. David navigates to the teacher page of the teacher he wants to rate.
  2. Then he clicks on the "Rate Tutor" button in the rating section.
  3. David rates the tutor on a scale of 1 to 5 and writes a comment (FR-14).
  4. He then clicks on the "Submit Rating" button to send the rating.

  Outcome: David has successfully submitted a rating for the tutor.

- **UC-05:** Felix Sets up the tutor page and connects with a Student
  Persona: Tutor
  Short description: Felix sets up his tutor page on the platform, provides detailed information about his skills and availability, and connects with a student to offer his tutoring services.
  Main Scenario:

  1. Felix starts by creating his profile page. To achieve this, he uploads an introduction video of himself and fills his page with pictures and information. (FR-12)
  2. Felix is then classified according to his skills and level of skills on the information he provides. This gives him the opportunity to stand out, as this information is visible to the students. (FR-19)
  3. He then enters the time slots in which he would like to offer tutoring. (FR-10)
  4. Felix uses the messaging system to get in touch with a student and communicates about requests, wishes etc. (FR-09)
  5. After the student has booked a tutoring session, Felix receives payment via the payment system. (FR-07)
  6. Felix can view the payments he has received in the payment history and thus always has an overview of his finances. (FR-08)

  Outcome: Felix has successfully set up his tutor profile, communicated with a student, and provided tutoring services. He has received payment and can track his earnings through the payment history.

- **UC-06:** Emma approves the content of the platform
  Persona: Administrator
  Short description: Emma, an administrator, reviews and approves content submitted by tutors and clients to ensure it meets platform guidelines before it is published.
  Main Scenario:

  1. Emma must first approve the content before it appears on the platform. She looks at the tutor profiles and approves them if they comply with the guidelines.
  2. Emma also checks all media that is published on the platform by the clients. (FR-16)

  Outcome: Emma has successfully reviewed and approved content, ensuring that all tutor profiles and media comply with the platform's guidelines before being published.

- **UC-07:** Emma handles the reported users
  Persona: Administrator
  Short description: Emma, an administrator, reviews and takes action on users who have been reported for violations or inappropriate behavior on the platform.
  Main Scenario:

  1. Emma takes care of the reported users. For this purpose, she has a separate page on which she can view all reported users.
  2. Emma navigates to her Administrator page. There she looks at the individual cases and then decides on the reported user. (FR-15)

  Outcome: Emma has successfully reviewed the reported users and taken appropriate action, ensuring that the platform remains a safe and compliant environment for all users.

3 List of non-functional requirements:

Requirement ID: NF-01
Title: Development and Deployment Standards
Description: Application shall be developed, tested and deployed using tools and servers approved by Class CTO and as agreed in Milestone 0. Application delivery shall be from chosen cloud server.

Requirement ID: NF-02
Title: Browser Compatibility
Description: Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of two major browsers.

Requirement ID: NF-03
Title: Mobile Responsiveness
Description: All or selected application functions must render well on mobile devices.

Requirement ID: NF-04
Title: Data Storage Specifications
Description: Data shall be stored in the database on the team's deployment cloud server.

Requirement ID: NF-05
Title: User Load Management
Description: No more than 50 concurrent users shall be accessing the application at any time.

Requirement ID: NF-06
Title: Privacy Policy Compliance
Description: Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.

Requirement ID: NF-07
Title: Language Specification
Description: The language used shall be English (no localization needed).

Requirement ID: NF-08
Title: User Experience Design
Description: Application shall be very easy to use and intuitive.

Requirement ID: NF-09
Title: Architecture Pattern Adherence
Description: Application should follow established architecture patterns.

Requirement ID: NF-10
Title: Code and Repository Maintenance
Description: Application code and its repository shall be easy to inspect and maintain.

Requirement ID: NF-11
Title: Use of Analytics
Description: Google analytics shall be used (optional for Fulda teams).

Requirement ID: NF-12
Title: Email Client Prohibition
Description: No e-mail clients shall be allowed.

Requirement ID: NF-13
Title: Payment Functionality Prohibition
Description: Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI.

Requirement ID: NF-14
Title: Application Security Practices
Description: Site security: basic best practices shall be applied (as covered in the class) for main data items.

Requirement ID: NF-15
Title: Media Integration
Description: Application shall be media rich (images, video etc.). Media formats shall be standard as used in the market today.

Requirement ID: NF-16
Title: Software Development Practices
Description: Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development.

Requirement ID: NF-17
Title: Code Management via GitHub
Description: For code development and management, as well as documentation like formal milestones required in the class, each team shall use their own GitHub to be set-up by class instructors and started by each team during Milestone 0.

Requirement ID: NF-18
Title: Application Branding
Description: The application UI (WWW and mobile) shall prominently display the following exact text on all pages "Fulda University of Applied Sciences Software Engineering Project, Fall 2021 For Demonstration Only" at the top of the WWW page. (Important so as to not confuse this with a real application).

4 Competitive analysis:

Features	**TutorSwipe**	Preply	Tutor.com	Wyzant
Search and Filter	**Yes**	Yes	Yes	Yes
Gamification	**Yes**	Yes	Limited	No
Earn Rewards	**Yes**	Limited	Limited	No
Manage Media	**Yes**	Yes	Limited	No
Video Swiping	**Yes**	No	No	No
Download Meeting Schedule	**Yes**	No	No	No

Planned Advantage:

TutorSwipe's standout feature is its video swiping function, which allows users to find their favorite tutor in a manner reminiscent of Tinder and TikTok. Tailored specifically for the Fulda University community, TutorSwipe enables students to locate tutors through personalized filters and arrange tutoring sessions. The platform boasts robust content moderation and a reward system, ensuring a secure and motivating environment that some competitors lack. Additionally, tutors and users can download their meeting schedules for optimal organization.

5 High-level system architecture and technologies used:

5.1 Software Stack

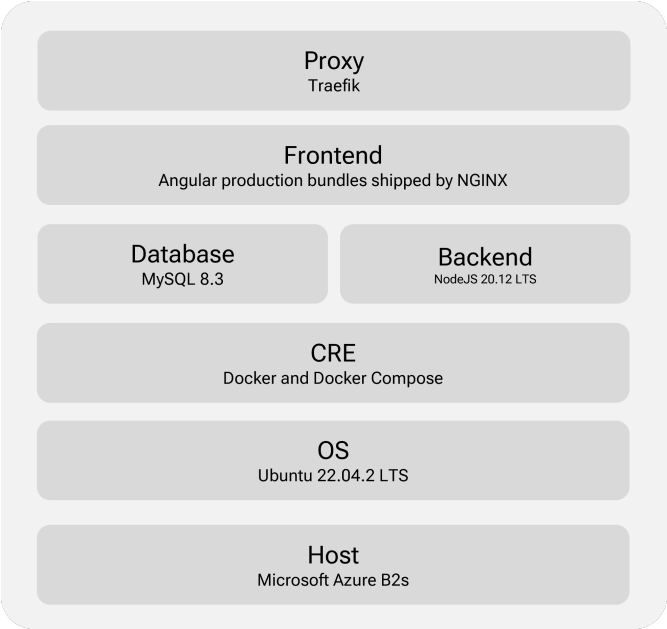


Figure 1: High-level Software Stack

5.2 Software Architecture

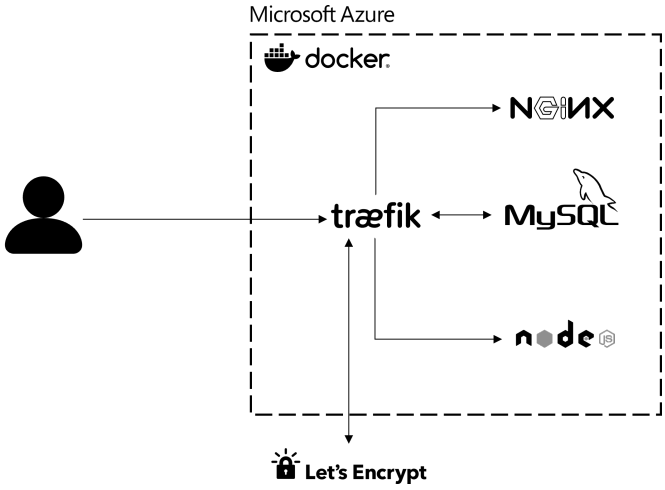


Figure 2: Architecture

8 Functional Requirements - prioritized:

Priority 1 Requirements

Unregistered Users

Requirement ID: FR-04.1
Title: Browse Tutors with Video Swiping
Description: Provides users with a visually engaging interface to discover tutors by swiping through video profiles. Users can view short introductory videos of tutors, swipe left or right to express interest, and access detailed profiles for more information, facilitating an interactive and intuitive browsing experience.

Requirement ID: FR-13.1
Title: Search and Filter
Description: Users can perform advanced searches and apply filters to narrow down tutor options based on specific criteria, such as subject expertise, availability, and pricing.

Registered Users

Requirement ID: FR-01.1
Title: Register Users
Description: Allows new users to create an account on the platform, providing necessary information such as username, email address, password, etc. User can register as a Student or as a Tutor.

Requirement ID: FR-02.1
Title: Log In User
Description: Allows registered users to log in to their accounts. Upon successful authentication, users gain access to their profiles, settings, and other platform features.

Requirement ID: FR-03.1
Title: Manage User Profiles
Description: Allows registered users to update and manage their profiles, including personal information, preferences, and settings. Profiles can differ depending on user type.

Requirement ID: FR-09.2
Title: Send Messages
Description: Users can communicate with each other through text-based messages within the platform. This feature allows direct interaction between students and tutors, enabling them to discuss tutoring arrangements, ask questions, and exchange information.

Requirement ID: FR-12.1
Title: Manage Media
Description: Users can upload, organize, and manage media files within their profiles or related to tutoring sessions. This feature supports various media types, including images, documents, and videos.

Administrators

Requirement ID: FR-16.1
Title: Approve Content
Description: Platform administrators can review and approve user-generated content, such as tutor profiles, media uploads, and educational materials, before they become publicly accessible. This feature ensures content quality and compliance with platform guidelines and standards.

Priority 2 Requirements

Registered Users

Requirement ID: FR-07.2
Title: Send and Receive Payments (mock)
Description: Allows users to send and receive payments within the platform for services rendered, such as tutoring sessions.

Requirement ID: FR-08.2
Title: Show Payment History
Description: Users can view their complete payment history, including transaction details, dates, and amounts. This feature allows users to track their financial activities and monitor transactions related to platform usage.

Requirement ID: FR-10.2
Title: Schedule Meetings
Description: Tutoring sessions or meetings between students and tutors through the platform's scheduling system. This feature allows users to set preferred dates, times, streamlining the process of arranging and managing appointments.

Requirement ID: FR-14.2
Title: Rate Users
Description: After completing tutoring sessions, students can rate and provide feedback on tutor's performance and professionalism. This feature helps maintain quality standards, builds trust within the platform users, and informs future students' decisions when selecting tutors.

Requirement ID: FR-15.2
Title: Report Users
Description: Users can report inappropriate behavior, violations of terms of service, or other concerns regarding fellow users. This feature initiates a review process by platform administrators to address reported issues.

Requirement ID: FR-19.2
Title: Classify Tutors
Description: Tutors are classified or categorized based on various attributes such as subject expertise, teaching style, experience level, and student feedback. This classification system helps students to easily find and select tutors who best match their learning needs and preferences. Additionally, it helps tutors showcase their specialities and stand out within the platform, facilitating more effective tutor-student matching.

Priority 3 Requirements

Registered Users

Requirement ID: FR-05.3
Title: Agree to Terms and Conditions
Description: Before accessing the platform's features, users must agree to the terms and conditions outlining the rules, rights, and responsibilities controlling platform usage.

Requirement ID: FR-06.3
Title: Onboard Users
Description: New users complete an onboarding process to set up their profiles.

Requirement ID: FR-11.3
Title: Download Meetings Schedules
Description: Allows users to download meeting schedules as an importable file type.

Requirement ID: FR-17.3
Title: Earn Rewards
Description: Students can earn rewards for various actions and achievements within the platform, such as completing tutoring sessions, or participating in platform events. These rewards can include virtual badges and points.

Requirement ID: FR-18.3
Title: Gamify User Experience
Description: Integrates gamification elements into the platform to enhance user engagement and motivation. Gamification mechanics such as points, levels, challenges, and leaderboards are implemented to make the learning and tutoring experience more interactive and enjoyable.

Requirement ID: FR-20.3
Title: Organize Classrooms
Description: Allows students to arrange tutor sessions for multiple individuals simultaneously.

9 List of main data items and entities:

9.1 Data Items and Entities

- **User** (see User Types and Privileges for details)
- **Lesson**: Is a time slot offered by a tutor and can be booked by a student. The knowledge is taught during this slot.
- **Classroom**: A virtual room where one tutor can teach multiple students.
- **Rating**: A student or tutor can receive ratings. Ratings can be a factor in a student's decision to choose a tutor.
  - **Comment**: A text describing the experience with the evaluated person.
  - **Stars (1-5)**: A rating score for the evaluated person. 1 is the worst and 5 the best mark.
- **Media**: A media file that can be uploaded and edited by the users of the application. A use case can be offering an introduction of a tutor or presenting a learning plan.
  - **Document**: A document file like PDF or Excel.
  - **Video**: A video file like MP4.
  - **Image**: An image like JPG or PNG.
- **Admin Dashboard**: The visual UI for administrators where they can manage the application. Detailed tasks are described in User Types and Privileges. This dashboard is only accessible to authenticated administrators.

9.2 User Types and Privileges

- **Administrator**: Administrator of the platform that manages the approval of new tutors, content, and policies.
  - All permissions
- **Visitor**: A visitor of the website that is interested in becoming a client or tutor.
  - Registration
  - Filter Tutors
- **Tutor**: A person who has been checked by one of the administrators and is approved. The state of the tutor can be verified (e.g. after being a tutor for one year and after checking the police clearance certificate).
  - Login
  - Receive/write messages
  - Update profile
  - Add/edit lessons to his calendar
  - Write ratings to students

  - Write a (bug) reporting
  - Receive badges
  - Create new classroom
  - Invite student to a classroom
- **Student**: A person who is registered on the platform and is not an administrator.
  - Login
  - Receive/write messages
  - Update profile
  - Book free demo lesson
  - Update his calendar with free time schedules
  - Book lesson
  - Write ratings to tutors
  - Write a (bug) reporting
  - Receive badges
  - Join classroom by invitation

9.3 Data Structure Overview

Lessons are the core, where tutors share knowledge and students absorb it. Classrooms foster group learning dynamics, encouraging collaboration and discussion. Feedback through assessments guides future interactions and helps users make informed decisions. Multimedia resources enrich lessons by providing supplementary material and visual aids. Administrators manage the platform through the Admin Dashboard, ensuring smooth operation and compliance with standards.

10 UI Mockups and Storyboards:

10.1 UC-01: Visitor Exploration

Sophia uses the Visitor Access to enter platform's landing page, navigates to video swiping (FR-04) and search filter (FR-13) functionality. Sophia can register as student or as tutor (FR-01). Sophia can see additional information in About Us, Privacy Policy and Imprint pages.

