Master Team Project Fall 2024
NeuAnfang
Team 3 (local)

Member Name	**Role**
Zubeena Shireen Sheikh zubeena-shireen.sheikh@informatik.hs-fulda.de	Team Lead and Document Master
Muhammad Zaid Akhter	Backend Lead
Masood Ahmed Mohiuddin	Frontend Lead
Humdaan Syed	GitHub Master
Andrii Kuripka	Team Member (Frontend)

Milestone 2
27/11/2024

Date Submitted	**27/11/2024**

Content and structure for Milestone 2 document for review by institutors:

1. Functional Requirements - Prioritised:

Following is a prioritised list of the functional requirements:

Priority 1 (Must have):

1. **Apartment Posting and Pricing:**

   1.1 Landlords can add detailed information about their properties, including rent, location, size, available amenities, and conditions.

2. **Share Properties:**

   2.1 Landlords can generate shareable links that can be shared on other social media platforms thereby increasing reach of their apartment.

3. **Ad Status Management:**

   3.1 Landlords can mark ads as draft, pending, active, rented, archived, or deleted.

5. **List View of Apartments:**

   5.1 The list view is sortable by filters and includes thumbnail images for quick browsing.

   5.2 Each property in the list view displays a summary of key information such as proximity to Fulda University.

6. **View of Apartments:**

   6.1 An interactive map shows property markers, allowing students to zoom in on specific neighbourhoods.

8. **Detailed Apartment View:**

   8.1 The detailed view includes images and an expanded list of amenities.

11. **Real-Time Chat:**

    11.1 Chat features include read receipts and user online status.

18. **Ad Approval by Admin:**

    18.1 The site admin reviews and approves each property before it becomes visible to students.

18.2 Admins can communicate directly with landlords to resolve flagged issues.

19. Landlord Dashboard with Analytics:

19.1 The dashboard displays basic metrics such as the total number of views, inquiries, and wishlists for each property.

19.2 Landlords can mark properties as archived, deleted or updated.

21. Advanced Filtering Options:

21.1 Filters include advanced criteria like apartment age, pet policies, etc.

21.2 Students can combine multiple filters for complex searches.

27. Student Subletting:

27.2 Students can specify furniture and utility usage conditions for sublets.

32. Student Account Verification:

32.1 Verification requires university email addresses.

32.2 Verified accounts are prominently tagged for enhanced trust.

33. Ad Search Feature

33.1 Students can search for properties using a search bar with keywords, such as type of apartment/room, location (e.g., "studio apartments", “single room”).

33.3 Results are dynamically updated when students refine their search using filters and sort options.

Priority 2 (Desired):

1. Apartment Posting and Pricing:

1.2 The system suggests a price range based on the average cost per square meter and local rental trends, which landlords can adjust based on their preferences.

1.3 Landlords receive prompts to complete missing details to improve property visibility and accuracy.

2. **Share properties:**

   2.2 Sharing options include direct integration with popular platforms (e.g., Facebook, Instagram, WhatsApp).

3. **Ad Status Management:**

   3.2 Notifications are sent to students when the status of a property they interacted with changes.

4. **User Profiles and Document Upload:**

   4.1 Students must complete a profile, including name, university email, study program, and contact information.

   4.2 Landlords must create a profile that includes name, email, phone number, and address for identity verification.

   4.3 Students can opt to add supplementary documents, such as prior rental references, to strengthen their application.

   4.4 The document upload feature accepts multiple formats (e.g., PDFs, images) and verifies them for authenticity.

6. **Map View of Apartments:**

   6.2 Additional overlays show amenities such as grocery stores and public transport stops.

7. **Save Search Criteria:**

   7.1 Students can save multiple sets of search criteria with custom names for easier recall.

   7.2 Notifications are triggered when new properties match saved criteria.

9. **Commute Information:**

   9.1 Provides estimated commute times using public transport, biking, walking, or driving to key locations like university, groceries, drugstore and hospitals.