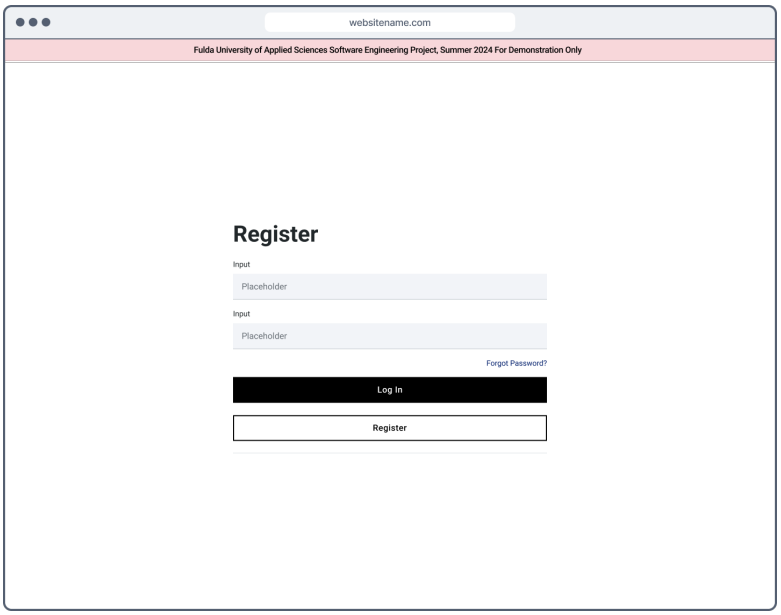


Figure 3: Registration

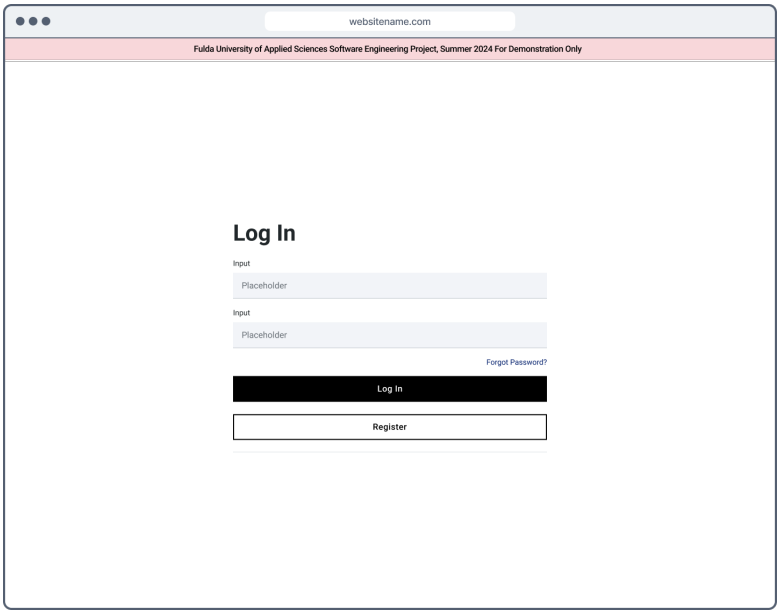


Figure 4: Login

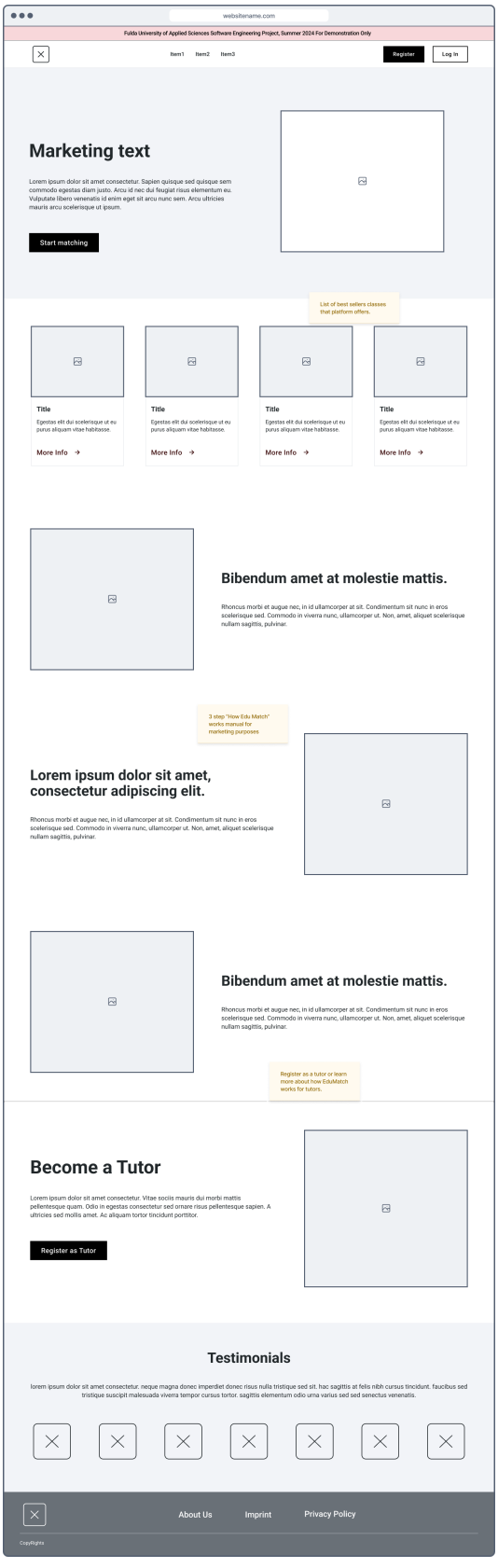


Figure 5: Landing page

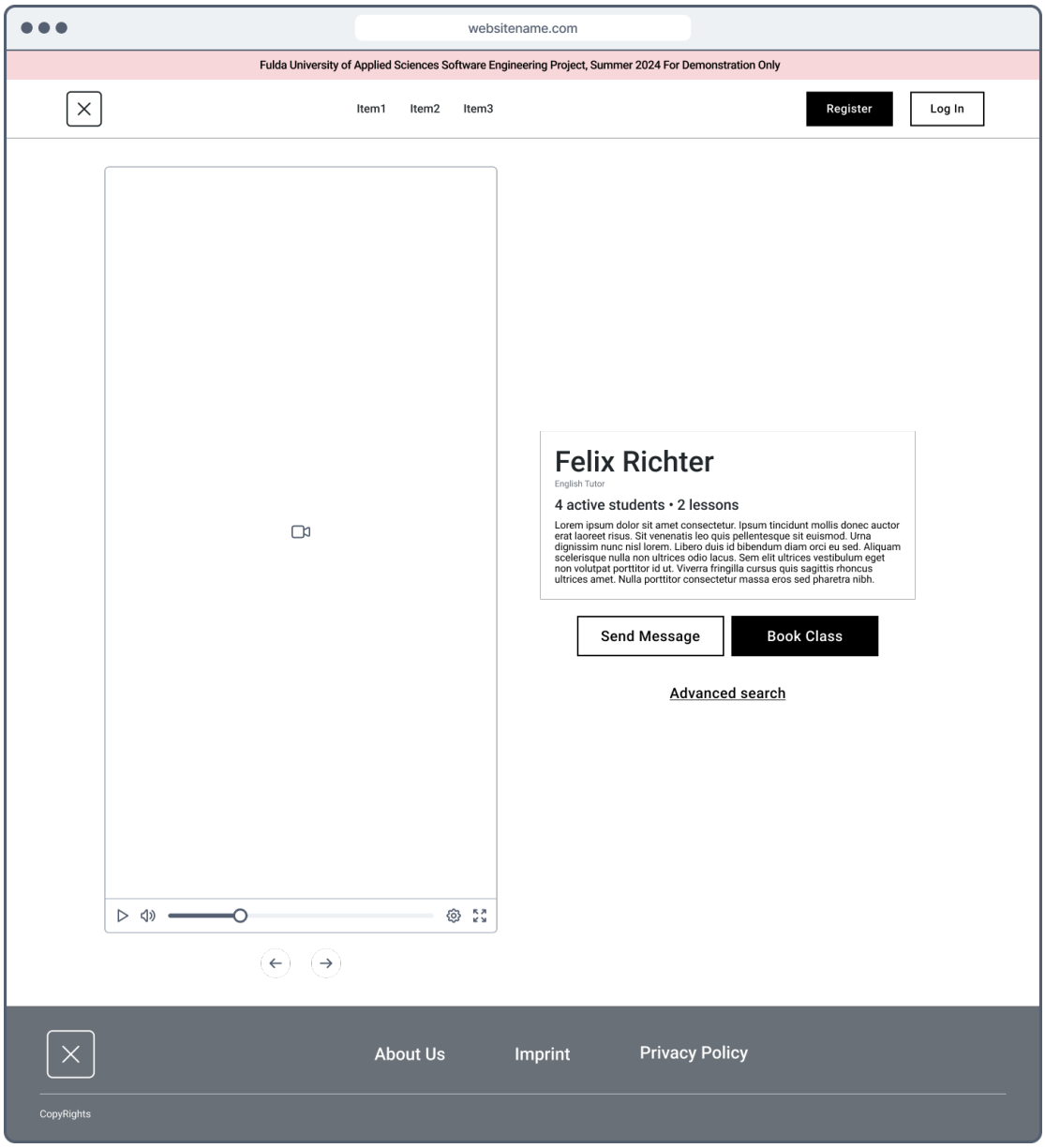


Figure 6: Tutor Swiping

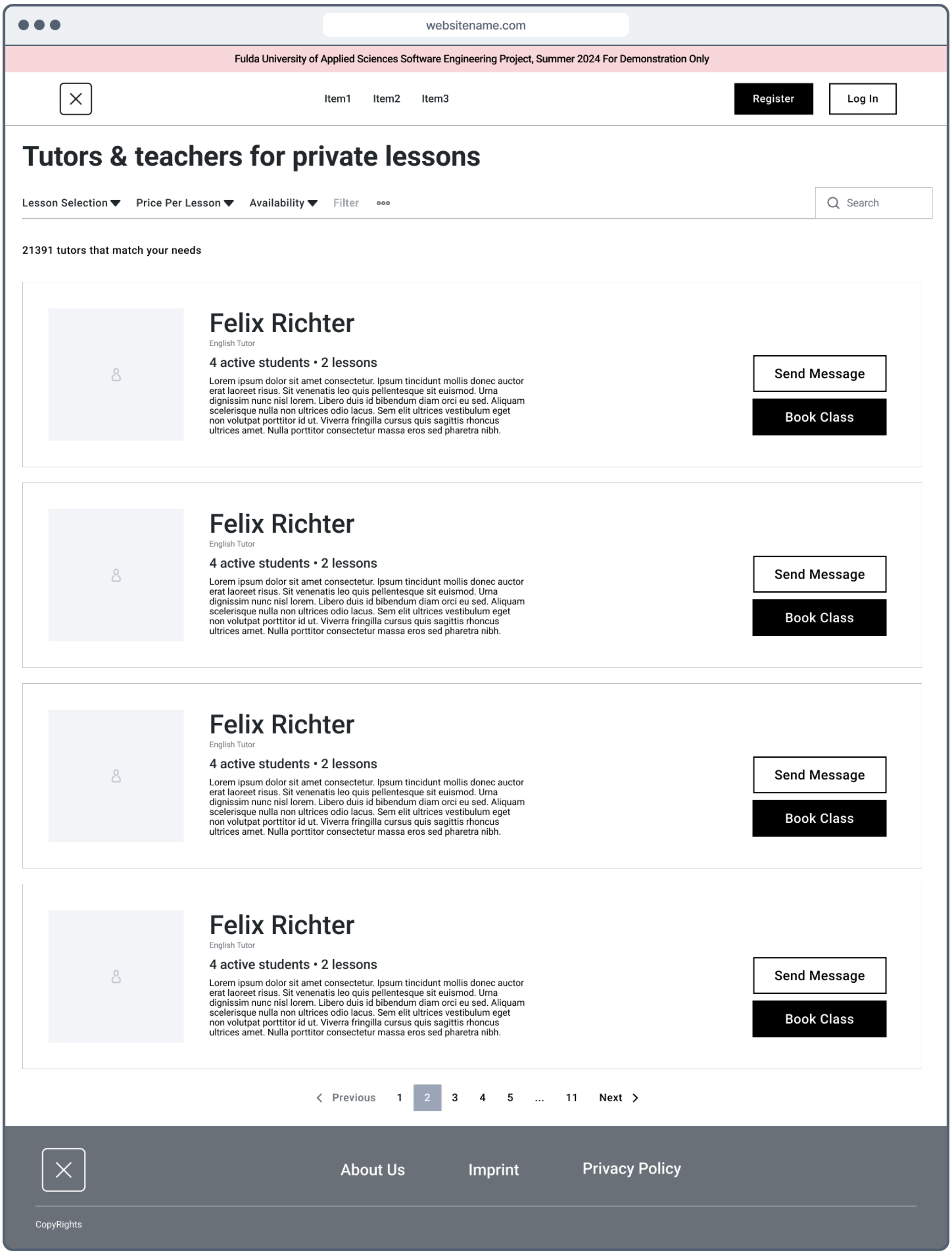


Figure 7: Tutor Search and Filter

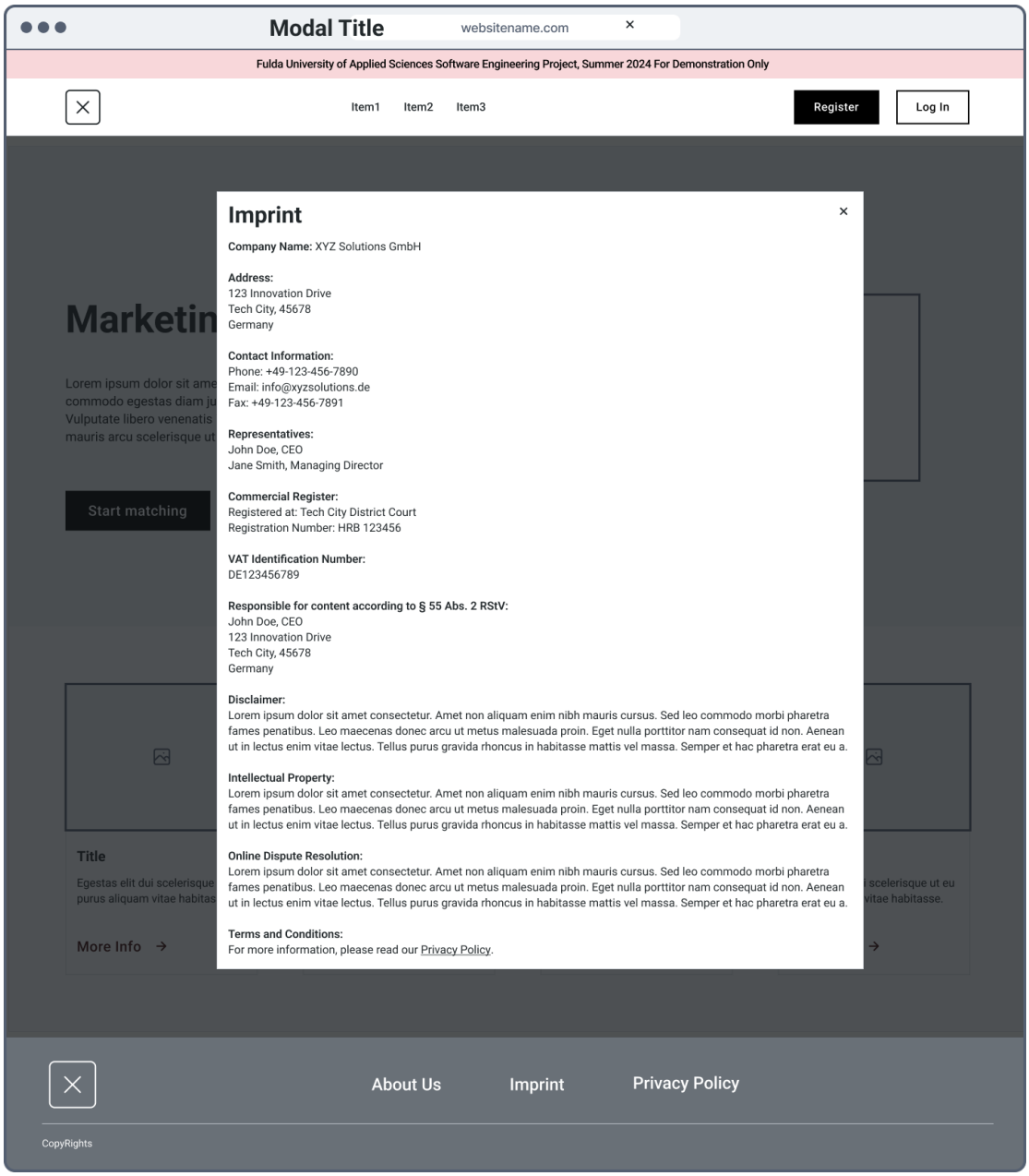


Figure 8: Imprint

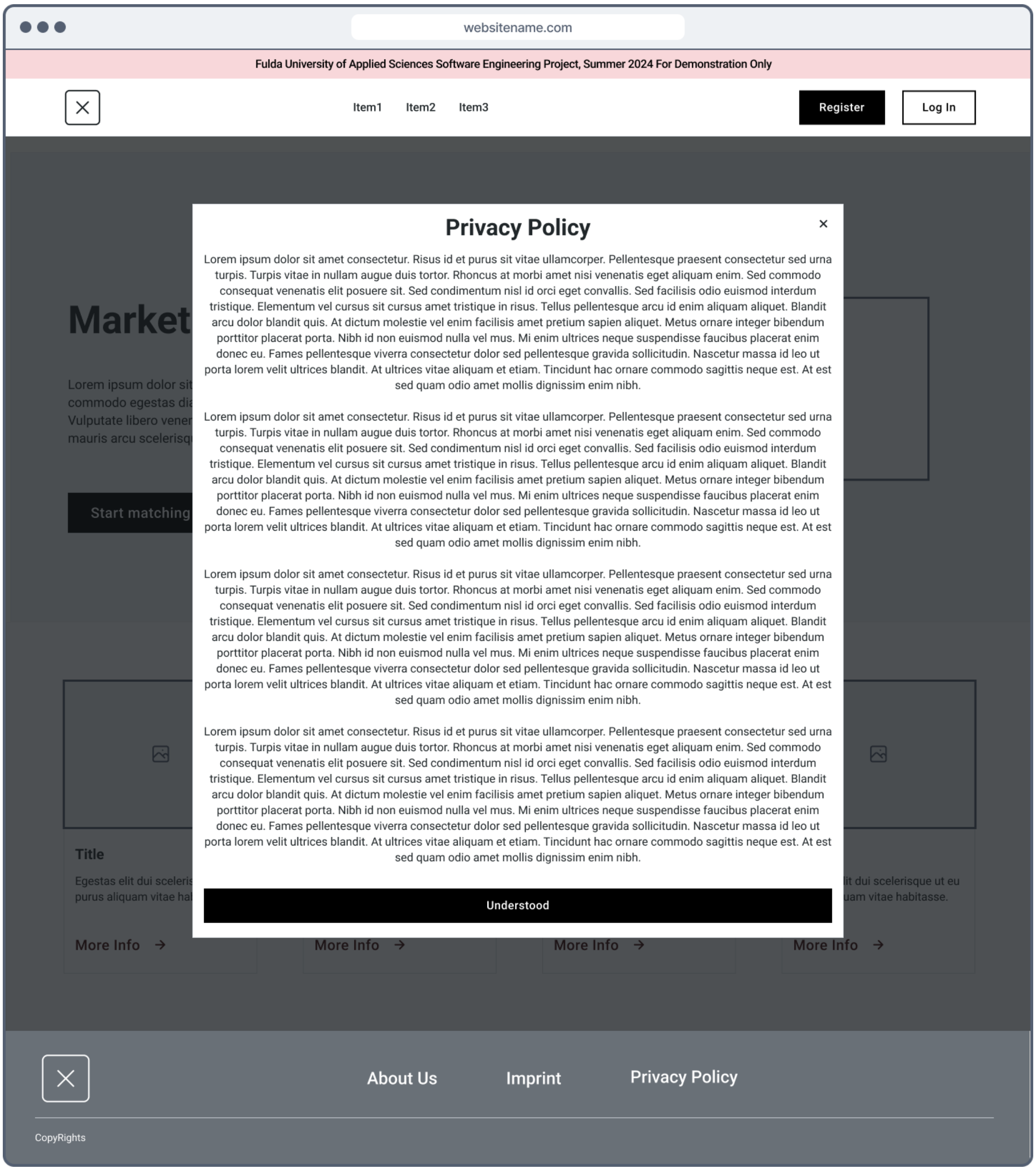


Figure 9: Privacy Policy

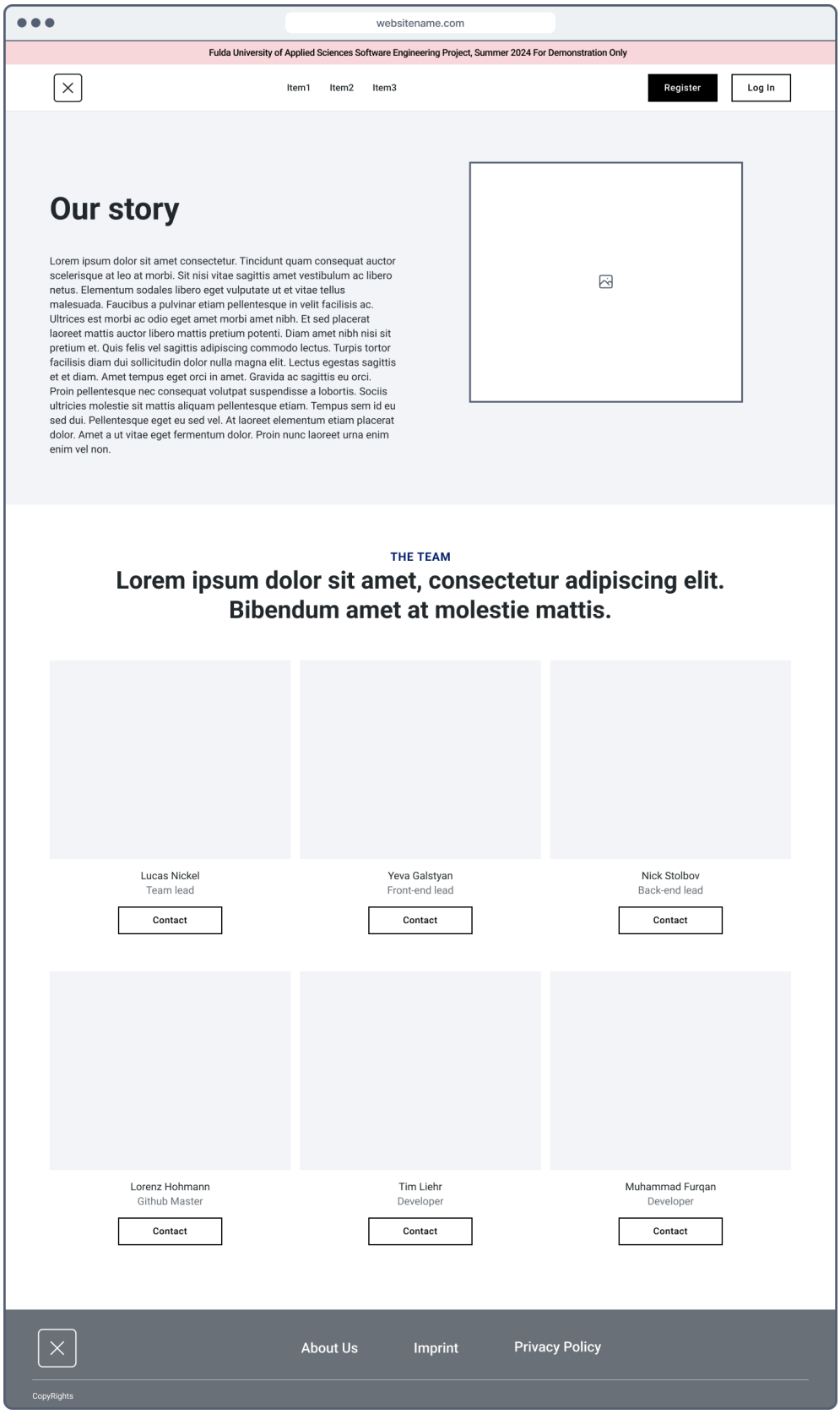


Figure 10: About us

10.2 UC-02: Registration

David registers successfully to the platform as a student.

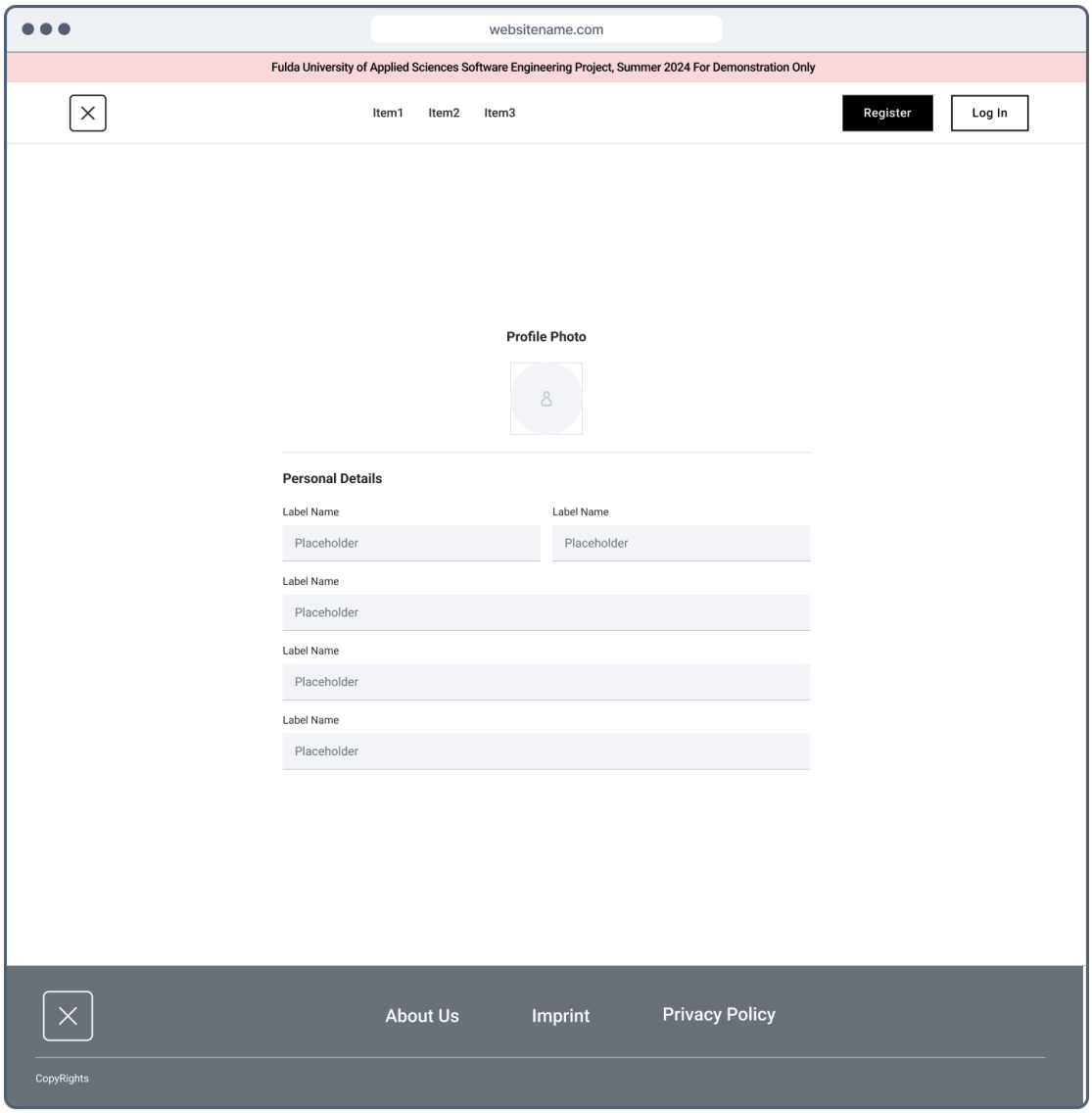


Figure 11: Fill Student Profile

10.3 UC-03: Tutor Search

David can use video swiping (FR-04) and tutor searching (FR-13) as a student. David can navigate to the page of a specific tutor and if the tutor is suitable, he can book a tutoring session and export his meetings calendar (FR-11). In addition David can check his profile to see booking history, calendar and rewards he gain. With his account David can also use functionality of becoming a Tutor himself.