   9.2 Links to public transport schedules and route planners are included for convenience.

10. **Wishlist Apartments:**

    10.1 Students can organise apartment properties into wishlists.

10.2 Alerts notify students of changes in the status or availability of wishlisted apartment properties.

12. Student Documents Sharing:

12.1 Students can choose which documents to share during the inquiry process, maintaining control over their data.

12.2 The system securely encrypts shared documents to protect sensitive information.

13. Chat File Sharing:

Supports drag-and-drop file uploads for easy sharing of documents and images.

14. Block/Unblock Users in Chat:

14.1 Blocking a user disables further communication and hides chat history.

14.2 Users can manage their block list from their profile settings.

15. Offer Apartment/Retract Offer in Chat:

15.1 Landlords can attach terms and conditions to offers made through chat.

15.2 Notifications are sent to both parties when an offer is retracted.

16. Accept/Reject Offer and Rate Landlord:

16.1 Students can leave detailed feedback and ratings for landlords after accepting or rejecting an offer.

16.2 Ratings influence a landlord’s visibility in searches.

17. View and Follow Landlord Profiles for Updates:

17.1 Landlord profiles display a cumulative rating and all active properties

17.2 Students are notified when landlords they follow post new properties.

19. Landlord Dashboard with Analytics:

19.3 Highlights the most popular properties and provides simple tips, such as "Add more photos" or "Adjust price for better visibility," to improve engagement.

20. Customizable Notifications:

Users can enable or disable specific types of notifications (e.g., new property alerts, status updates, or chat messages) individually.

22. In-App Apartment Viewing Scheduler:

22.1 Scheduling integrates with external calendars (e.g., Google Calendar).

22.2 Both parties can customise reminders for scheduled viewings.

23. Message Templates for Landlords:

Templates can include placeholders (e.g., #StudentName, #PropertyTitle) for personalization.

25. Matching Ads with Students:

25.1 Matches are prioritised based on how closely properties meet students' preferences.

25.2 Students receive a daily list of new matches.

27. Student Subletting:

27.1 Sublet properties are flagged as temporary and require landlord approval before posting.

28. Report ads:

28.1 Reporting options include categories such as "Fraudulent" or "Inappropriate Content".

29. LLM bot for Accommodation Queries:

29.1 The chatbot answers accommodation-related questions in English to assist students.

29.2 Provides general guidance on topics like rental processes, legal requirements, and housing tips.

30. Show interested/views per ad:

Metrics include unique views, number of inquiries, and shares.

31. Trending/Popular ads:

31.1 Properties with high engagement are tagged as "Trending" for enhanced visibility.

31.2 Students receive alerts for trending ads matching their preferences.

33. Ad Search Feature

33.2 The search feature supports natural language queries (e.g., "apartments under €500 with a balcony").

33.4 Search results highlight the best matches based on students' preferences, saved search criteria, and popularity of properties.

Priority 3 (Opportunistic):

8. Detailed Apartment View:

8.2 The detailed view includes 360-degree virtual tours (if available).

11. Real-Time Chat:

11.2 A "quick reply" function allows landlords and students to use predefined responses.

24. Neighborhood Information Section:

24.1 Displays reviews from previous tenants about the neighbourhood.

24.2 Includes details about safety and affordability.

26. Match Students for Landlord:

26.1 Landlords can view a dashboard of matched students ranked by compatibility.

26.2 Direct communication options are enabled for matched students.

28. Report ads:

28.2 Users are notified of the outcome of their reports.

32. Student Account Verification:

32.3 Verification requires confirmation of enrollment.

2. List of main data items and entities:

User is a generic term used for referring students and landlords together. The application has the following user types:

A **Guest** is an unauthenticated user. A guest can browse the platform and search for properties but they see a subset of the available information. Some information like the exact location of the apartment and landlords contact info is hidden from them.

A **Student** represents a verified user with an hs-fulda.de email domain. They form the primary user base of the platform, with capabilities to search properties, maintain a wishlist of preferred properties, communicate with landlords, and save their search preferences. Students can report inappropriate content and must be able to demonstrate their affiliation with Fulda University.

A **Landlord** is a verified user who owns or manages rental properties. They can create and manage properties, respond to student inquiries, and track the status of their properties. Landlords must provide valid contact information and maintain their properties, including updating availability and responding to offers from potential tenants.

The **Admin** holds system-level privileges and is responsible for platform integrity. They review and approve new properties, moderate user content, manage user accounts, and handle disputes. Admins also monitor platform activity and can generate system reports for operational insights.

Some other main data items in the application are defined as follows:

An **Property** is the central entity of the platform, representing a rental property ad. It contains comprehensive property information including location, price, amenities, and media content. Each property has a unique identifier and maintains various states from creation through to rental or archival.

The platform offers two primary viewing modes: **List View** provides a structured, scannable format showing apartment summaries with key details and quick actions, while **Map View** offers a geographical representation of available properties, featuring interactive markers and distance-based searching from campus.

Search Criteria encompasses the filtering and sorting parameters users can apply to find suitable apartments. This includes price ranges, location preferences, property features, and distance from campus. Users can save their search profiles for future use.

The **Chat** system facilitates direct communication between students and landlords. Each conversation is tied to a specific property and maintains a complete message history. The system supports basic attachment sharing while adhering to platform security guidelines.

An **Offer** represents a formal expression of interest from a student to a landlord regarding a specific property. It includes proposed terms and maintains various states from initial submission through to acceptance or rejection.

The **Wishlist** feature allows students to save and organize their favourite properties. Users can add personal notes and set alert preferences for saved properties.

A **Review** is feedback from the student given to the landlord. It includes a rating and a comment about the student's experience with the landlord.

A **Landmark** is a place of interest like grocery stores, shopping centers, hospitals, gyms etc. They are used to prepare a neighbourhood guide showing routes and commute times from apartments to landmarks.

A **Report** is a flag by the user. A user can flag content on the site and report it to the admin, the admin can review this content for violation of site policies.

Notifications keep users informed of relevant platform activity. These can be system alerts, chat messages, or property updates. Each notification has a priority level and expiration date where applicable.

A **Session** tracks user activity on the platform, maintaining security and authentication state. It includes login information, device details, and activity timestamps to ensure secure access to the platform.

A **Template** is a quick way for landlords/students to chat. Users can predefine long messages and use it quickly in a chat scenario.

An **Appointment** refers to an apartment viewing appointment between a landlord and a student. **Reminders** can also be added to appointments.

The platform also maintains several status categories to track the state of various entities:

Property Status:
Properties can be Pending (awaiting approval), Active (available for rent), Rented (transaction completed), Archived (no longer available), or Rejected (failed approval).

User Status:
User accounts may be Pending Verification, Active, Suspended, or Deactivated, reflecting the user's standing on the platform.

Chat Status:
Chat conversations can be Active, Archived, Blocked, or Reported, managing the flow of communication between users.

Offer Status:
Offers progress through states including Pending, Accepted, Rejected, Expired, or Withdrawn, tracking the rental negotiation process.