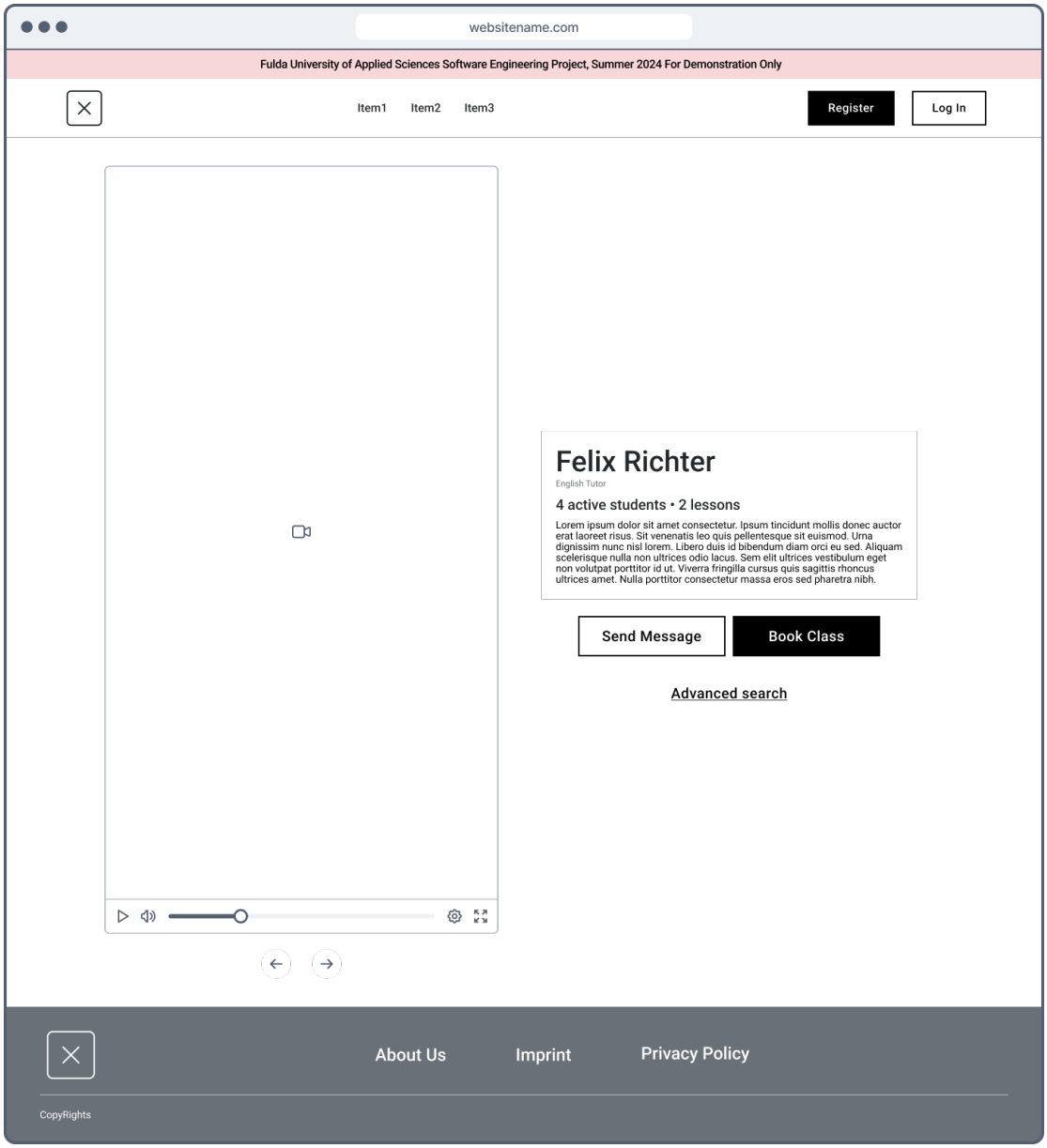


Figure 12: Tutor Swiping

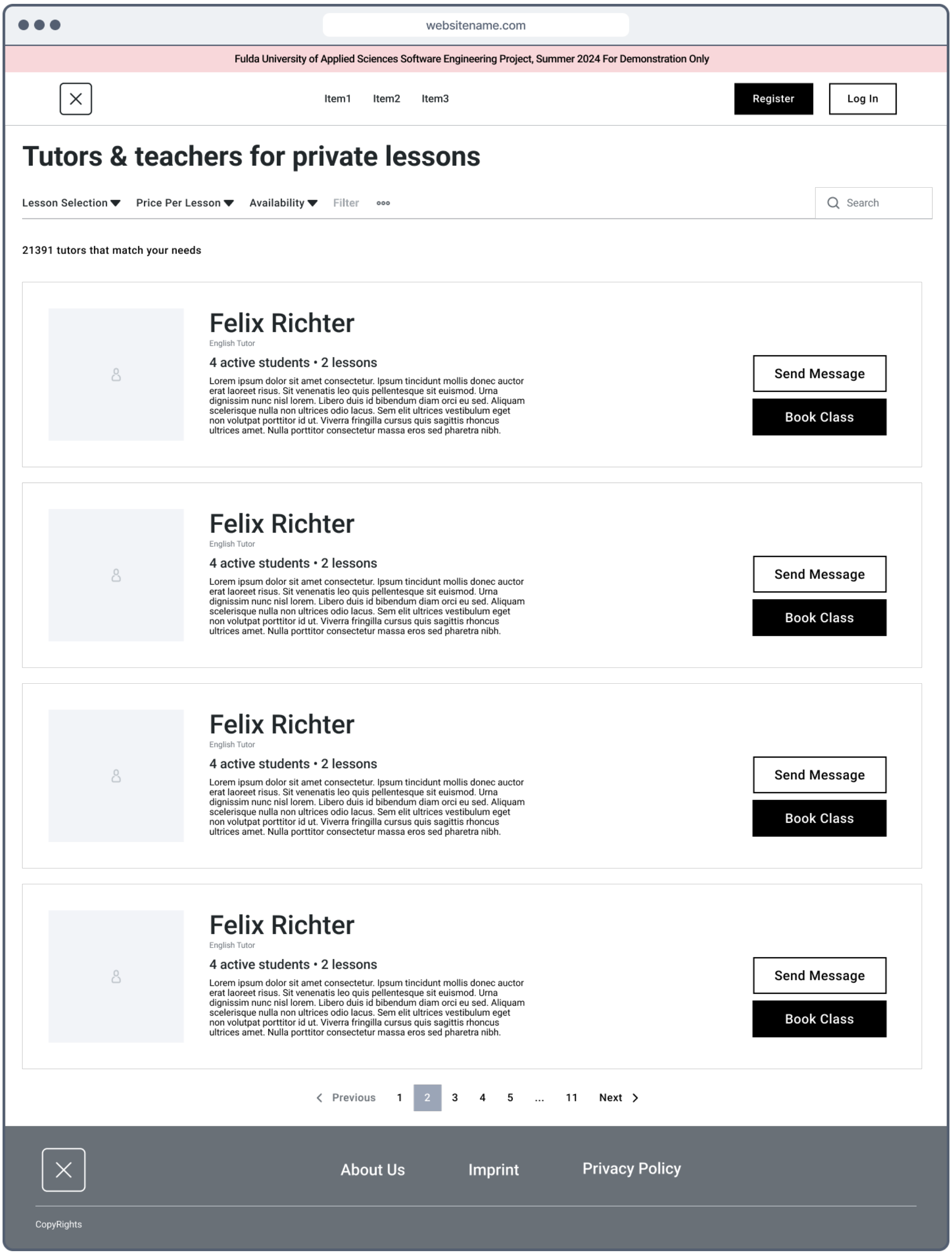


Figure 13: Tutor Search and Filter

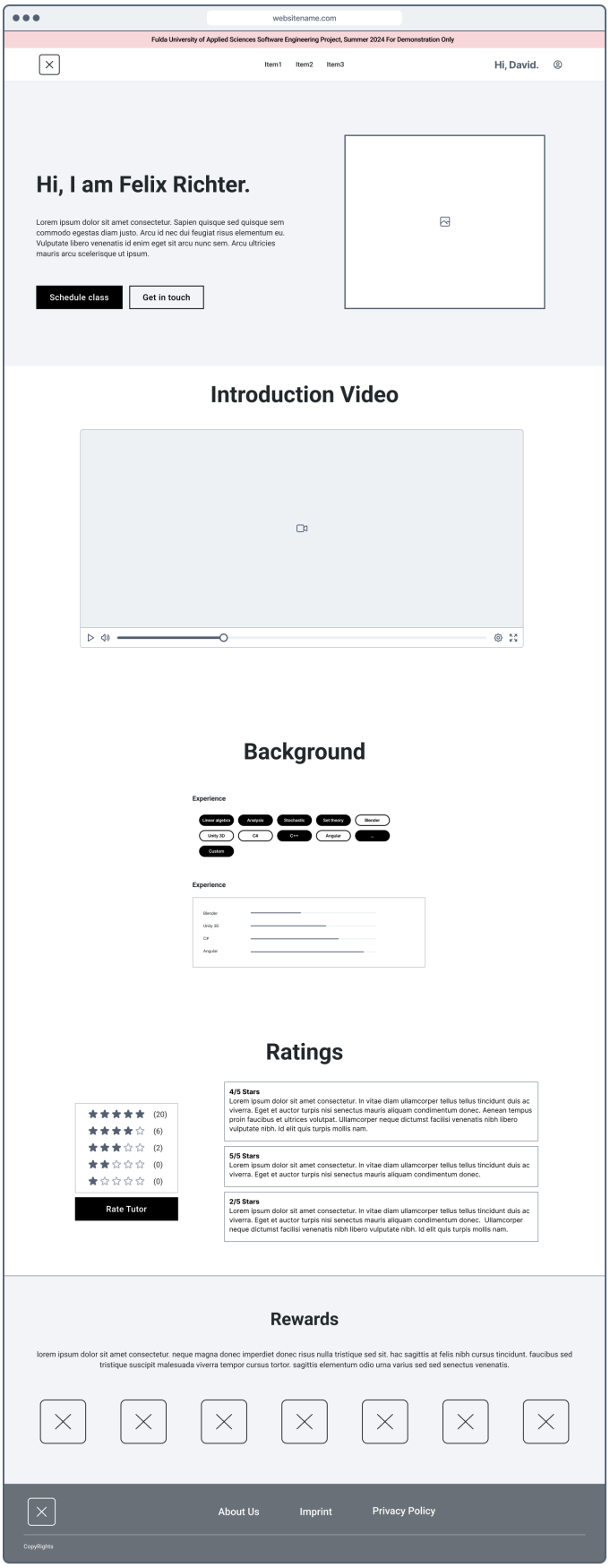


Figure 14: Tutor page

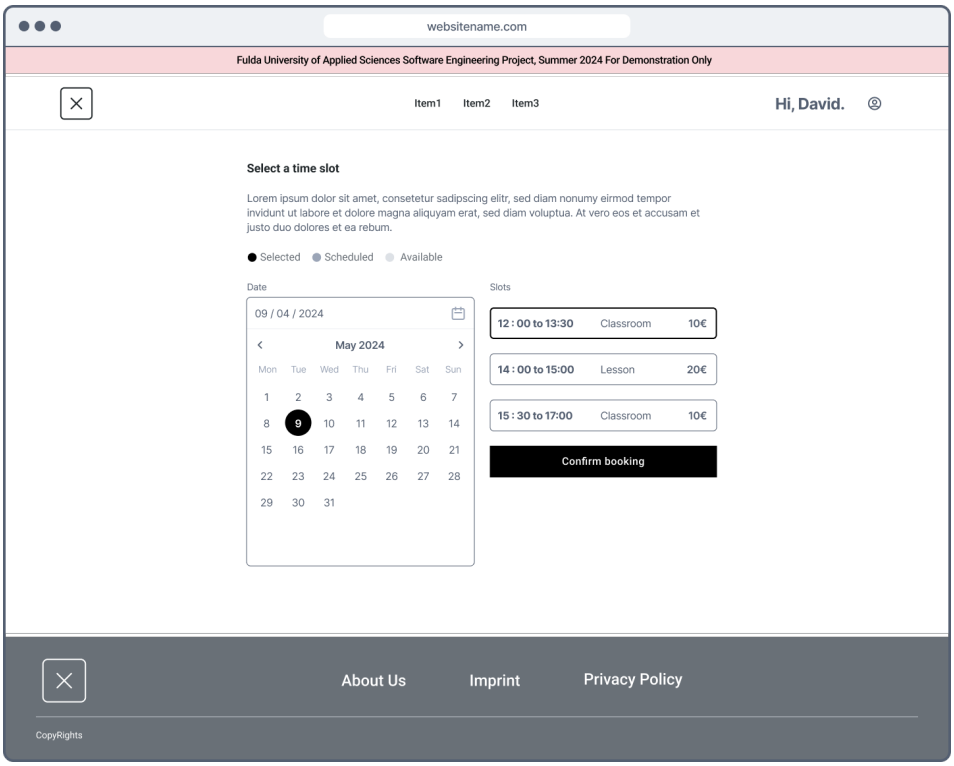


Figure 15: Schedule Meeting

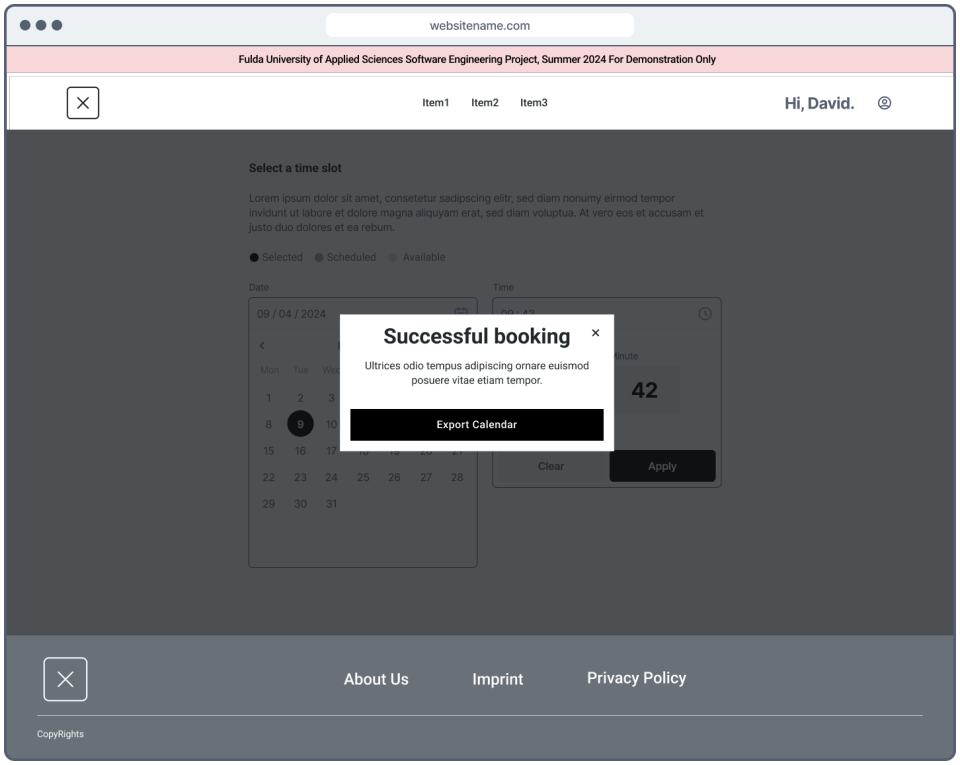


Figure 16: Export Calendar

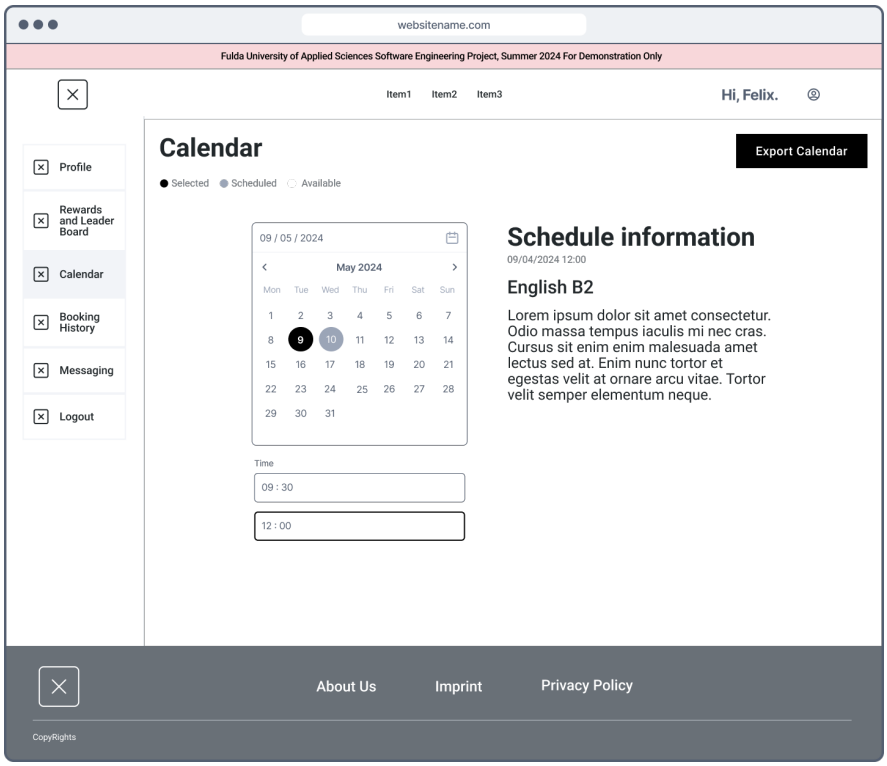


Figure 17: Calendar

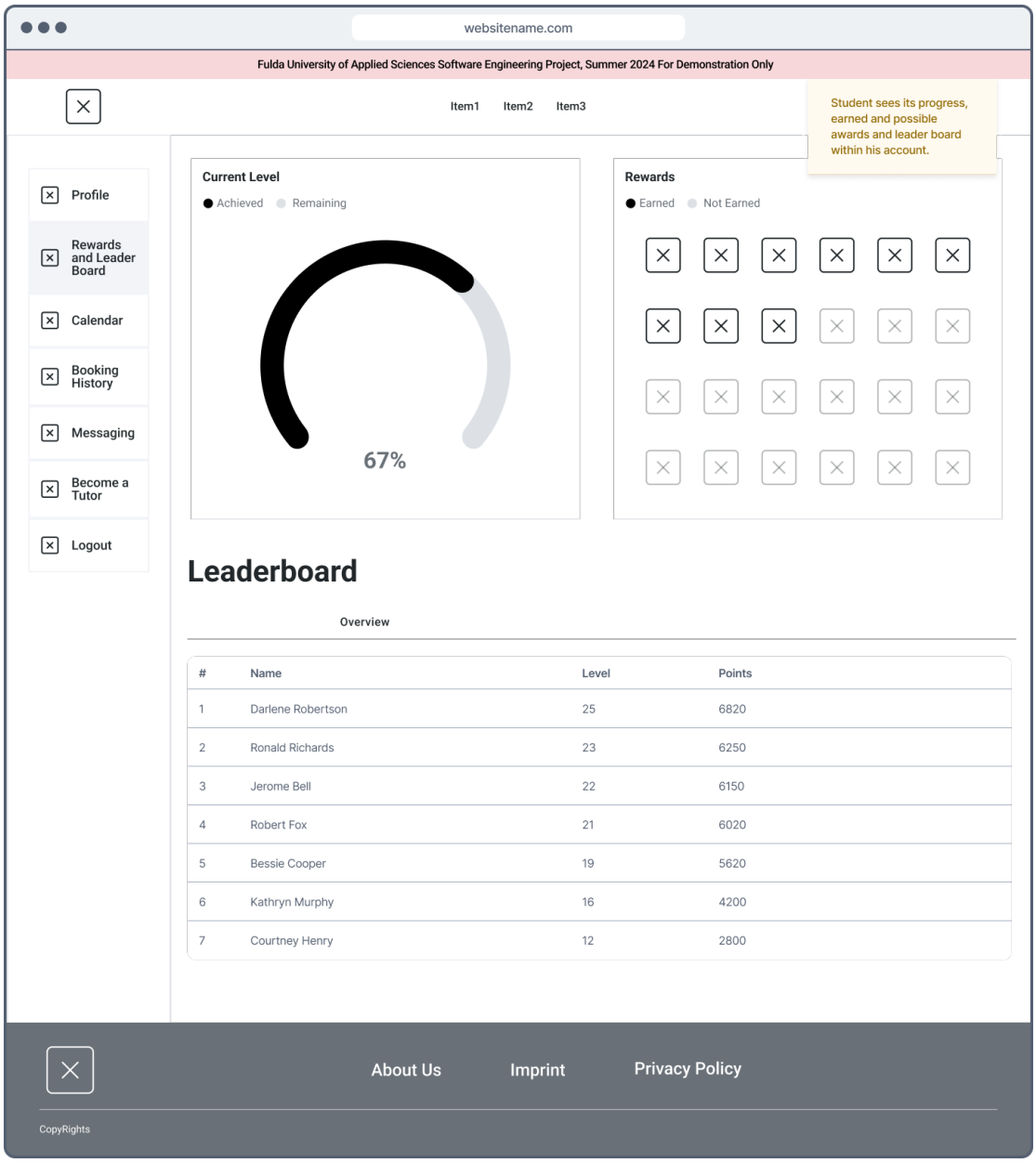


Figure 18: Rewards

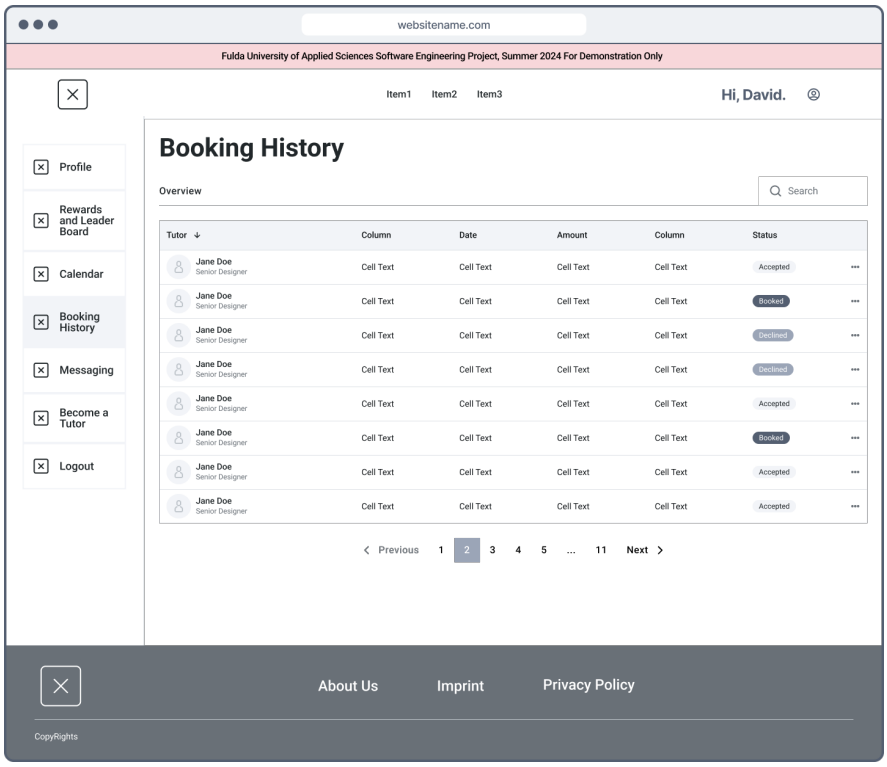


Figure 19: Booking History

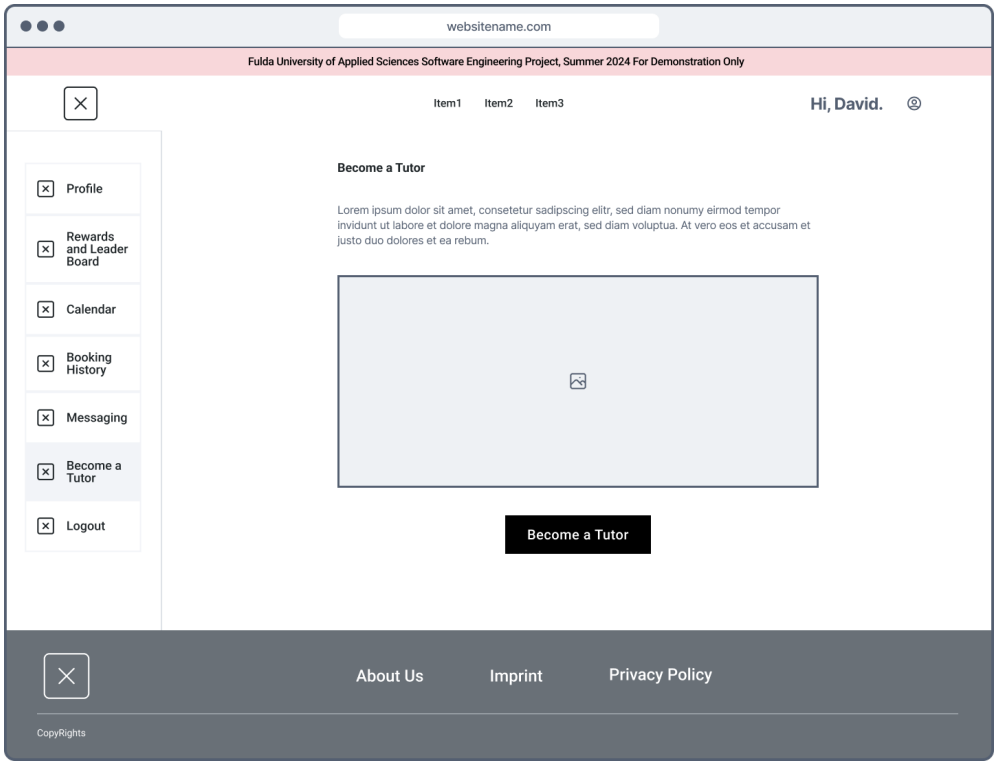


Figure 20: Become a Tutor

10.4 UC-04: Tutor Rating

After the tutoring session has taken place, David can rate the tutor based on his experience to give feedback to other students and help them make informed decisions.

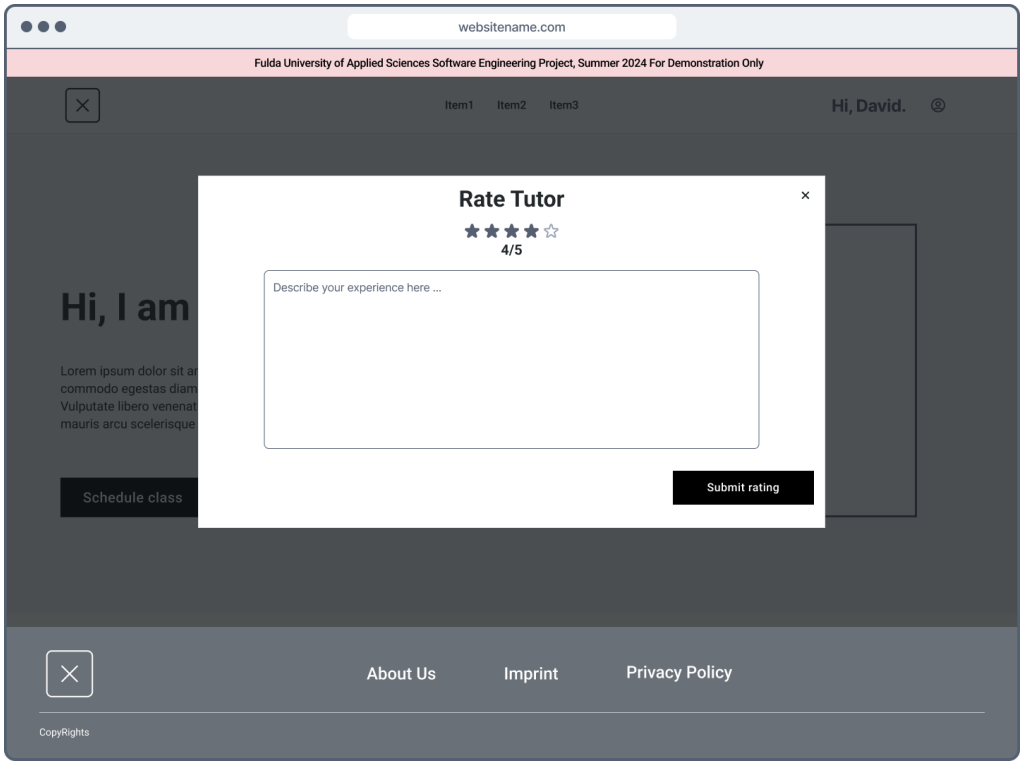


Figure 21: Rate Tutor

10.5 UC-05: Tutor Set up

Felix registers to the platform as a Tutor and fills his profile information and picture, uploads introductory video (FR-12). Felix can add free time spots in his calendar for group or individual classes (FR-10) and use messaging system (FR-09) for further communications. Felix can see his booking history(FR-08) and rewards in his account.