3. UI Mockups and Storyboards:

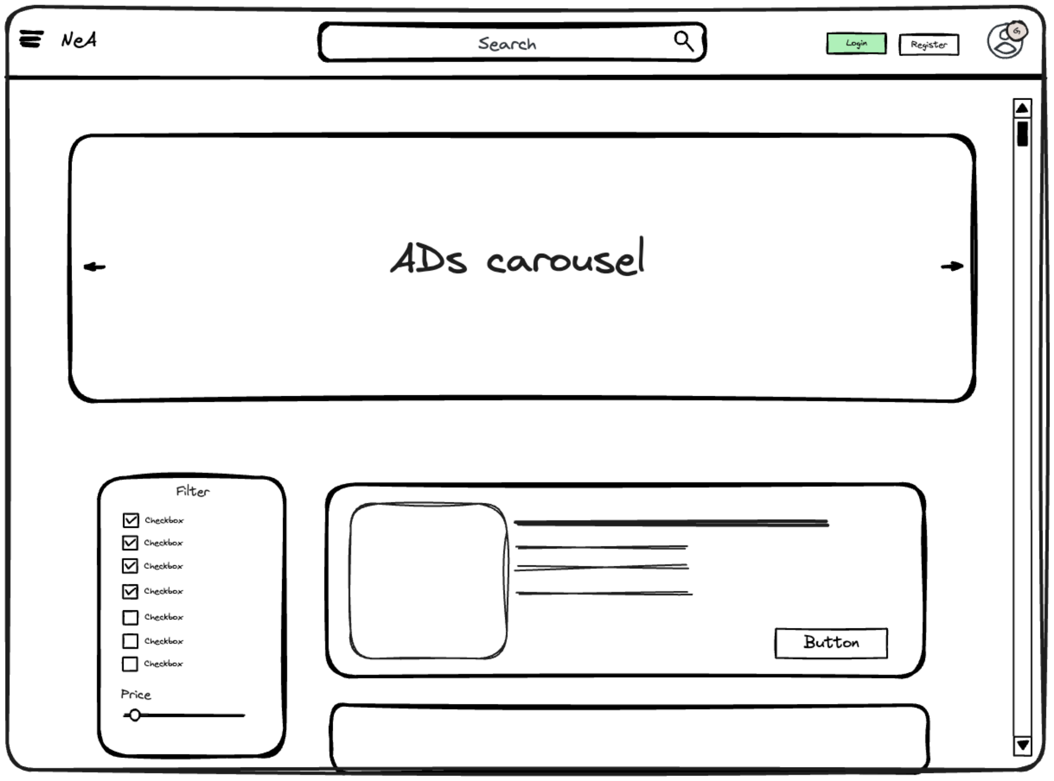


Fig 3.1: Home page (Guest)