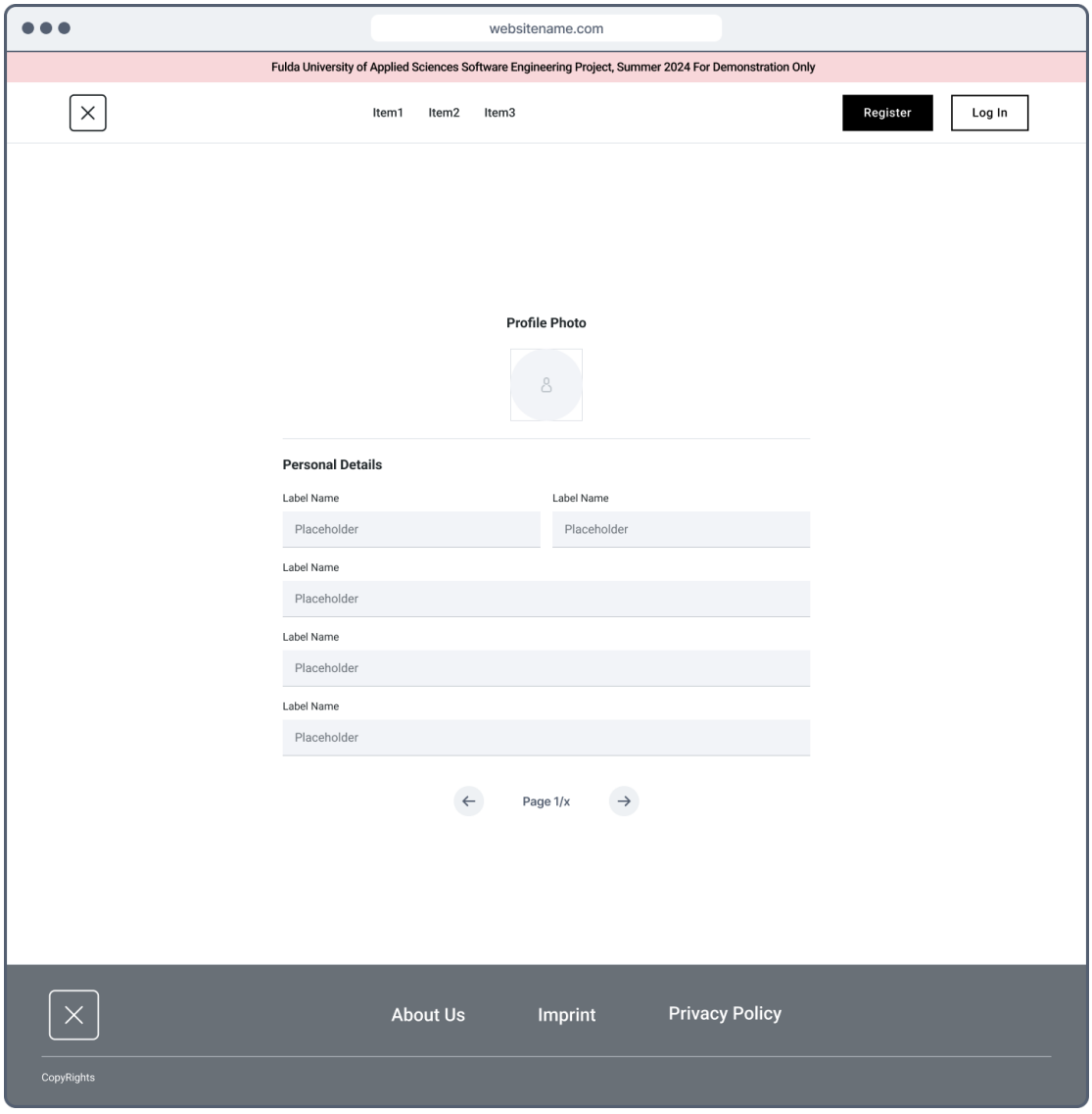


Figure 22: Fill tutor information

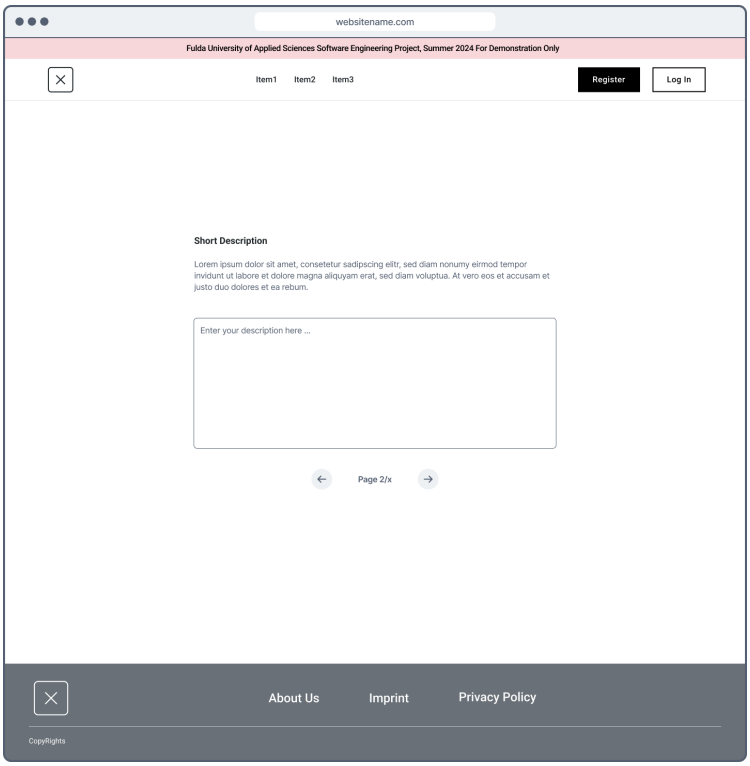


Figure 23: Short Description

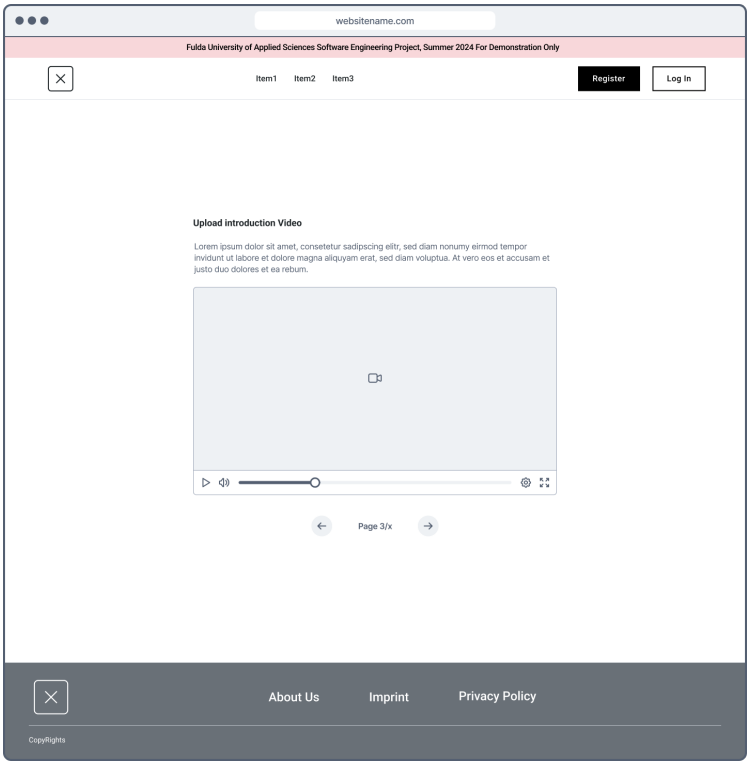


Figure 24: Introductory Video

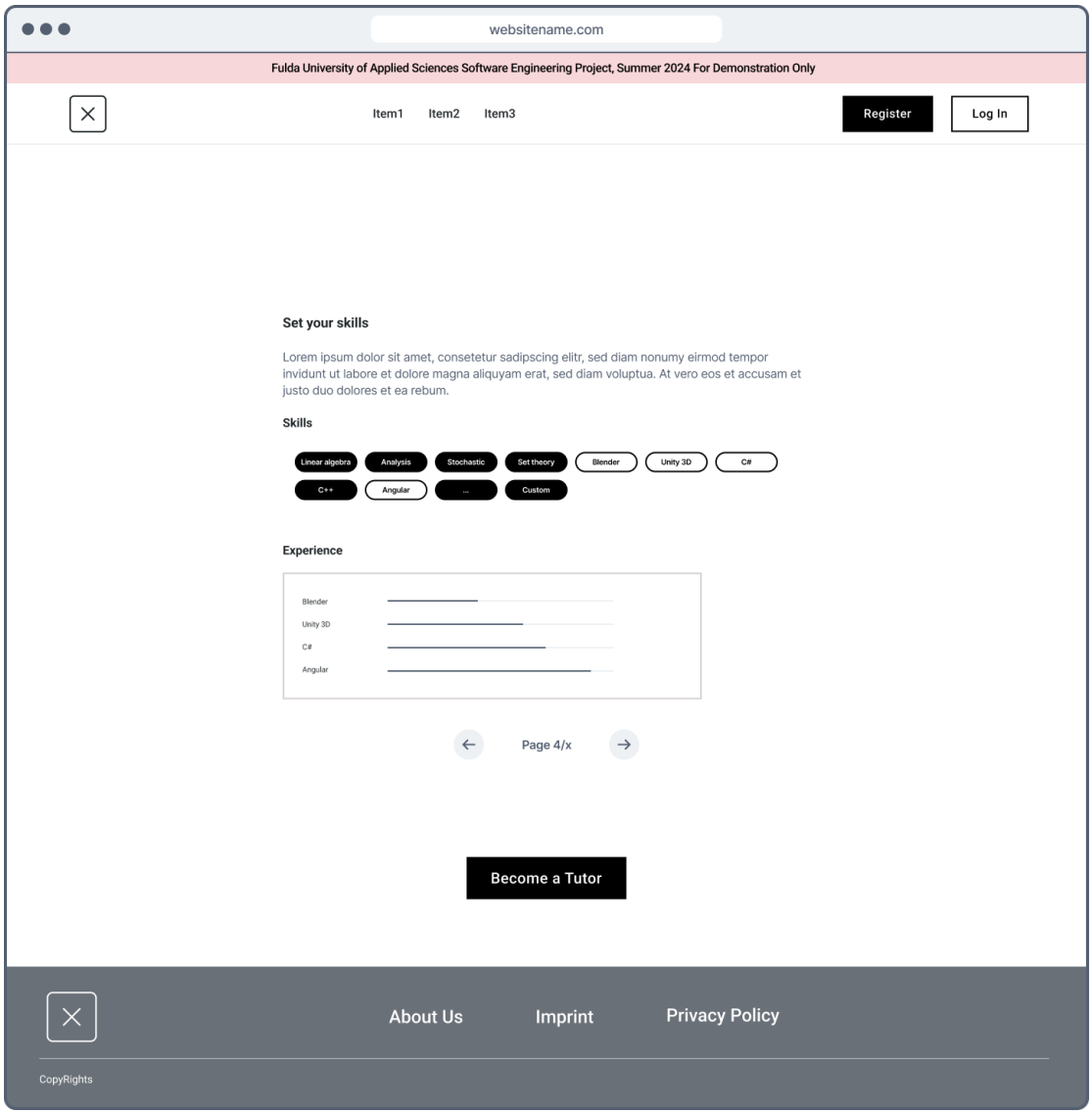


Figure 25: Tutor Skills

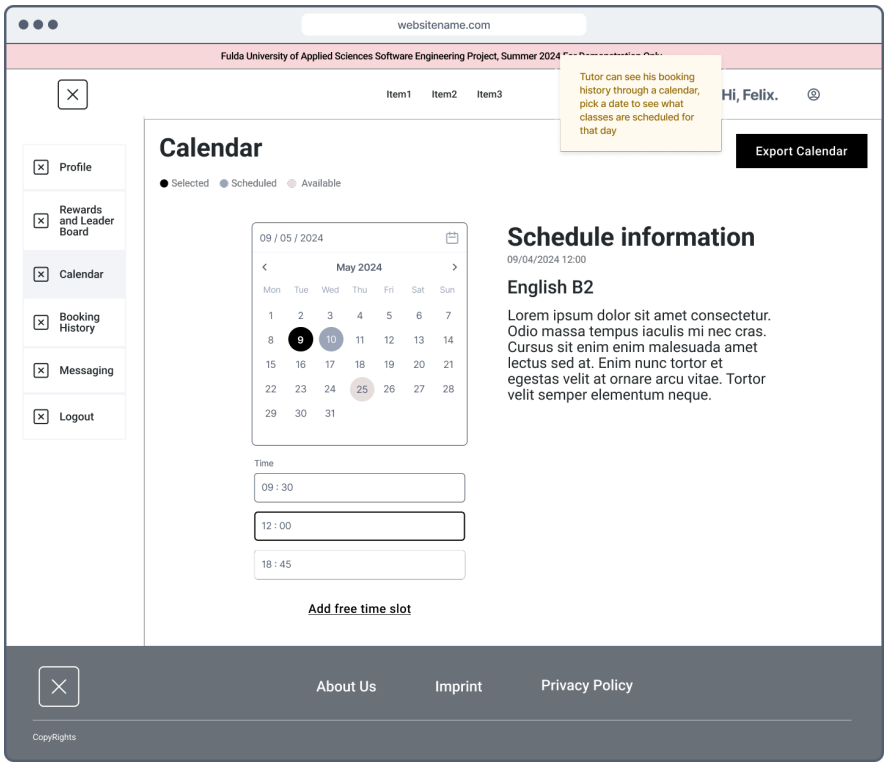


Figure 26: Calendar View - Selected time slot

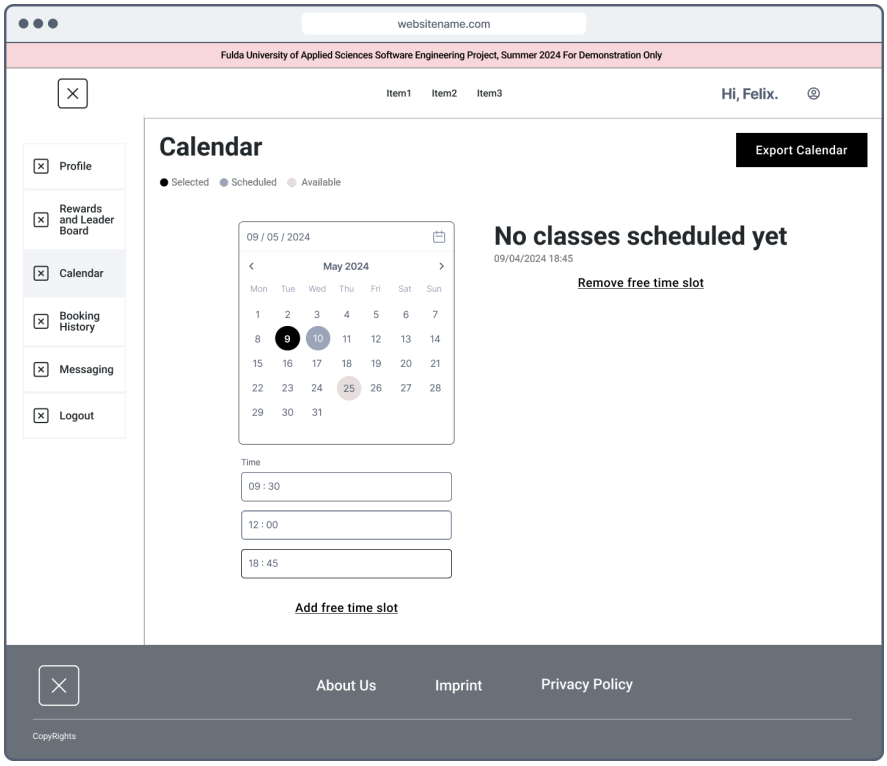


Figure 27: Calendar View - Free time slot

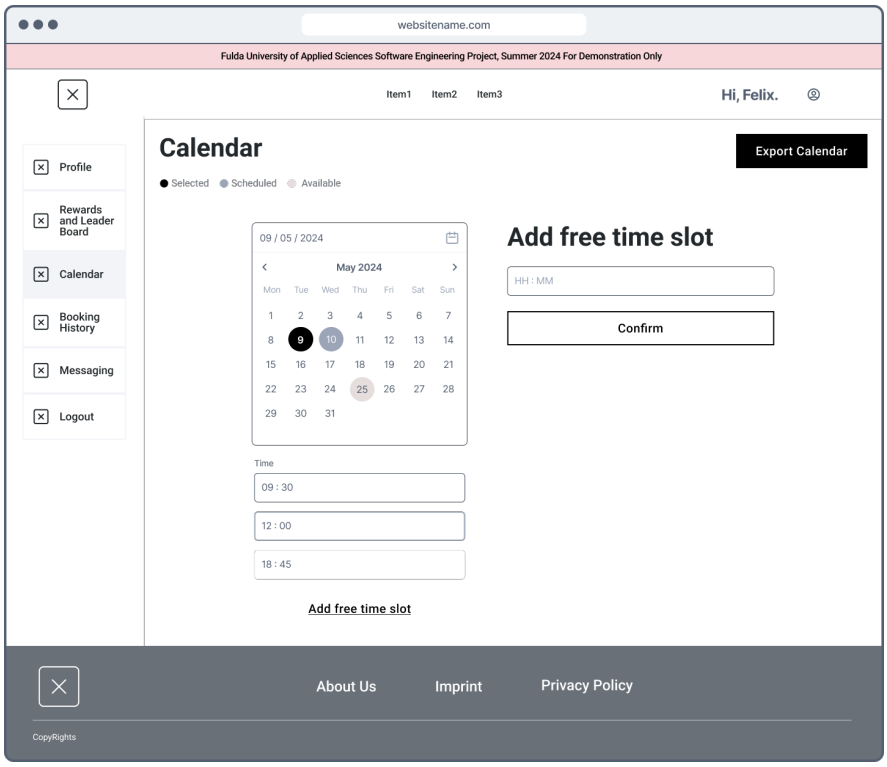


Figure 28: Calendar View - Add time slot

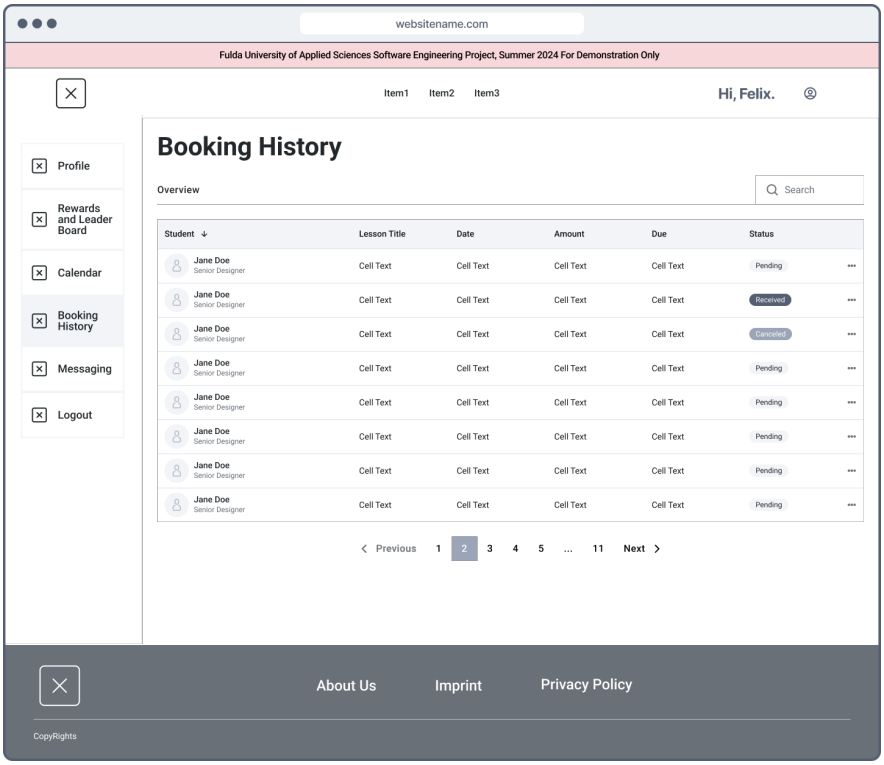


Figure 29: Booking History

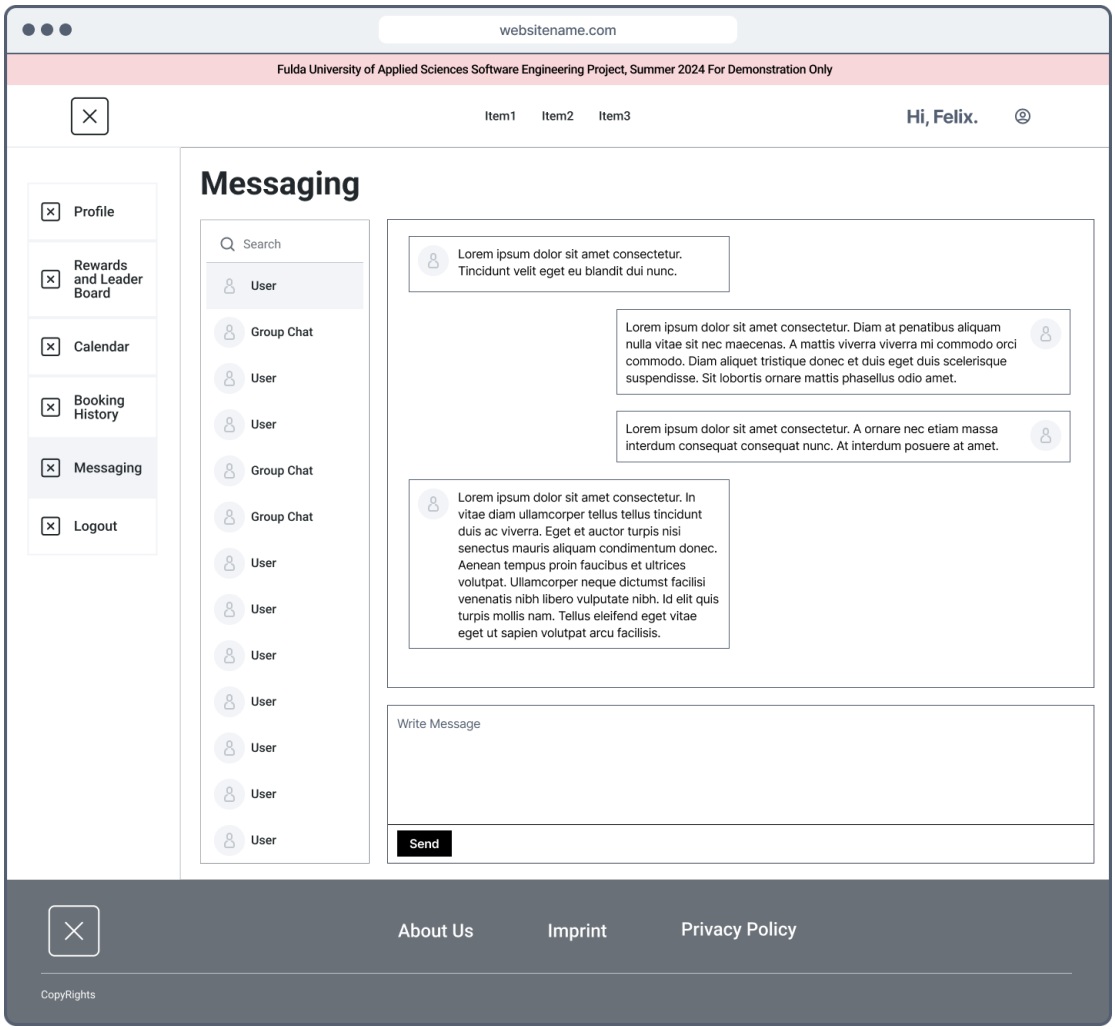


Figure 30: Messaging

10.6 UC-06: Site Administration: Content and Tutor Management

Emma must first approve the tutor and content (FR-16) before it appears on the platform. She looks at the tutor profiles and approves them if they comply with the guidelines.

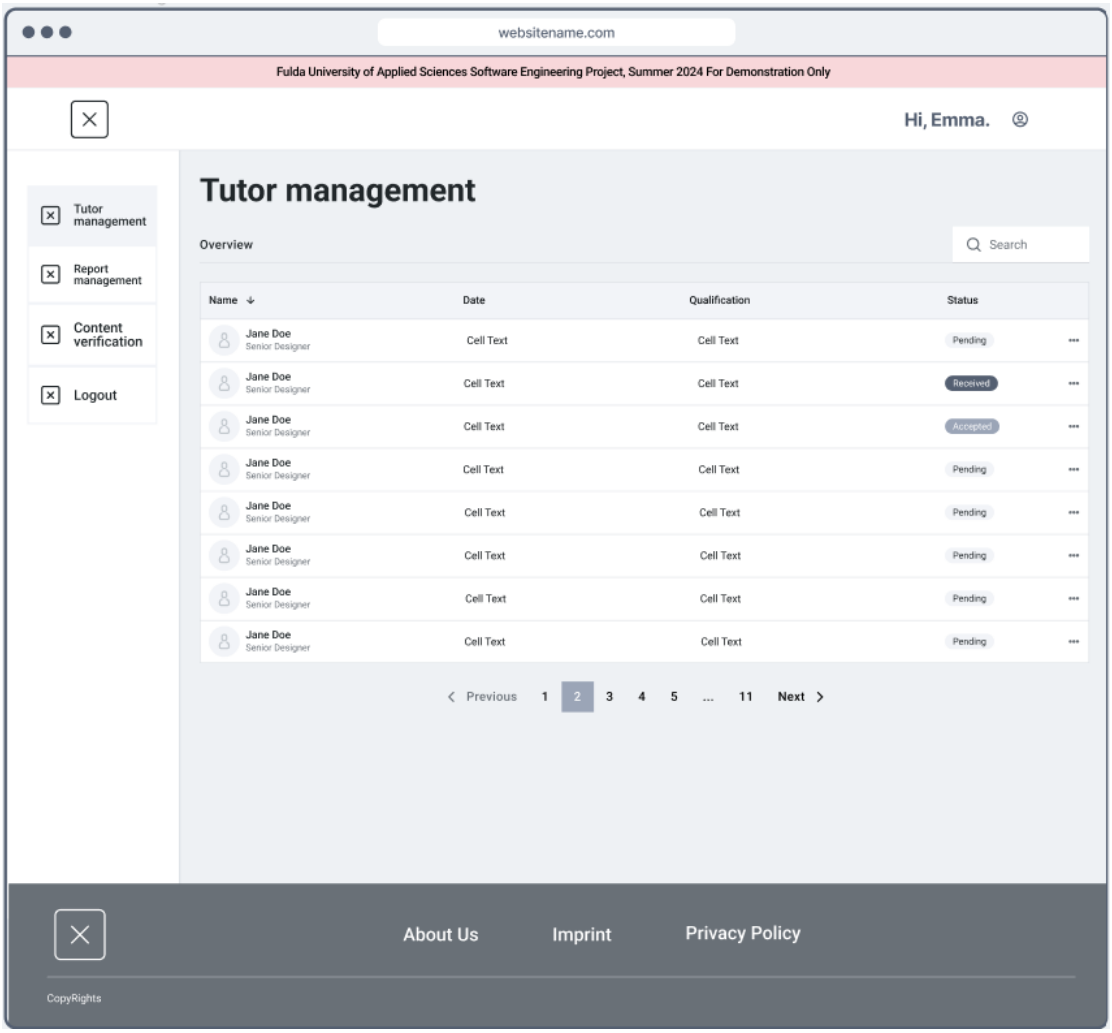


Figure 31: Tutor Management

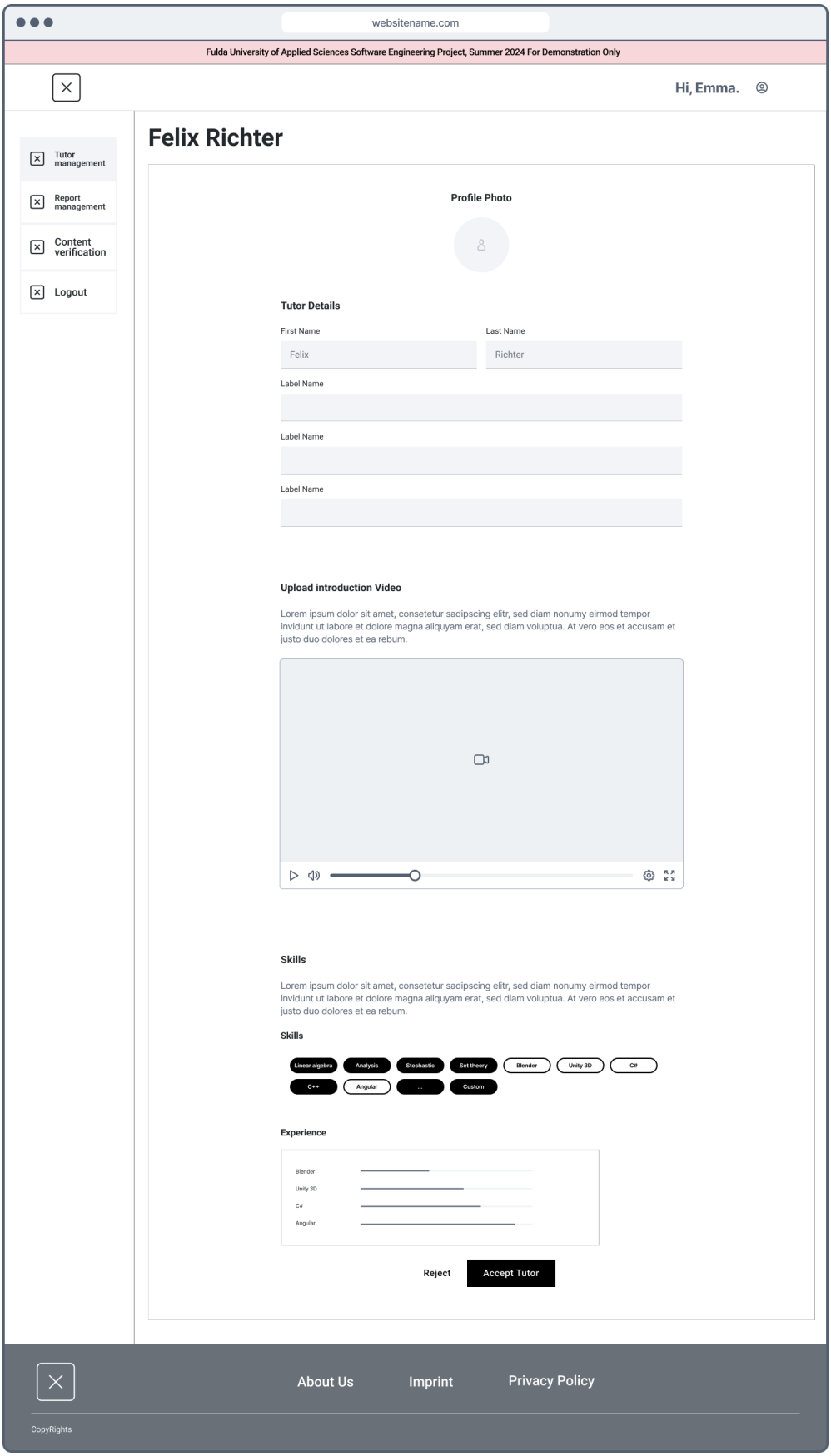


Figure 32: Tutor Review

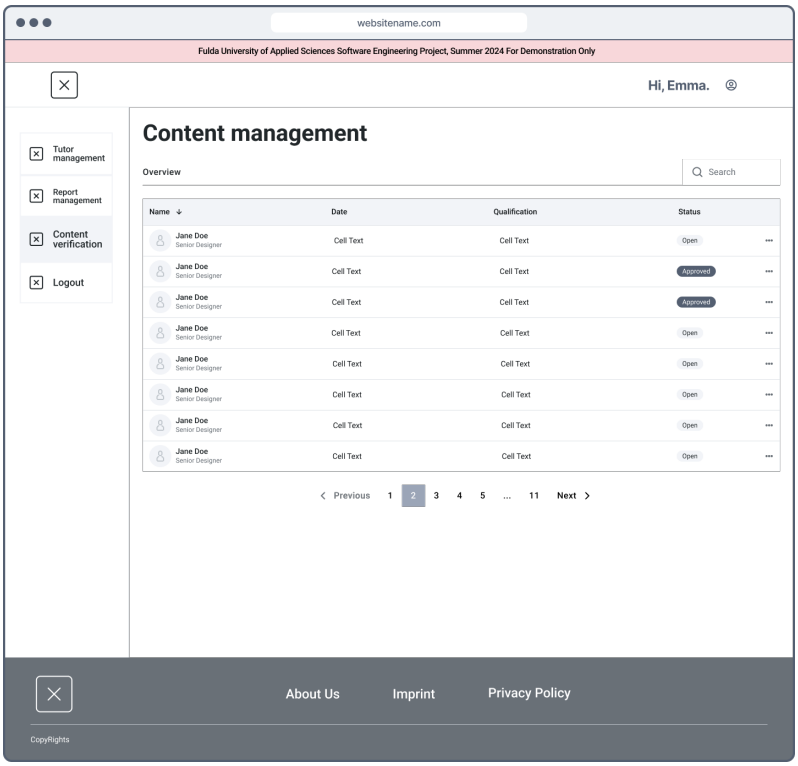


Figure 33: Content Management

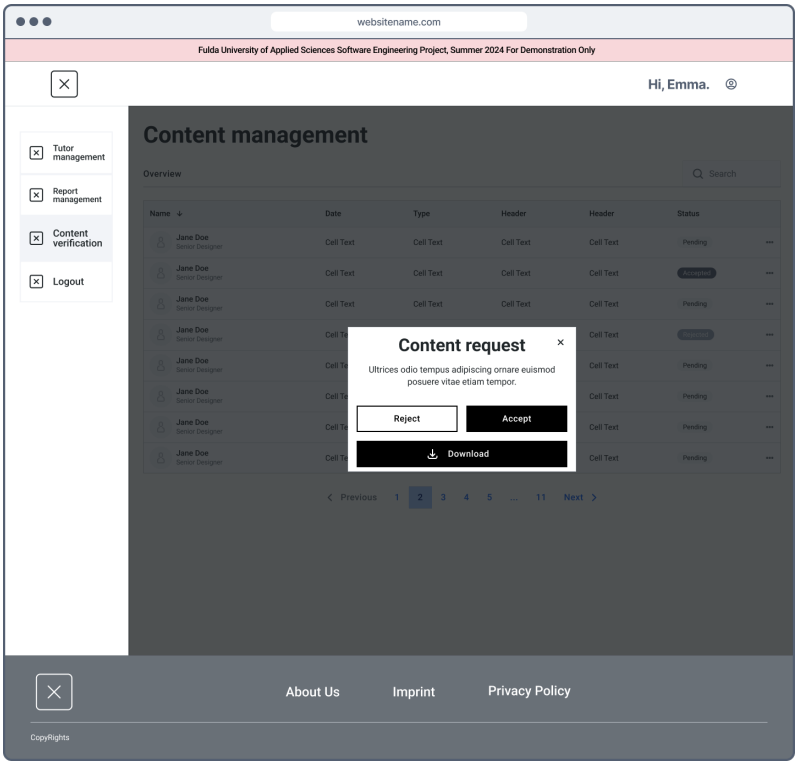


Figure 34: Content Confirmation

10.7 UC-07: Site Administration: Report management

Emma takes care of the reported users. For this purpose, she has a separate page on which she can view all reported users. Emma navigates to her Administrator page. There she looks at the individual cases and then decides on the reported user. (FR-15)

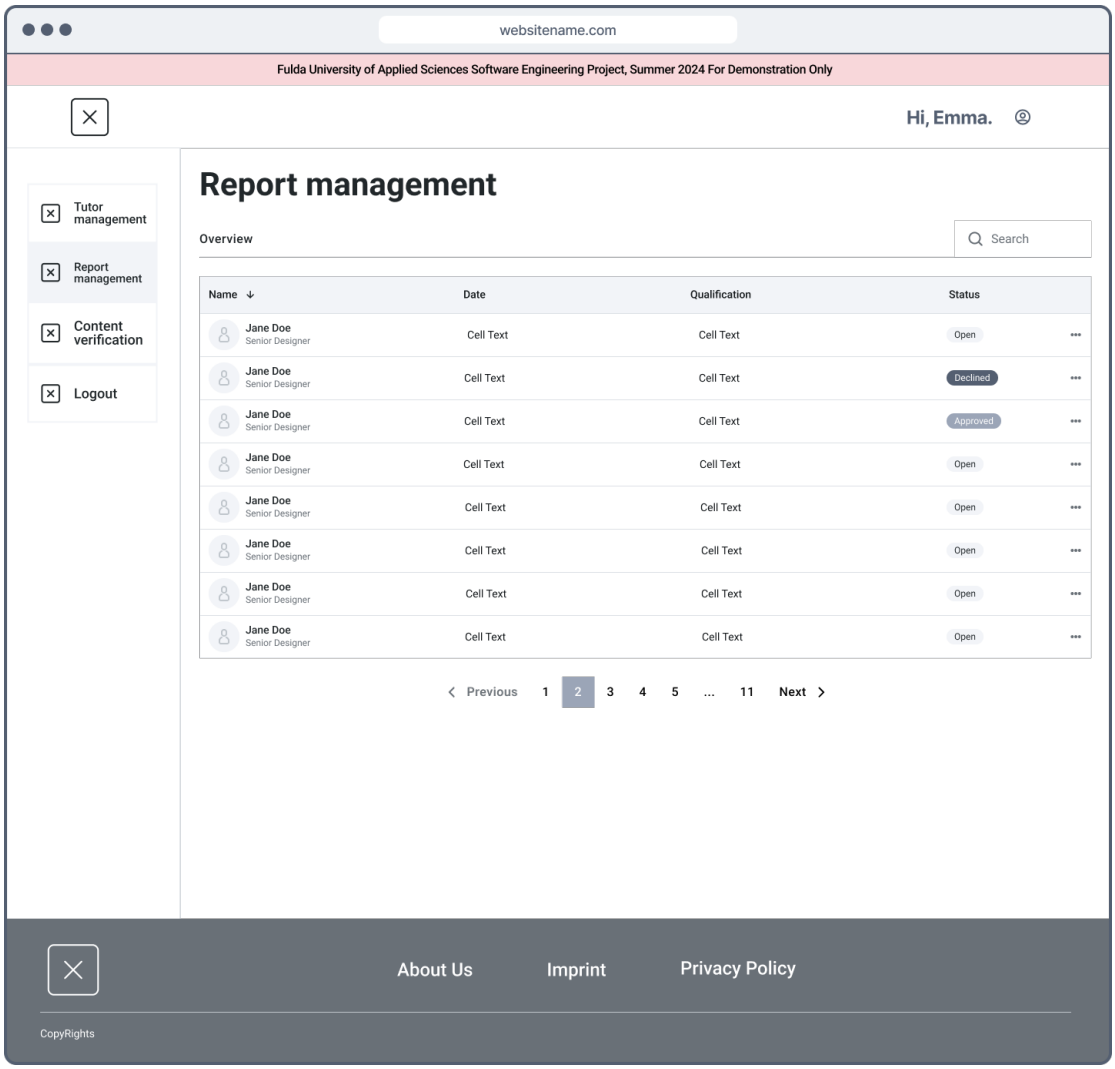


Figure 35: Report Management

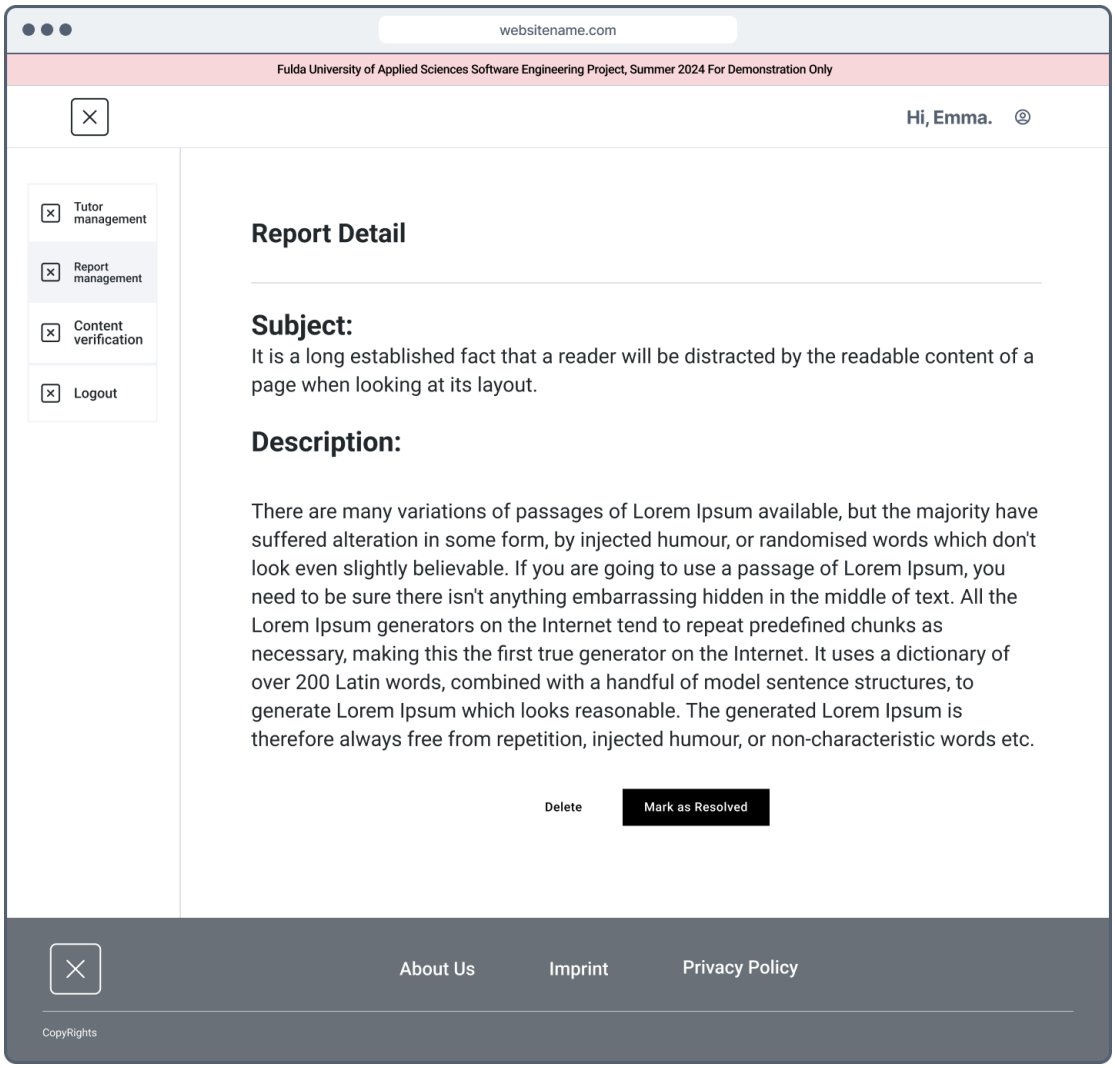


Figure 36: Report Details

11 High level Architecture:

11.1 DB organization

We have three databases in our project, that are described below:

- **App (production)**:
  - **Name**: gdsd_database
  - **Task**: This is our production database, which contains all the live data from our production application. The database is publicly accessible via *database.gdsd.stinktopf.de*.
- **App (development)**:
  - **Name**: gdsd_dev_database
  - **Task**: This is our development database. It runs locally in a Docker container and is for development purposes only. Every team member has it's own instance.
- **Keycloak (production)**:
  - **Name**: gdsd_keycloak
  - **Task**: This database is for our Keycloak instance. It's on the same server as our application production database and has different credentials for security reasons.

Figure 37 shows our database schema for the app.

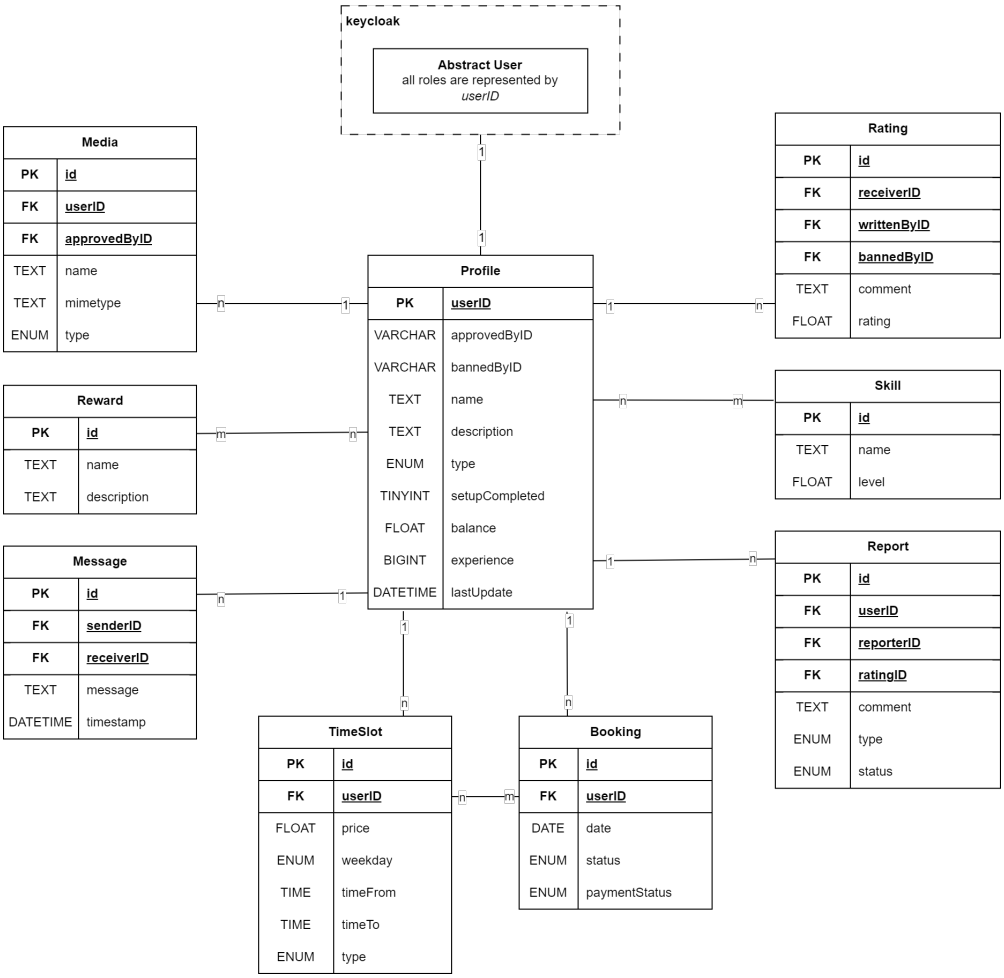


Figure 37: ER model

Additional information about the ER model:

- **The user and privileg handling is made by our keycloak instance. Therefore these entities are not represented in the model.**
- The *status* attribute in entity *Booking* can have the following values:
  - BOOKED
  - ACCEPTED
  - DECLINED
- The *paymentStatus* attribute in entity *Booking* can have the following values:
  - OPEN
  - PAYED

- The *type* attribute in entity *Profile* can have the following values:
  - STUDENT
  - TUTOR
  - ADMINISTRATOR
- The *type* attribute in entity *Report* can have the following values:
  - USER
  - RATING
- The *status* attribute in entity *Report* can have the following values:
  - OPEN
  - DECLINED
  - APPROVED
- The *weekday* attribute in entity *TimeSlot* can have the following values:
  - MONDAY
  - TUESDAY
  - WEDNESDAY
  - THURSDAY
  - FRIDAY
  - SATURDAY
  - SUNDAY
- The *type* attribute in entity *TimeSlot* can have the following values:
  - LESSON
  - CLASSROOM
- The *type* attribute in entity *Media* can have the following values:
  - PROFILE
  - INTRO_VIDEO
  - DOCUMENT
  - MISC

11.2 Media storage

The production server consists of two disks. The first disk is the internal disk that contains the operating system, configuration and so on. The second disk is an external disk that contains only the uploaded media data of our application. This disk is mounted on */datastore* at boot time and is passed as a volume to our docker containers. The following figure is a visualisation of the storage structure.

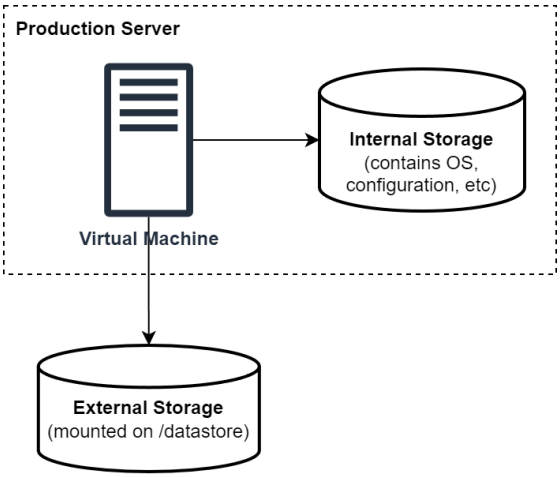


Figure 38: Visualization of storage structure

11.3 Search/filter architecutre and implementation

We will implement search functionality using SQL with the `LIKE` operator for flexible and efficient searching, suitable for text-based queries such as tutor names and skills.

Search items will be organized based on:

- **Name**: Users can search for tutors by name.
- **Rating**: Users can filter tutors based on their ratings.
- **Skills**: Users can filter tutors based on specific skills.
- **Timeslots**: Users can filter tutors based on specific timeslot.

The search functionality will utilize SQL queries with the `LIKE` operator for pattern matching. For example:

```
SELECT * FROM profile
WHERE name LIKE '%search_term%' AND type = 'tutor'
ORDER BY RAND();
```

11.4 Own APIs

Profile API

- **GET /profiles**: Retrieve a list of all profiles (including filters and sorting by query parameters).
- **GET /profiles/swiper**: Retrieve a list of all profiles prepared for the swiping feature.
- **GET /profiles/{id}**: Retrieve detailed information about a specific profile.
- **PUT /profiles**: Update a profile.

Skill API

- **GET /skills**: Retrieve a list of all skills.

Reward API

- **GET /rewards**: Retrieve a list of all rewards.

Timeslot API

- **GET /timeslots**: Get the time slots for the authenticated user.
- **GET /timeslots/{userID}**: Get the time slots for a specific user.
- **POST /timelots**: Create a new timeslot.
- **DELETE /timelots/{id}**: DELETE a specific timeslot.

Booking API

- **GET /bookings**: Retrieve a list of bookings for the authenticated user.
- **GET /bookings/history**: Retrieve a list of all bookings of the authenticated users timeslots.
- **POST /bookings**: Create a new booking.
- **PUT /bookings**: Update the booking.
- **PUT /bookings/payment**: Update the payment state of a booking.

Rating API

- **GET /ratings/{userID}**: Retrieve ratings for a specific tutor.
- **POST /ratings**: Submit a new rating.

Messaging API

- **GET /messaging**: Retrieve a list of messages for the authenticated user.
- **POST /messaging**: Send a message.

Media Management API

- **GET /media/{id}**: Retrieve a specific media file.
- **GET /media/user/{userID}**: Retrieve all media files of a specific user.
- **POST /media**: Upload a new media file.
- **PUT /media/{id}**: Updates data of media file.
- **DELETE /media/{id}**: Delete a media file.

Report API

- **POST /report**: Create a new report.
- **PUT /report/{id}**: Update details of a report.

11.5 Algorithms

We currently have not planned of using non-trivial algorithms. All algorithms used will be implemented inside the scope of the project.

11.6 Additional Software

Keycloak

We have decided to use Keycloak as our identity management system (IAM), to manage user roles (student, tutor, administrator). Keycloak will streamline user management by providing robust features for authentication, authorization, and user role management in a secure environment. This additional software was already approved by the CTO.