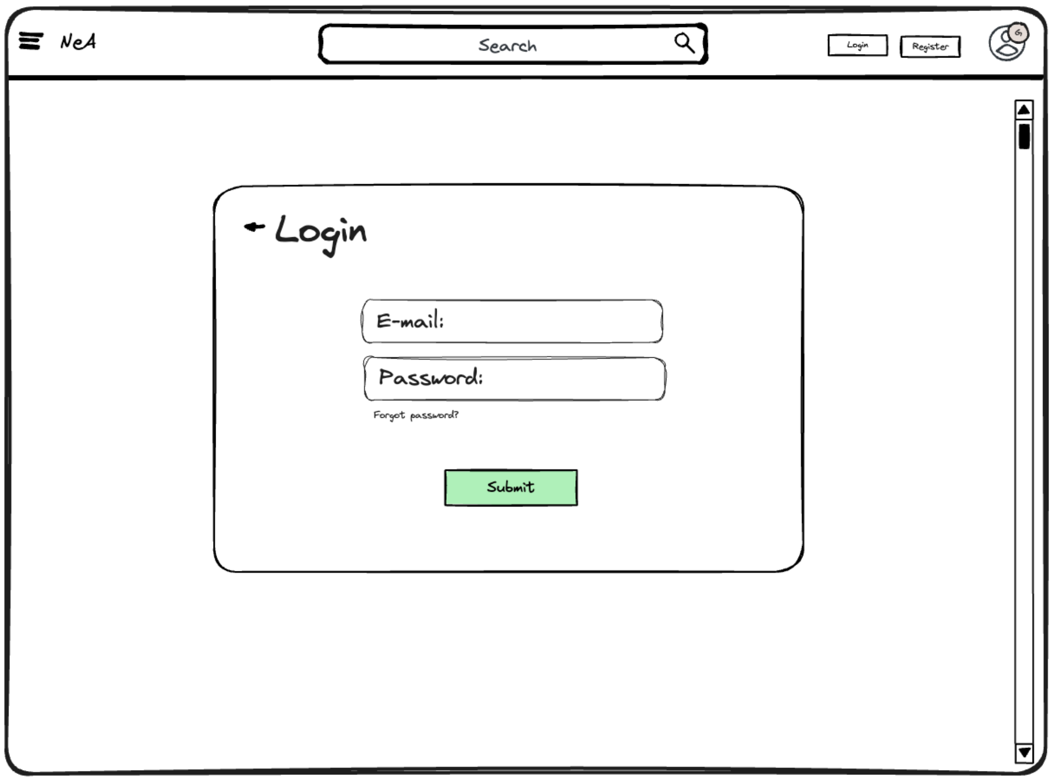


Fig 3.2: Login Page

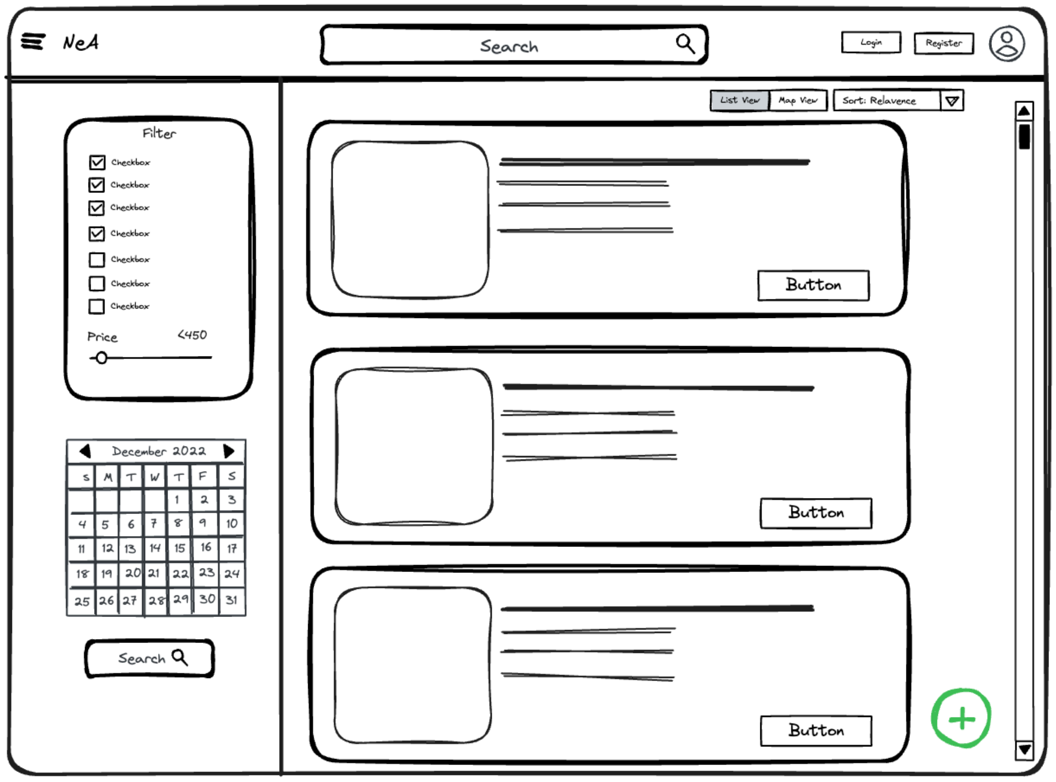


Fig 3.3: Search/List View (Landlord)

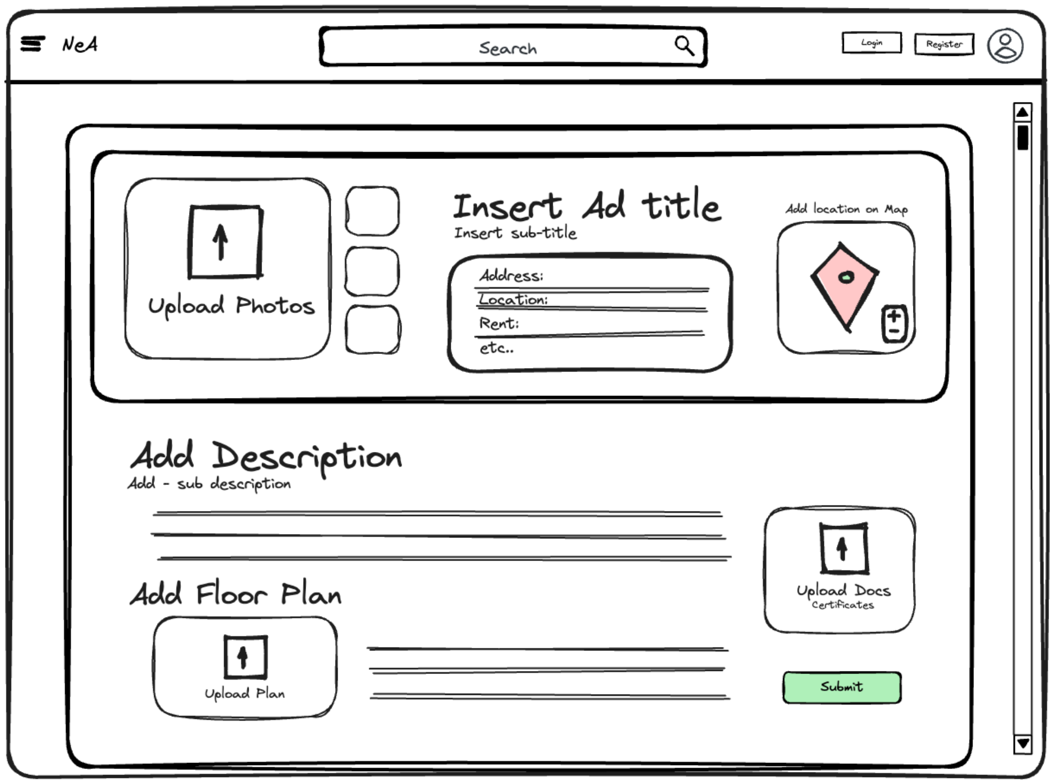
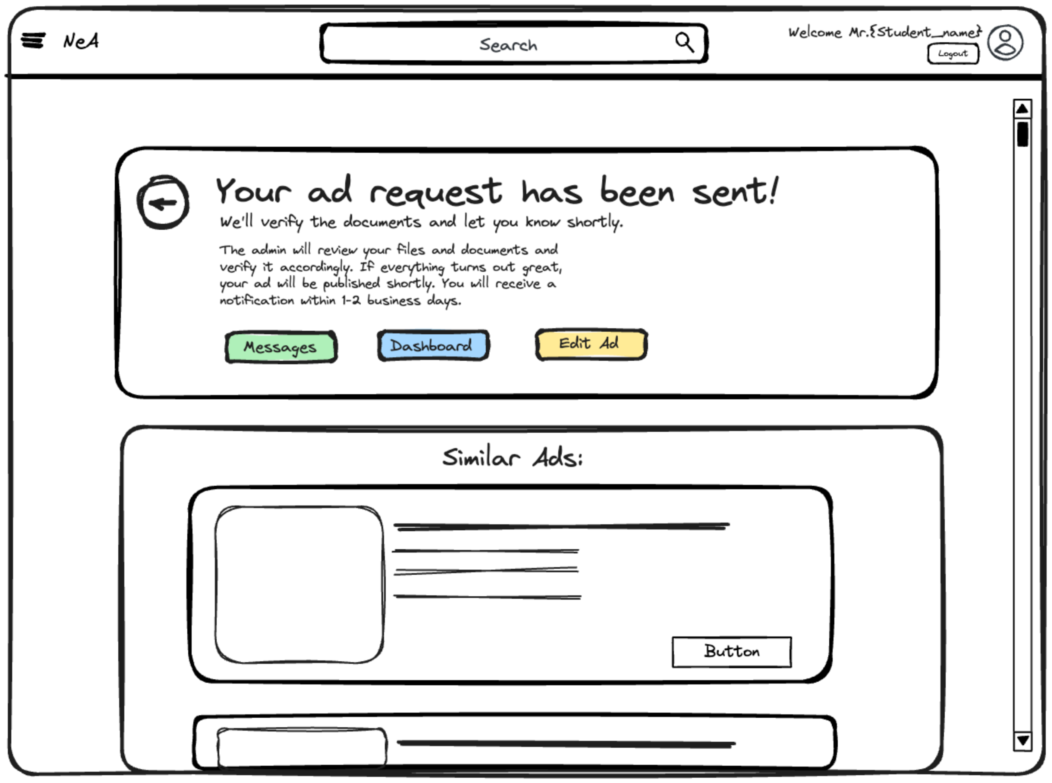


Fig 3.4: Create Property Page



Your ad request has been sent!

We'll verify the documents and let you know shortly.

The admin will review your files and documents and verify it accordingly. If everything turns out great, your ad will be published shortly. You will receive a notification within 1-2 business days.