12 High Level UML Diagrams:

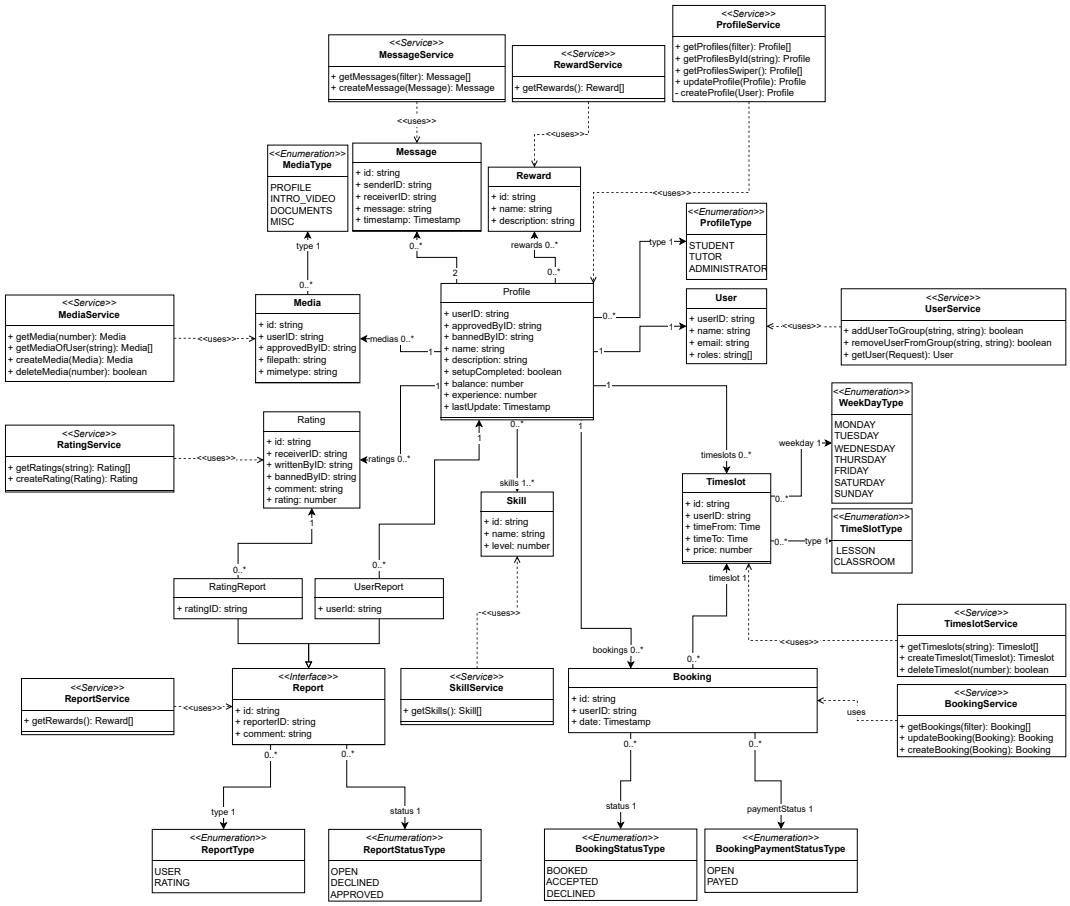


Figure 39: High-level UML class diagram

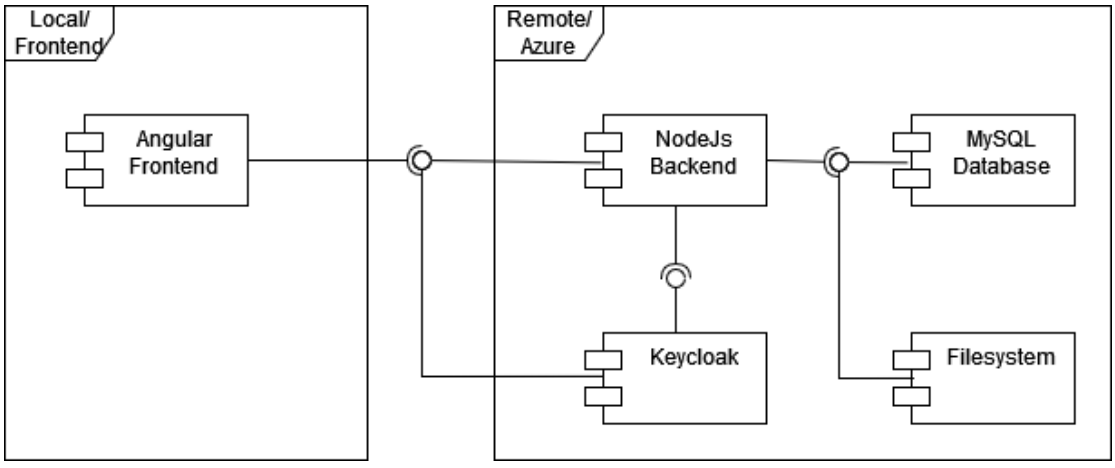


Figure 40: High-level UML component diagram

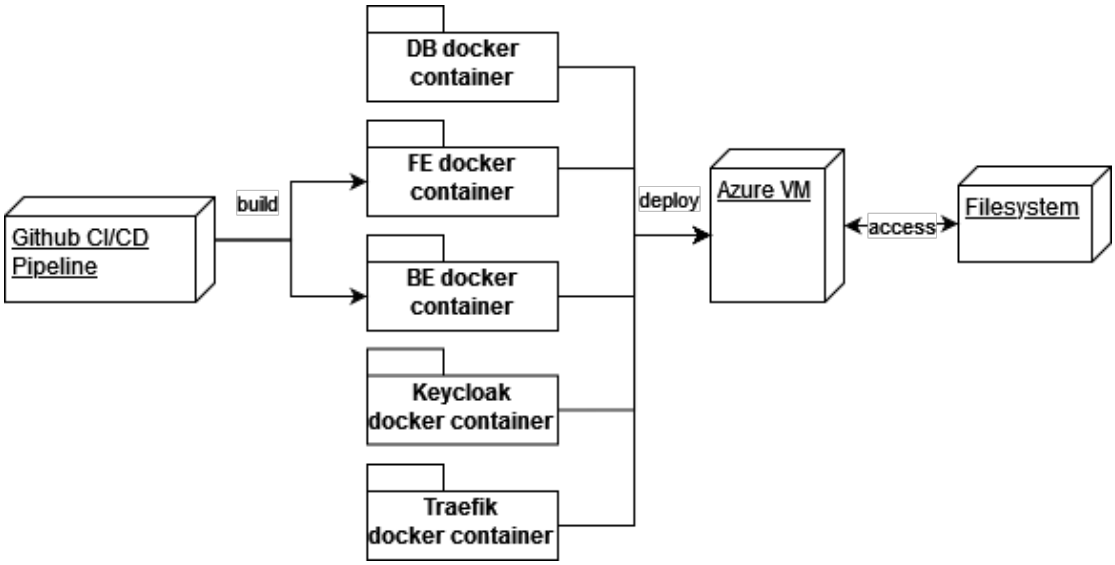


Figure 41: High-level UML deployment diagram

13 Key risks:

Skill risks:

1. Complexity of WebSocket chat deployment:

    - **Risk:** We may lack skills in deploying chats with WebSockets.
    - **Mitigation:** We will work on the WebSocket Chat as early as possible within the scope of the vertical prototype.

2. Lack of experience in working with passkeys:

    - **Risk:** The team currently lacks the knowledge to implement security solutions using passkeys.
    - **Mitigation:** The frontend and backend teams will work together at an early stage to coordinate the implementation of passkeys.

Schedule risks:

1. Time slots not overlapping:

    - **Risk:** Members have different schedules due to other projects, classes and jobs risking communication and project progress.
    - **Mitigation:** We will plan meetings in advance to accommodate everyone's schedules and to ensure that everyone is involved.

2. Spontaneous member absence:

    - **Risk:** A member suddenly not beeing able to work on the project without a timly notice could lead to the project not beeing able to hit it's deadlines. This could happen by various ways like, personal appointments, health related issues, etc.
    - **Mitigation:** We will always plan in a buffer to our deadlines to migigate this problem. If this risk occures at least one member should be able to take over the workload of the omitted person.

Technical risks:

1. Inadequate VM Specifications:

    - **Risk:** The selected VM might not meet the requirements of our project.
    - **Mitigation:** We can either switch to a larger VM or streamline the architecture to better fit the current VM's capabilities.

Teamwork risks:

1. Personal discrepancies:

    - **Risk:** In a big team is always the possibility for personal discrepancies or disagreements that could impact the work or the quality of the project or the vision.
    - **Mitigation:** We will keep the work inside the team on a professional level and try to resolve any conflicts as soon as they occure

2. Miscommunication.:

    - **Risk:** Miscommunication or lack of communication can lead to misunderstandings and mistakes in the project.
    - **Mitigation:** We will implement regular team meetings and use collaborative tools to ensure everyone is on the same page.

Legal/Content risks:

1. Copyright infringement:

    - **Risk:** When working with media, there is always the chance of using something that we are not allowed to. Especially in a media rich web application were most of the conent is uploaded by individual users.
    - **Mitigation:** For this the site administrators will perform a content verification of all media uploaded, to ensure no copyright violations.

2. Document authenticity:

    - **Risk:** Tutors will need to upload some documents to verify their expertise. These could be edited or fake.
    - **Mitigation:** The administrators will also perform a content and background verification for the documents and the tutors themself.

3. GDPR infringement during development:

    - **Risk:** There is a risk that the systems and the application process the personal data of uninvolved third parties.
    - **Mitigation:** We switch off the website when it is not in use. It would be worth considering restricting access to the website during development to the university VPN only.

4. Azure Costs:

    - **Risk:** There is a serious risk that the costs in Azure will get out of control and our credit balance will be exhausted.
    - **Mitigation:** Automation to switch off the VM and regular checks of the remaining balance by Lorenz Hohmann.

14 Project management:

From the start of our project, we implemented structured project management methods to ensure optimal workload organization and fair task distribution.

We use an iterative development approach, organizing our tasks on a Kanban board in GitHub Projects. Tasks for each milestone are linked to a GitHub milestone, and the team lead ensures fair workload distribution based on estimates and prioritization. Tasks are assigned to team members with clear responsibilities.

If a task isn't completed satisfactorily within an iteration, additional resources are allocated in the next iteration. This method proved effective during Milestone 1.

Team members manage the lifecycle of issues, with the team leader verifying the closure of tasks. For Milestone 2 and beyond, we will enhance efficiency by ensuring front-end and back-end teams operate more independently while agreeing on common interfaces. This will involve direct reporting from the front-end, back-end leads, and the GitHub master to the team lead.

We use the Kanban board on GitHub, Discord, and weekly personal meetings for project management. This combination facilitates effective communication and coordination within the team.

1 Protocol

1.1 M3

- **Swipe Feature:** The swipe functionality works.
- **Chat:** The chat functionality works.
- **Search for Tutor:** The search for tutors works as expected.
- **Try All Tutors:** Trying all tutors works correctly.
- **Change Tutor Profile Image:** Changing the tutor profile image works.
- **Admin Dashboard:** Media management within the admin dashboard works properly.
- **Search for All Tutors:** The search for all tutors is responsive.

1.2 M4

- **Usability Test:**
  - Usability tests will be done with at least 5 test individuals.
  - A Likert scale will be used to gather feedback.
  - A bar scale will show the results.
  - Testers can provide additional feedback in a comments field.
  - The results of the test will be documented.
  - This documentation is extra and not part of the 2-page limit.
  - Testers do not have to be from the university; anyone can participate.
- **QA Part:**
  - The versions of Chrome and Firefox used for testing need to be added.
  - The operating system and its version should be mentioned.
  - Document the code functions and HTTP requests, including inputs and expected outputs, in JSON or another format.
  - Perform unit testing manually.
  - Automatic tests were not used due to time constraints.
- **Peer to Peer Reviews:**
  - Each team member will participate in peer-to-peer reviews.
  - At least one file from each team member must be checked by another team member.
  - The peer review process must be documented.
  - Every team member must review a file from another team member.
  - Document the peer review, the reaction of the reviewer, and the reaction of the programmer.

1.3 M5 Final Presentation

- **Presentation:**
  - 30 minutes to showcase our work.
  - Followed by individual discussions.
  - 20–30 minutes to show the source code.
  - Discuss the big picture, including the framework used.
  - Present interesting source files.
- **Preparation:**
  - Two days before the final presentation, the team lead will send an email with one zip file.
  - The zip file will contain 5 subfolders, one for each member.
  - Each subfolder will have 3 files: 2 front-end and 1 back-end.
  - GitHub Master can include deployment scripts or database scripts.
- **File Submission:** Files must be submitted 2 days before the presentation.
- **Final Presentation Date:** September 12th at 9:00 AM.

1 Product Summary:

1.1 Product Name:

- TutorSwipe

1.2 Product Description:

TutorSwipe is an innovative online tutoring platform designed to connect students with the most suitable tutors through an engaging and intuitive video swiping feature. By leveraging video profiles, TutorSwipe enhances the tutor selection process, making it more personalized and efficient. Our platform is user-friendly, enabling students to quickly find and connect with tutors who meet their specific educational needs. We prioritize trustworthiness and security, ensuring a safe and reliable environment for both students and tutors.

1.3 Marketing and Sales Strategy:

TutorSwipe is a platform aimed exclusively at students at Fulda University of Applied Sciences. Our marketing concept is accordingly target group-orientated:

- **Partnership with Fulda University of Applied Sciences:** We could make our platform available to students at Fulda University of Applied Sciences as part of a collaboration with the Selbstlernzentrum.
- **Referral Programs:** Students can earn rewards such as meal vouchers for the university canteen by referring their peers to the platform.
- **Sponsorship of University Events:** By sponsoring university events, we can increase brand awareness and directly engage with students.
- **Presence at the University Job Fair:** Having a booth at the university's job fair allows us to introduce our platform directly to students and gather valuable feedback.
- **Social Media Presence:** We will actively engage with students on popular social media platforms such as Instagram, Facebook, and TikTok, creating engaging content, running targeted ads, and hosting live Q&A sessions to connect with our audience, targeting students of HS Fulda.
- **Billboard advertising:** We rent all billboards near Fulda University of Applied Sciences and advertise our platform there.
- **TutorSwipe buses:** We have the bus routes 6 and 16 wrapped with TutorSwipe advertising. These buses connect Fulda University of Applied Sciences with the central bus station. We can also run video adverts inside the buses.

1.4 Final List of Committed Functions:

We are committed to implement the functional requirements already specified in M2. If possible, we will also implement further P2 and P3 requirements.

The following requirements will be implemented by the end of the project:

- Unregistered users can browse tutors through a video swiping interface.
- Unregistered users can perform searches and apply filters to find tutors.
- Unregistered users can register an account as either a student or a tutor.
- Registered users can log in to their accounts to access platform features.
- Registered users can update and manage their personal profiles.
- Registered users can send and receive text messages within the platform.
- Registered users can upload, organize, and manage various media files.
- Administrators can review and approve user-generated content before it is publicly accessible.

By committing to these core functions, we ensure that we deliver a robust and functional platform. Should time and resources allow, we will aim to implement additional P2 and P3 requirements.

1.5 Unique Selling Points:

- **Video Swiping:** Provides users with a dynamic and engaging way to discover tutors. Users can view short introductory videos of tutors and swipe left or right to indicate their interest, making the tutor selection process more interactive and user-friendly compared to traditional methods.
- **Additional Features:** Gamification and rewards, scheduling system and calendar export, and organizing classrooms, which will be implemented if possible, but there is no guarantee of their implementation as they are P2 and P3 requirements and the hardening of core functions takes priority.

2 Usability Test Plan:

2.1 Test objectives:

The objective of this usability test is to evaluate the video swiping feature for finding tutors on the platform. The primary focus is to assess user satisfaction, ease of use, and overall effectiveness of the feature. This feature allows users to swipe through video profiles of tutors to find a suitable match based on their preferences.

We aim to understand how intuitively users can navigate through the tutor profiles and whether the video format enhances their decision-making process. The feature's design should facilitate a seamless and engaging experience, making it easier for students to connect with potential tutors. By assessing this feature, we intend to gather insights on any usability issues and areas for improvement, ultimately aiming to enhance the overall user experience on the tutoring platform.

2.2 Test background and setup:

System Setup: The system setup includes a functioning version of the tutoring platform with the video swiping feature enabled. The platform should be accessible via a standard web browser on both desktop and mobile devices. The test will be conducted on Google Chrome 126.0.6478.56/57 and Firefox 127.0.2 on desktop devices. All necessary user accounts and sample data (e.g. tutor video profiles) must be pre-configured.

Starting Point: Participants will begin the test from the homepage of the tutoring platform, logged into a pre-configured student account.

Intended Users: The intended users for this test are students of Hochschule Fulda seeking tutoring services. These users are familiar with basic online navigation and have experience using web-based platforms for educational purposes.

Measurement Criteria: We will focus on user satisfaction, measured through a Likert scale questionnaire administered after the completion of the task.

2.3 Usability Task description:

Participants will receive the following instructions:

1. Please start from the homepage of the tutoring platform, logged into the provided student account.
2. Use the video swiping feature to browse through the tutor profiles and search for a tutor.

3. Select the tutor you feel is the best match for your needs.
4. Once you have selected a tutor, mark the task as complete.
5. After completing the task, fill out the Likert scale questionnaire.
6. Afterwards, please tell your supervisor what you liked and didn't like about the software and what potential for improvement you see.

Effectiveness Measurement: Effectiveness is measured by the task completion rate. A task is successfully completed if the participant selects a tutor using the video swiping feature. The target effectiveness rate is 100 %.

Efficiency Measurement: Efficiency will be measured by the time taken to complete the task. This will be recorded from the moment the participant starts the task to the moment they select a tutor. The target efficiency time is 5 minutes.

2.4 User Satisfaction Evaluation

Participants will be asked to respond to the following Likert scale questions:

Q1. How easy was it to navigate the video swiping feature?

○	○	○	○	○
Very difficult	Difficult	Neutral	Easy	Very easy

Q2. How satisfied are you with the tutor selection process using the video swiping feature?

○	○	○	○	○
Very dissatisfied	Dissatisfied	Neutral	Satisfied	Very satisfied

Q3. How likely are you to use the video swiping feature again in the future?

○	○	○	○	○
Very unlikely	Unlikely	Neutral	Likely	Very likely

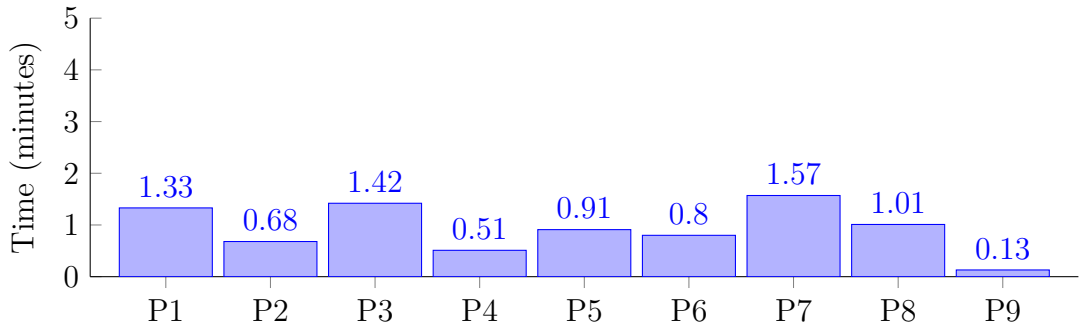
3 Usability Test Results:

The results of our usability test are presented below. The usability test was carried out with 9 participants (P). The actions that we derived from the results of the usability test are also discussed.

3.1 Effectiveness

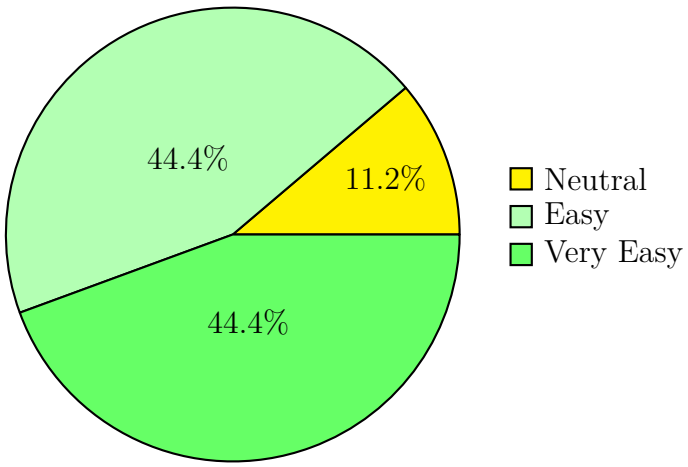
All participants completed the test successfully. Target effectiveness achieved.

3.2 Efficiency Diagram

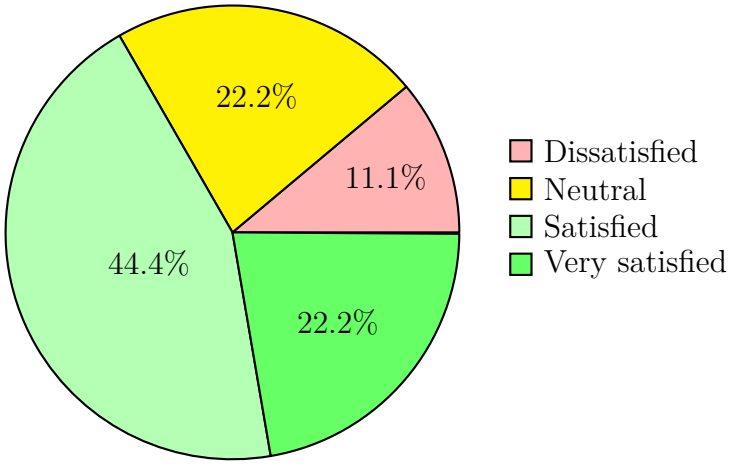


The targeted efficiency has been achieved.

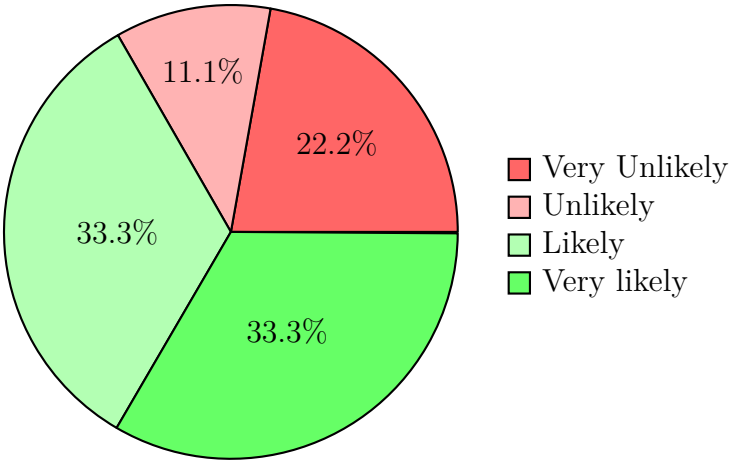
3.3 Q1. Ease of Navigation



3.4 Q2. Satisfaction with Tutor Selection Process



3.5 Q3. Likelihood of Using Video Swiping Feature Again



3.6 Free Text Responses

Participant 1

Good:

- Innovative idea to break up the traditional table view.

Bad:

- Only the profile picture is clickable.
- Swiping is too static, needs more animation.
- Clicking is cumbersome, consider keyboard integration.

Other:

- No additional comments.

Participant 2

Good:

- Structured, not overloaded.
- Rating visibility.
- Quick actions and simple design.

Bad:

- Shallow ratings, add demo ratings.
- Swiping should be draggable.
- Like/dislike buttons too small.
- Add text in video container.

Other:

- Skill click should filter advanced search.

Participant 3

Good:

- Easy to use.
- Ratings are beneficial.
- Structured and intuitive.

Bad:

- Like/dislike buttons too small.
- Colors are too cold.
- Looks empty.

Other:

- Add average response time.
- Add confirmation dialog for logout.
- Deactivate tab index on password visibility in registration.

Participant 4

Good:

- Original idea.

Bad:

- No specific bad comments.

Other:

- Pre-select topics before swiping.
- Show tutor qualifications.

Participant 5

Good:

- Easy navigation.
- Consistent color scheme.
- Efficient tutor search and messaging.

Bad:

- Like/dislike buttons could be larger.

Other:

- Add age information to tutor profiles.

Participant 6

Good:

- Clearly arranged.
- Compact tutor evaluations.

Bad:

- No specific bad comments.

Other:

- Enable scrolling without disliking.
- Allow returning to previous tutors.

Participant 7

Good:

- Usable once understood.
- Neutral.

Bad:

- Like/dislike buttons unclear.
- Prefer scrolling over swiping.
- Filters in advanced search not visible, need signal colors.
- Prefer search functionality over swiping.
- Wants full-screen videos.

Other:

- Advanced Search looked promising.
- Search bar should include subjects.
- Add pre-filters before swiping.
- Add navigation arrows.
- Display reviews under video selector.

Participant 8

Good:

- Easy to use.

Bad:

- No specific bad comments.

Other:

- Rename Advanced Search to Search.

Participant 9

Good:

- No specific good comments.

Bad:

- Button meanings unclear.

Other:

- No additional comments.

3.7 Possible actions Based on Feedback

We have identified potential features and changes. Consequently, we will evaluate them internally and decide which to implement.

- Make more areas clickable, not just the profile picture.
- Add animations to swiping.
- Integrate keyboard shortcuts.
- Show better-rated users first.
- Implement draggable swiping.
- Increase size of like/dislike buttons.
- Add text in video containers.
- Use warm colors for a more inviting interface.
- Add logout confirmation dialog.
- Deactivate tab index on password visibility in registration.
- Pre-select topics before swiping.

- Show tutor qualifications.
- Allow scrolling without disliking and returning to previous tutors.
- Use signal colors for filters in advanced search.
- Include subjects in search functionality.
- Add pre-filters before swiping.
- Add navigation arrows.
- Display reviews under video selector.
- Add age information to tutor profiles.
- Consider full-screen video options.
- Rename Advanced Search to Search.
- Clarify button meanings.

4 QA Test Plan:

4.1 Test objectives:

The primary objective of this QA test plan is to verify the functionality, performance, and user experience of the Video Swiping feature. Specifically, the tests aim to ensure that tutor videos load correctly, playback controls work as expected, the like functionality redirects properly, and appropriate error handling is in place for unavailable videos.

4.2 HW and SW setup:

Hardware Requirements:

- **Computer**: Windows PC with stable internet access
- **Processor**: Intel Core i5 or AMD Ryzen 5, multicore with 64-bit support
- **Minimum RAM**: 8 GB
- **Monitor**: FHD resolution (1920 x 1080)
- **Keyboard**: Full-size keyboard
- **Mouse**: Mouse with two buttons and a scroll wheel

Software Requirements:

- **Operating System:** Windows 10/11 (Version: 23H2)
- **Web browsers:** Google Chrome 126.0.6478.56/57 and Firefox 127.0.2

.

4.3 Feature to be Tested:

The Video Swiping Feature (FR-04) is evaluated as part of the QA test.

4.4 QA Test Plan Table for Video Swiping Feature:

#	Test Title	Test Description	Test Input	Expected Output	Results (Chrome)	Results (Firefox)
1	Video Loading	Tutor videos load correctly when swiped	Swipe to load multiple tutor videos **HTTP-Request:** https://back-end.gdsd.stink-topf.de/api/v1/media/MediaID	Videos load and play correctly **HTTP-Response:** Status Code: 200 OK Content-type: video/mp4		
2	Video Playback Controls	Ensure controls (play, pause, volume) work correctly	Use playback controls on the videos	Controls respond correctly and videos play/pause		

3	Like Functionality	Verify that liking a tutor redirects to the tutor's page	Like a tutor
			
			HTTP-Request:
			https://back-end.gdsd.stink-topf.de/api/v1/profiles/bff082a8-8509-4854-b22c-fc07480cc81f
				Redirects to the tutor's page
				
				HTTP-Response:
				Status Code: 200 OK
				Content-type: application/json
				Response:
				
				{
				"userID": "bff082a8-8509-4854-b22c-fc07480cc81f",
				"name": "Chloe Martinez",
				"type": "TUTOR",
				"description": "My name is Chloe Martinez and I am a software engineer.",
				"lastUpdate": "2024-07-04T17:20:42.000Z",
				"setupCompleted": true
				}		

4	End of Video List	Test behavior when no more tutor videos are available	Swipe through all available tutor videos	Appropriate message indicating no more videos are available and the option to navigate to the advanced search		

5 QA Test Results:

#	Test Title	Test Description	Test Input	Expected Output	Results (Chrome)	Results (Firefox)
1	Video Loading	Tutor videos load correctly when swiped	Swipe to load multiple tutor videos **HTTP-Request:** https://back-end.gdsd.stink-topf.de/api/v1/media/MediaID	Videos load and play correctly **HTTP-Response:** Status Code: 200 OK Content-type: video/mp4	Pass	Pass
2	Video Playback Controls	Ensure controls (play, pause, volume) work correctly	Use playback controls on the videos	Controls respond correctly and videos play/pause	Pass	Pass

3	Like Functionality	Verify that liking a tutor redirects to the tutor's page	Like a tutor
			
			HTTP-Request:
			https://back-end.gdsd.stink-topf.de/api/v1/profiles/bff082a8-8509-4854-b22c-fc07480cc81f
				Redirects to the tutor's page
				
				HTTP-Response:
				Status Code: 200 OK
				Content-type: application/json
				Response:
				
				{
				"userID": "bff082a8-8509-4854-b22c-fc07480cc81f",
				"name": "Chloe Martinez",
				"type": "TUTOR",
				"description": "My name is Chloe Martinez and I am a software engineer.",
				"lastUpdate": "2024-07-04T17:20:42.000Z",
				"setupCompleted": true
				}
					Pass	Pass

4	End of Video List	Test behavior when no more tutor videos are available	Swipe through all available tutor videos	Appropriate message indicating no more videos are available and the option to navigate to the advanced search	Pass	Pass

7 Security Self-check:

In our commitment to developing a robust and secure application, we adhere to industry best practices to safeguard our platform, our users, and our systems.

7.1 Major Assets and Threats

- **User Data:**
  - **Threats:** Unauthorized access, data breaches, identity theft
  - **Protection:** Secure authentication and authorization using Keycloak, strong password policies, and encryption transit data using HTTP/S.
- **Database:**
  - **Threats:** SQL injection, data corruption, unauthorized access
  - **Protection:** Use of ORM (Prisma) to escape SQL queries and enforce different user role permissions.
- **Media Files:**
  - **Threats:** Unauthorized access, malware, data breaches
  - **Protection:** Secure, access-controlled storage and manual approval of media files by an administrator.
- **Server:**
  - **Threats:** Unauthorized access, DDoS attacks, malware infections
  - **Protection:** Use of a reverse proxy, key-based authentication, firewalls, regular software updates, and Docker for application isolation.

7.2 Confirmation of Security Measures

Encryption of Passwords:

We use Keycloak as our authentication server, employing the PBKDF2 algorithm with salts and 27,500 hash iterations to ensure safe password storage. Keycloak ensures that the passwords are encrypted, as shown in Figure 1.

USER_ID	SECRET_DATA
661ef258-b7ac-495c-acec-fe...	{"value":"aq8E8IFSF0+3U3MY+XlFlrTyek4S8jLlaJN1o640HoF2XeGJKy8ft6kJCn7aYj0907cXxgYAv8yxk7KYgP8X0g==","salt":"vkWUvcKVOsanSIZEjUVX9w==","additi...
63643dcd-874f-449a-8140-5...	{"value":"tEqkdkpz4k5MiYwxLemdo1TdJwpxfE588U/QkIhCYBNm4AYIK6fUUmfAVA4itgQ5wzDzZ7eQKj/h0ozd4Bbl8w==","salt":"SdKIq3cqVeHzs2QUfMnTvw==","a...
2b7e42a1-2b5b-4b88-b1c0-...	{"value":"3lAFdnu3ATtufjs5aHsVa7Cw59RC16Hfo7xdpwXBD9nJ3GT4CiWpr1NGPL7fvJ2nk8/h3Cg1MAU0vw5SgIqAoQ==","salt":"pKvY4Dhvw6SIYYnSXEt7Cg==","ad...

Figure 1: Hashed password in Keycloak database

Backend and Input Data Validation:

We use Prisma ORM in our Node.js backend to prevent SQL injection and other database-related vulnerabilities. We validate search bar input for up to 40 alphanumeric characters in both the frontend and backend using string length restrictions to ensure only valid inputs are accepted.

Frontend Code (advanced-search.component.html):

```
[...]
<div class="flex justify-content-end">
  <input
    type="text"
    pInputText
    placeholder="Search"
    maxlength="40"
    class="w-8rem sm:w-auto"
    [(ngModel)]="search"
    (ngModelChange)="onSearch($event)"/>
</div>
[...]
```

Backend Code (profile.routes.ts):

```
[...]
// Abort request if name filter is over 40 characters long
if (filter.nameContains && filter.nameContains.length > 40) {
  res.status(400).send();
  return;
}
[...]
```

8 Non-functional Requirements Self-check:

Requirement ID: NF-01
Title: Development and Deployment Standards
Description: Application shall be developed, tested and deployed using tools and servers approved by Class CTO and as agreed in Milestone 0. Application delivery shall be from chosen cloud server.
Status: DONE

Requirement ID: NF-02
Title: Browser Compatibility
Description: Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of two major browsers.
Status: DONE

Requirement ID: NF-03
Title: Mobile Responsiveness
Description: All or selected application functions must render well on mobile devices.
Status: DONE

Requirement ID: NF-04
Title: Data Storage Specifications
Description: Data shall be stored in the database on the team's deployment cloud server.
Status: DONE

Requirement ID: NF-05
Title: User Load Management
Description: No more than 50 concurrent users shall be accessing the application at any time.
Status: DONE

Requirement ID: NF-06
Title: Privacy Policy Compliance
Description: Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.
Status: DONE

Requirement ID: NF-07
Title: Language Specification
Description: The language used shall be English (no localization needed).
Status: DONE

Requirement ID: NF-08
Title: User Experience Design
Description: Application shall be very easy to use and intuitive.
Status: DONE

Requirement ID: NF-09
Title: Architecture Pattern Adherence
Description: Application should follow established architecture patterns.
Status: DONE

Requirement ID: NF-10
Title: Code and Repository Maintenance
Description: Application code and its repository shall be easy to inspect and maintain.
Status: DONE

Requirement ID: NF-11
Title: Use of Analytics
Description: Google analytics shall be used (optional for Fulda teams).
Status: OPTIONAL, NOT IMPLEMENTED

Requirement ID: NF-12
Title: Email Client Prohibition
Description: No e-mail clients shall be allowed.
Status: DONE

Requirement ID: NF-13
Title: Payment Functionality Prohibition
Description: Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI.
Status: DONE

Requirement ID: NF-14
Title: Application Security Practices
Description: Site security: basic best practices shall be applied (as covered in the class) for main data items.
Status: DONE

Requirement ID: NF-15
Title: Media Integration
Description: Application shall be media rich (images, video etc.). Media formats shall be standard as used in the market today.
Status: DONE

Requirement ID: NF-16
Title: Software Development Practices
Description: Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development.
Status: DONE

Requirement ID: NF-17
Title: Code Management via GitHub
Description: For code development and management, as well as documentation like formal milestones required in the class, each team shall use their own GitHub to be set-up by class instructors and started by each team during Milestone 0.
Status: DONE

Requirement ID: NF-18
Title: Application Branding
Description: The application UI (WWW and mobile) shall prominently display the following exact text on all pages "Fulda University of Applied Sciences Software Engineering Project, Fall 2021 For Demonstration Only" at the top of the WWW page. (Important so as to not confuse this with a real application).
Status: DONE

1 Screenshots of our presentation

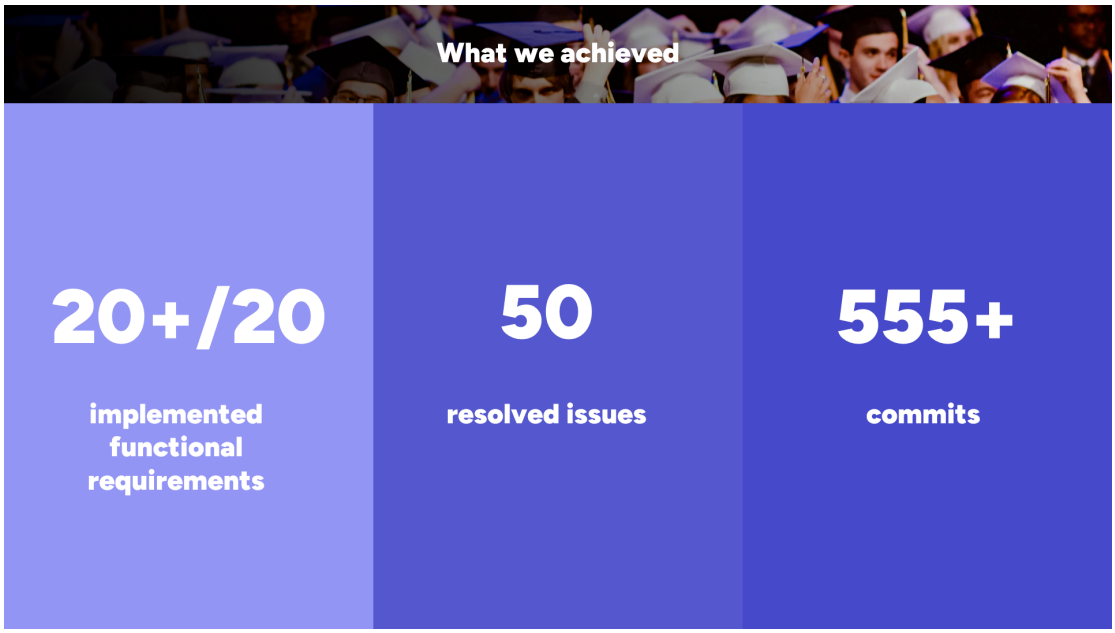


Figure 1: What we achieved

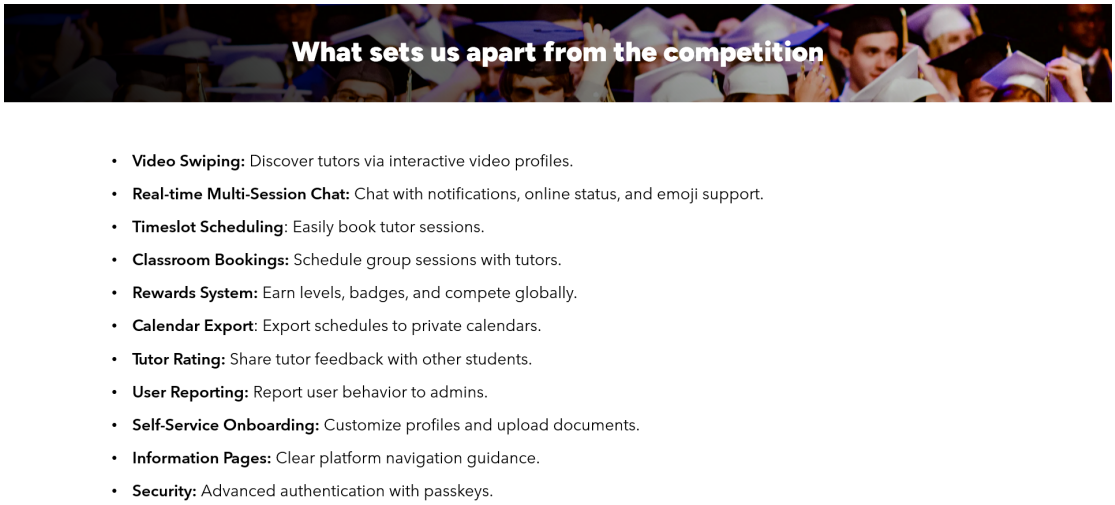


Figure 2: What we achieved

2 Screenshots of our application

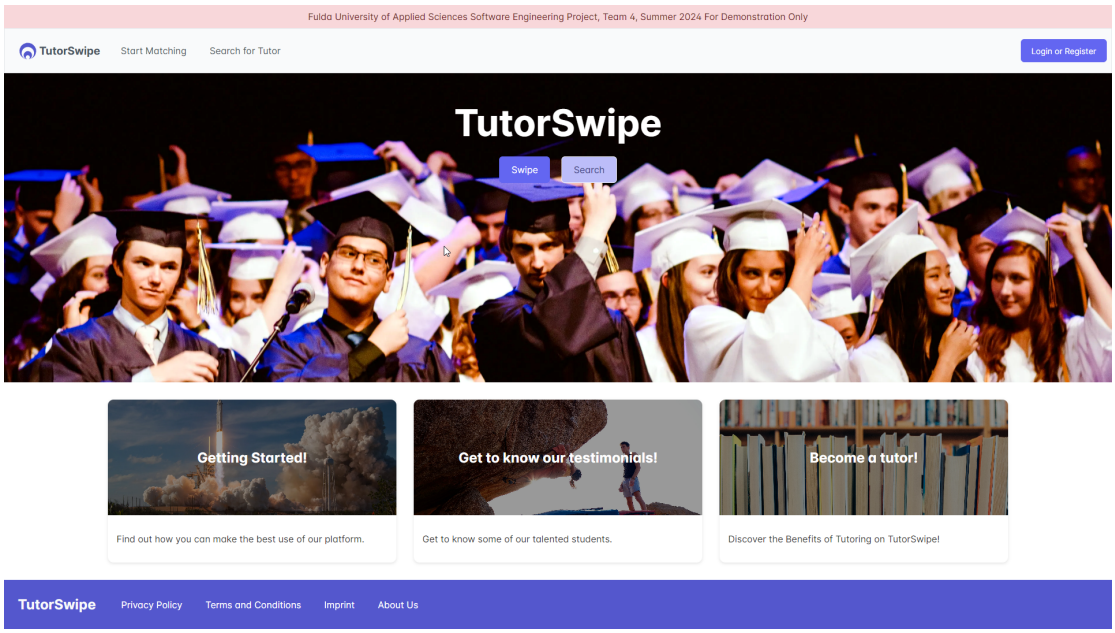


Figure 3: Landing page of TutorSwipe

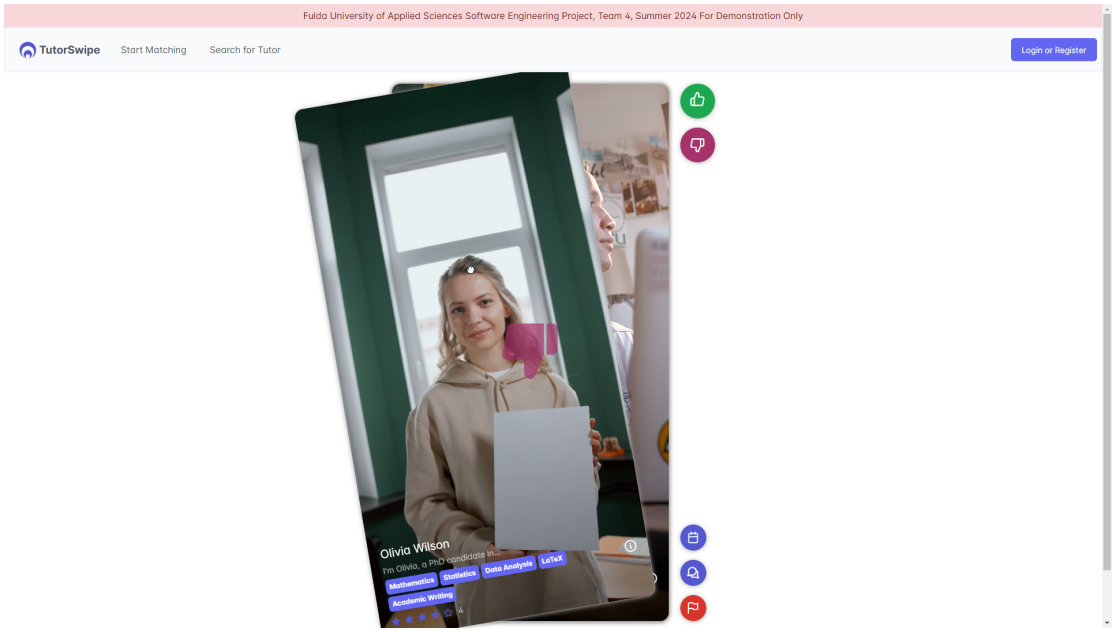


Figure 4: Video Swiping Feature

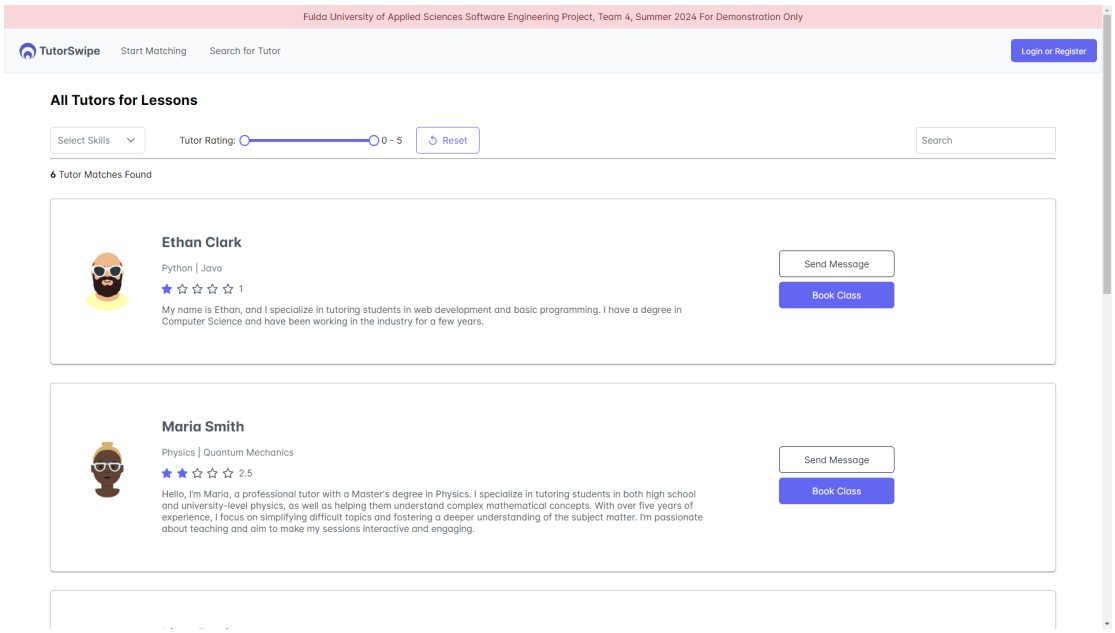


Figure 5: Advanced Search functionality with filters

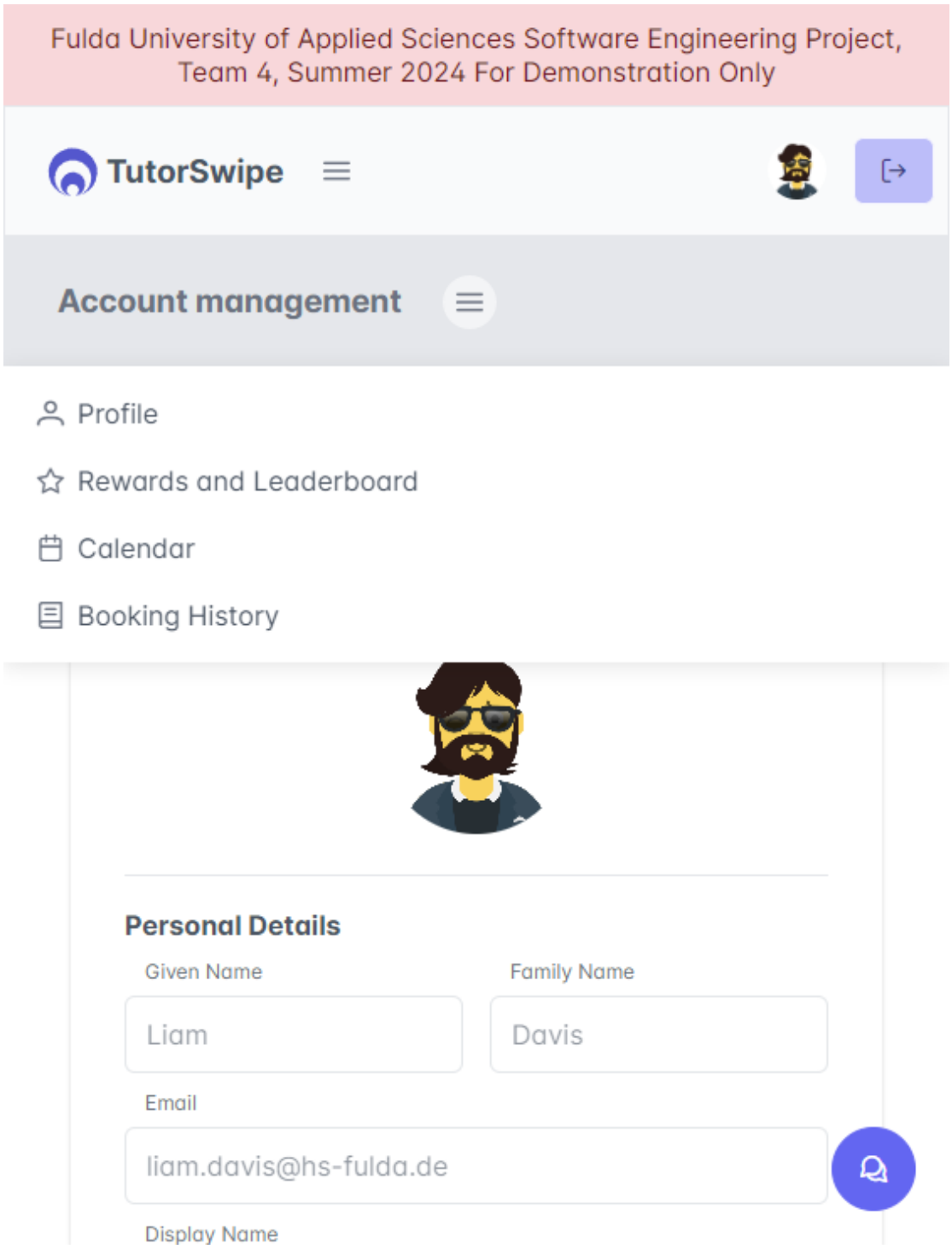


Figure 6: Tutor profile page (responsive view)