Fig 3.5: Property listing Submitted for Review

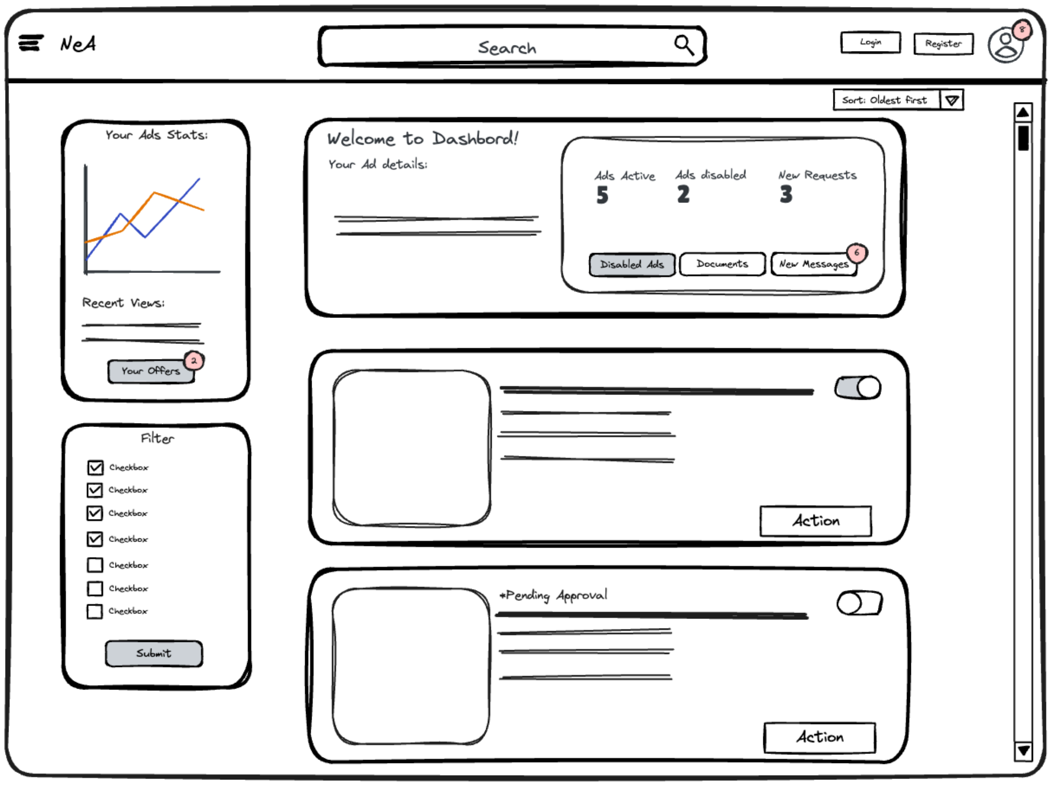


Fig 3.6: Landlord Dashboard