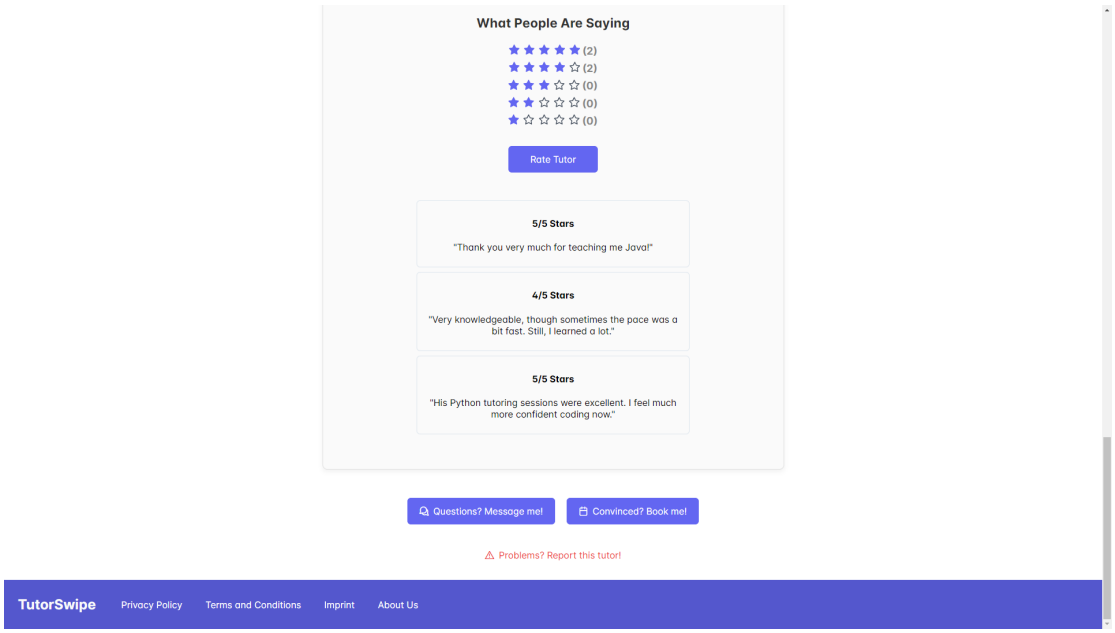


Figure 7: Ratings from students of tutor profile page

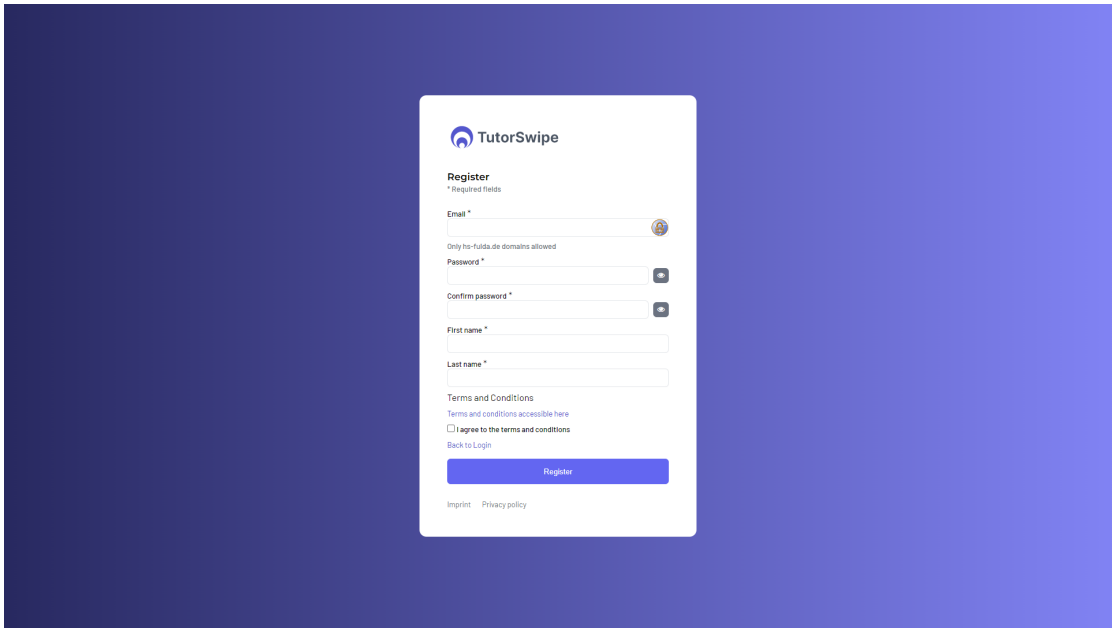


Figure 8: Signup mask of our Keycloak

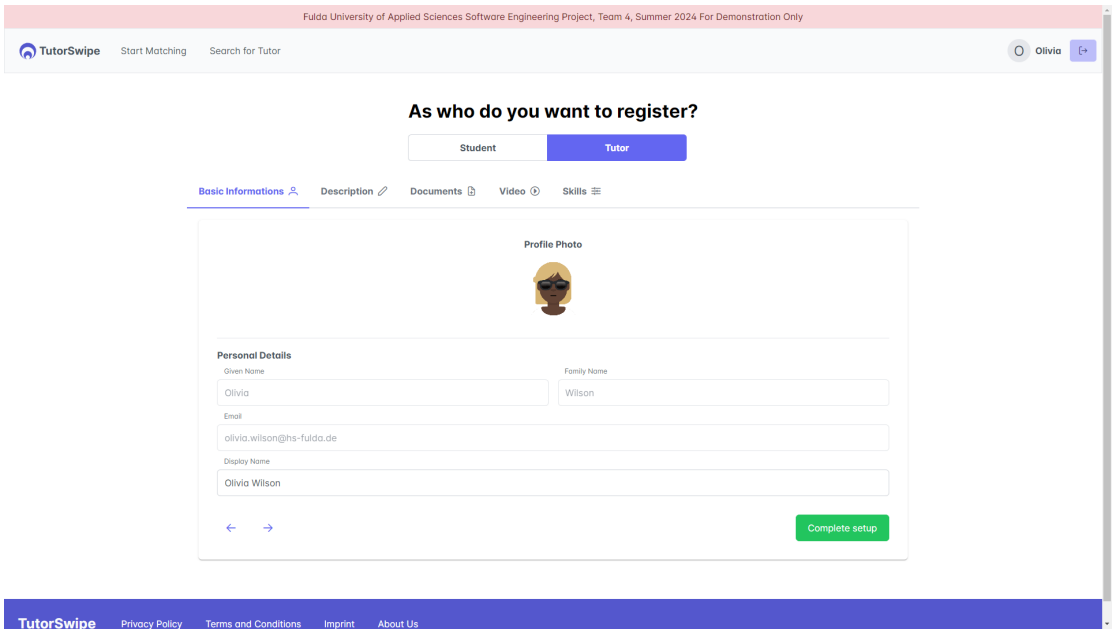


Figure 9: Onboarding wizard after Sign Up

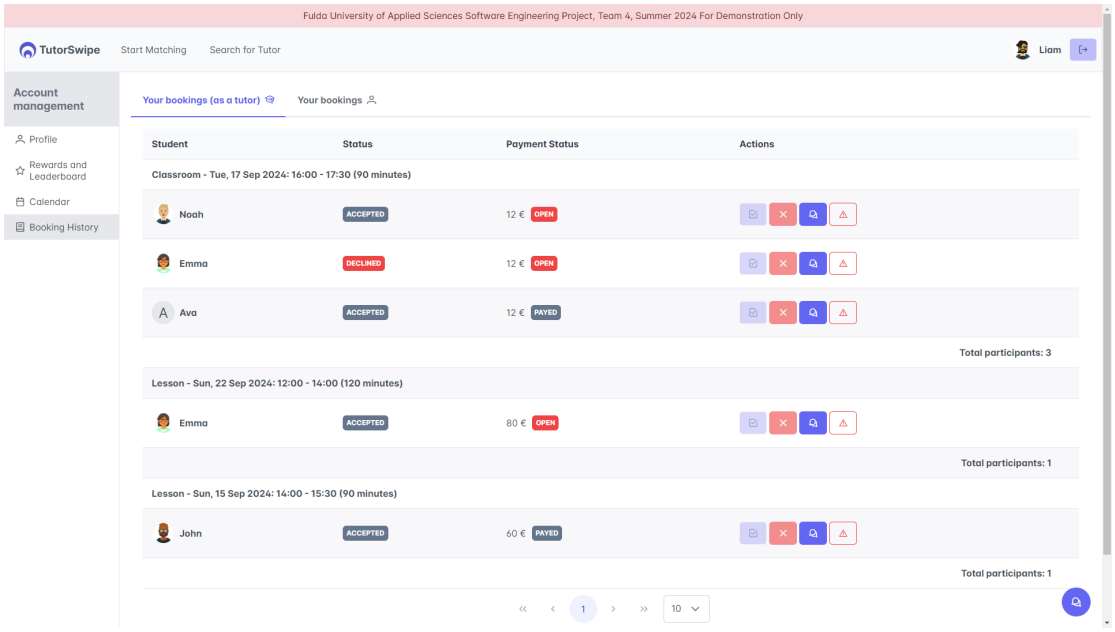


Figure 10: Booking history of tutor

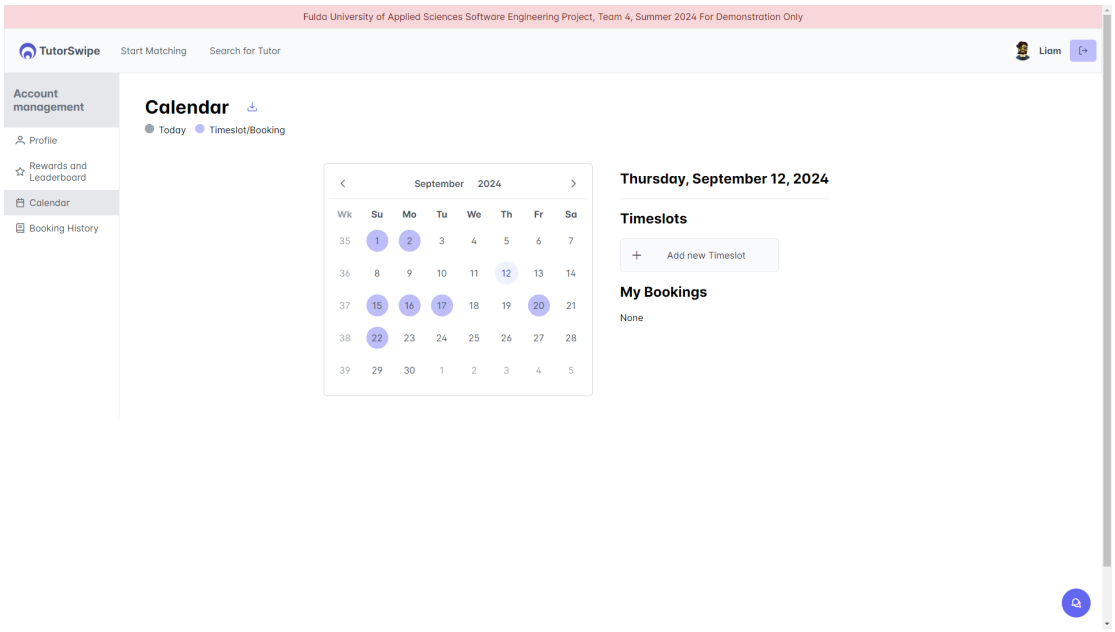


Figure 11: Calendar of tutor with option to export as .ics

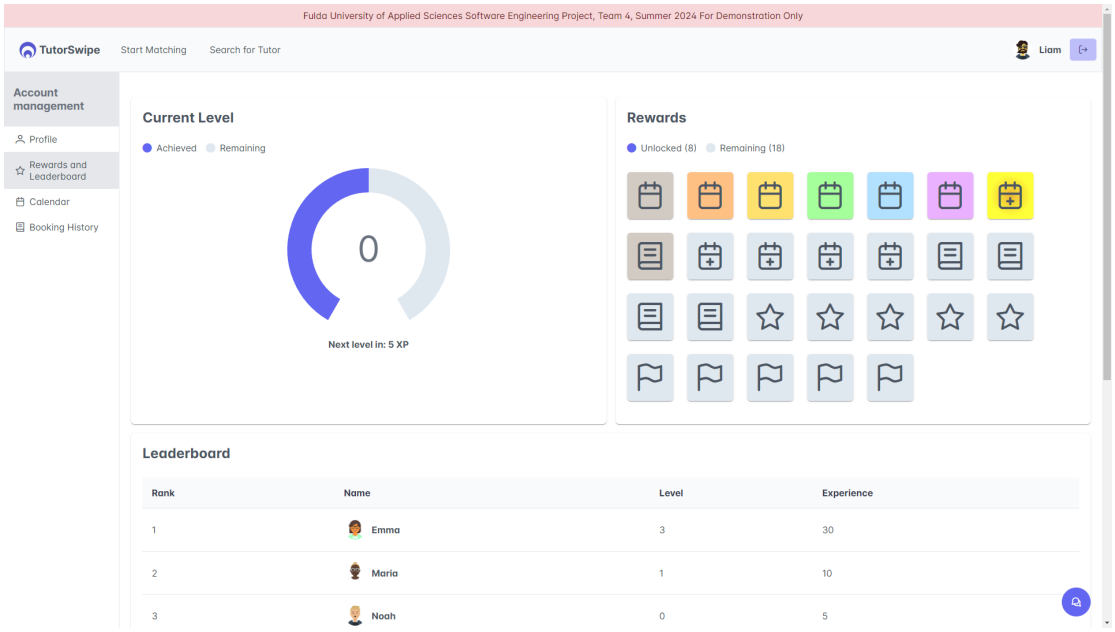


Figure 12: Global leaderboard with experience and levels

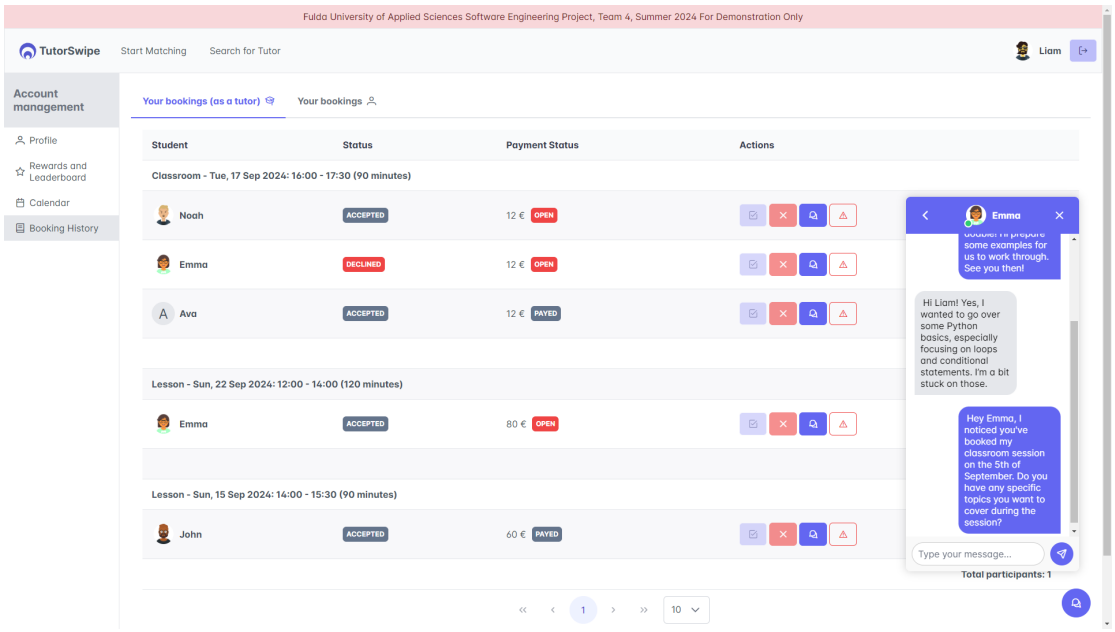


Figure 13: Real-time chat with online status of users

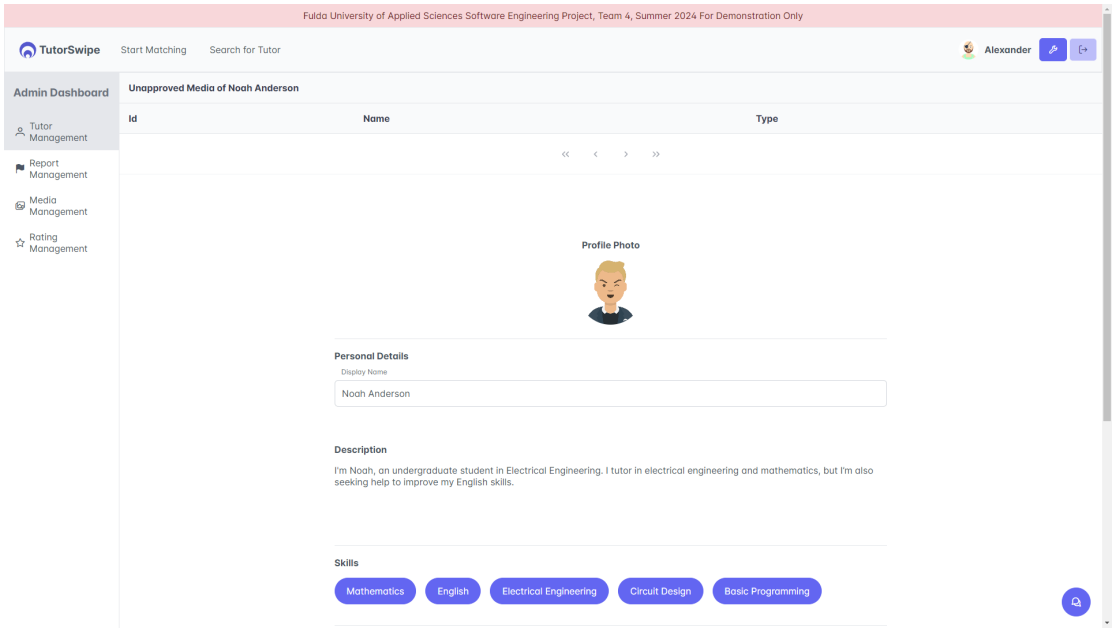


Figure 14: Admin Dashboard: Tutor Management

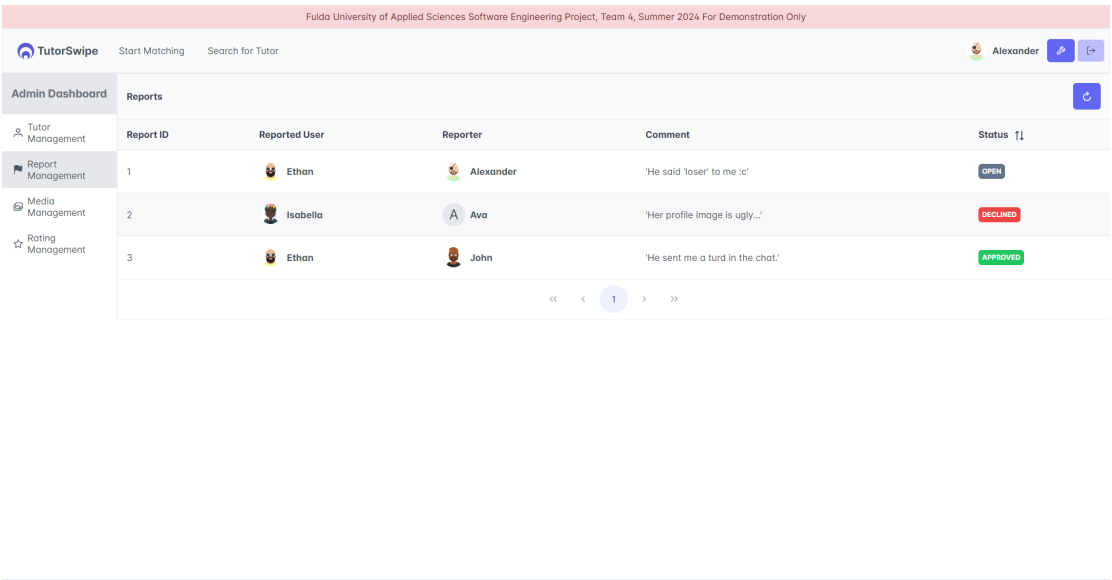


Figure 15: Admin Dashboard: Report Management

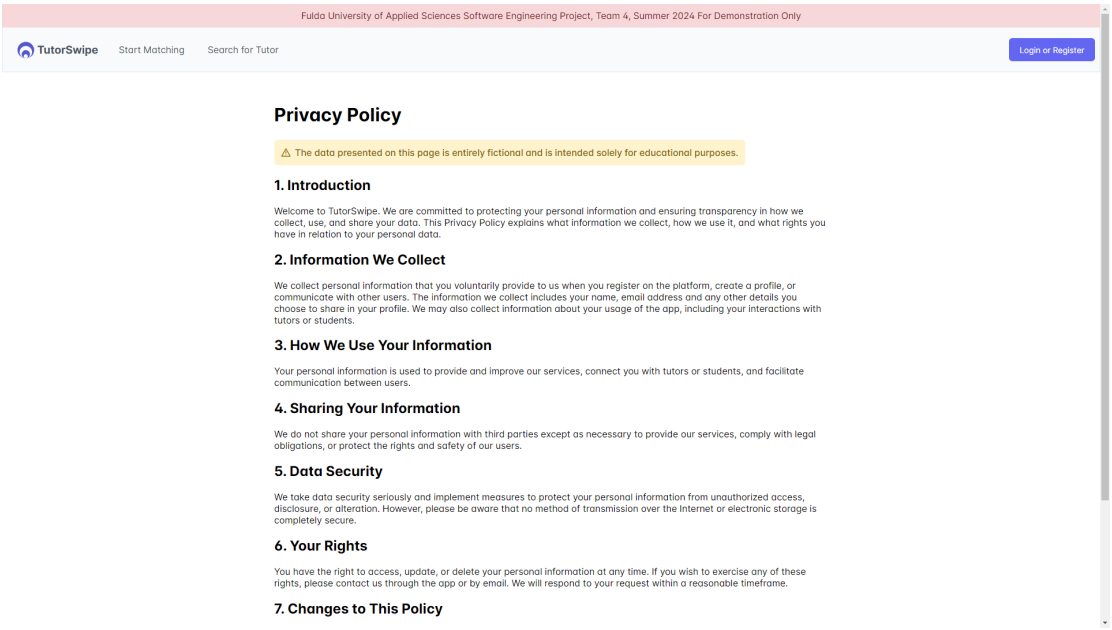


Figure 16: Privacy policy (Mock data)

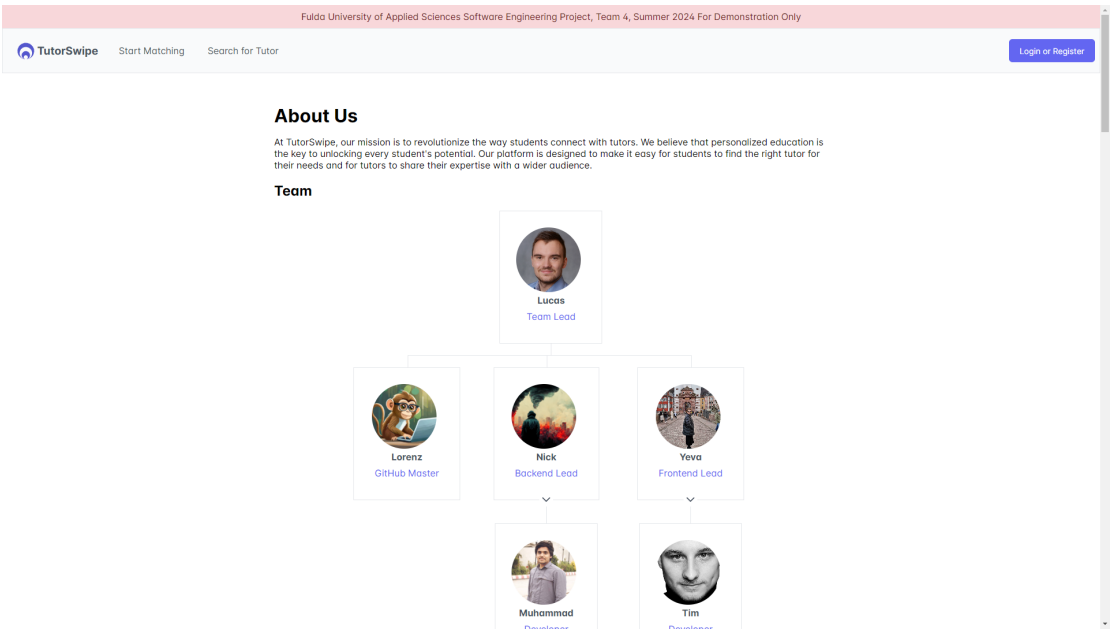


Figure 17: About Us page

3 Achieved functional requirements

We have fulfilled all initially defined (all priorities) functional requirements.

Priority 1 Requirements

Unregistered Users

Requirement ID: FR-04.1
Title: Browse Tutors with Video Swiping
Status: ACHIEVED

Requirement ID: FR-13.1
Title: Search and Filter
Status: ACHIEVED

Registered Users

Requirement ID: FR-01.1
Title: Register Users
Status: ACHIEVED

Requirement ID: FR-02.1
Title: Log In User
Status: ACHIEVED

Requirement ID: FR-03.1
Title: Manage User Profiles
Status: ACHIEVED

Requirement ID: FR-09.2
Title: Send Messages
Status: ACHIEVED

Requirement ID: FR-12.1
Title: Manage Media
Status: ACHIEVED

Administrators

Requirement ID: FR-16.1
Title: Approve Content
Status: ACHIEVED

Priority 2 Requirements

Registered Users

Requirement ID: FR-07.2
Title: Send and Receive Payments (mock)
Status: ACHIEVED

Requirement ID: FR-08.2
Title: Show Payment History
Status: ACHIEVED

Requirement ID: FR-10.2
Title: Schedule Meetings
Status: ACHIEVED

Requirement ID: FR-14.2
Title: Rate Users
Status: ACHIEVED

Requirement ID: FR-15.2
Title: Report Users
Status: ACHIEVED

Requirement ID: FR-19.2
Title: Classify Tutors
Status: ACHIEVED

Priority 3 Requirements

Registered Users

Requirement ID: FR-05.3
Title: Agree to Terms and Conditions
Status: ACHIEVED

Requirement ID: FR-06.3
Title: Onboard Users
Status: ACHIEVED

Requirement ID: FR-11.3
Title: Download Meetings Schedules
Status: ACHIEVED

Requirement ID: FR-17.3
Title: Earn Rewards
Status: ACHIEVED

Requirement ID: FR-18.3
Title: Gamify User Experience
Status: ACHIEVED

Requirement ID: FR-20.3
Title: Organize Classrooms
Status: ACHIEVED

4 How we worked as a team

We held weekly meetings, either at the university or on Discord, to discuss progress and coordinate tasks. Within the smaller backend and frontend teams, we had additional coordination to ensure smooth development. Our team lead was responsible for planning and allocating tasks. Deployment to the production server took place before each weekly meeting and was managed by the GitHub Master. We followed a branching strategy using feature branches and pull requests (PRs) to ensure code quality and collaboration.