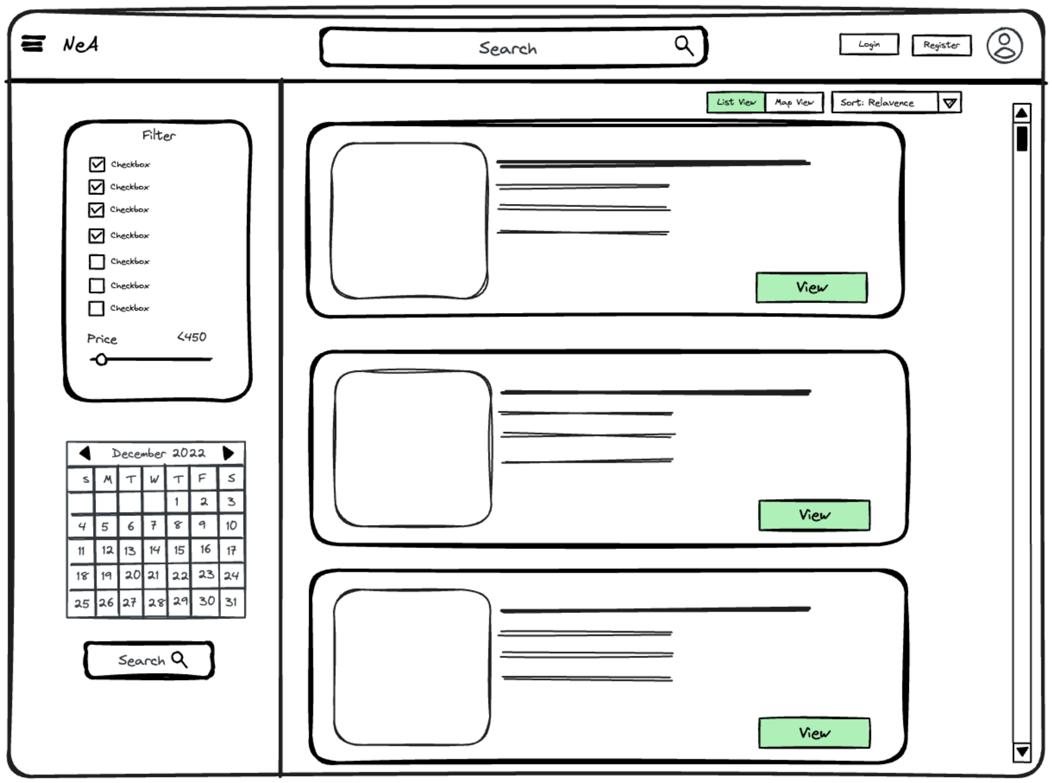


Fig 3.7: Search/List View (Student)

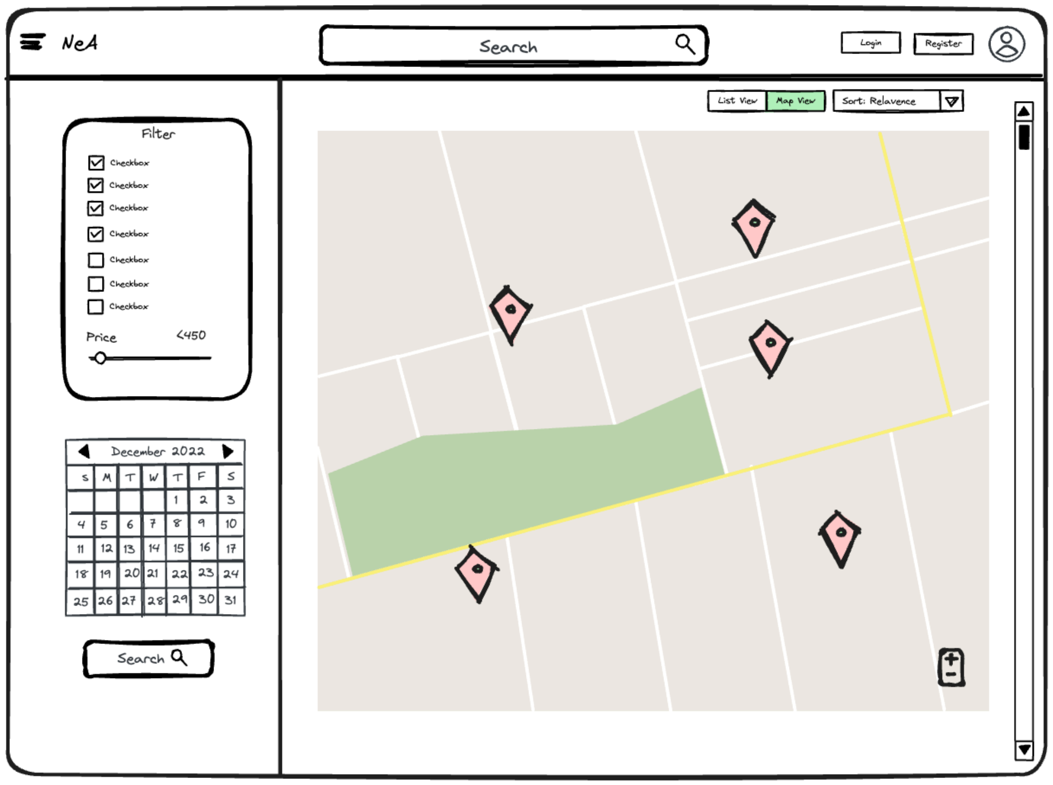


Fig 3.8: Map View

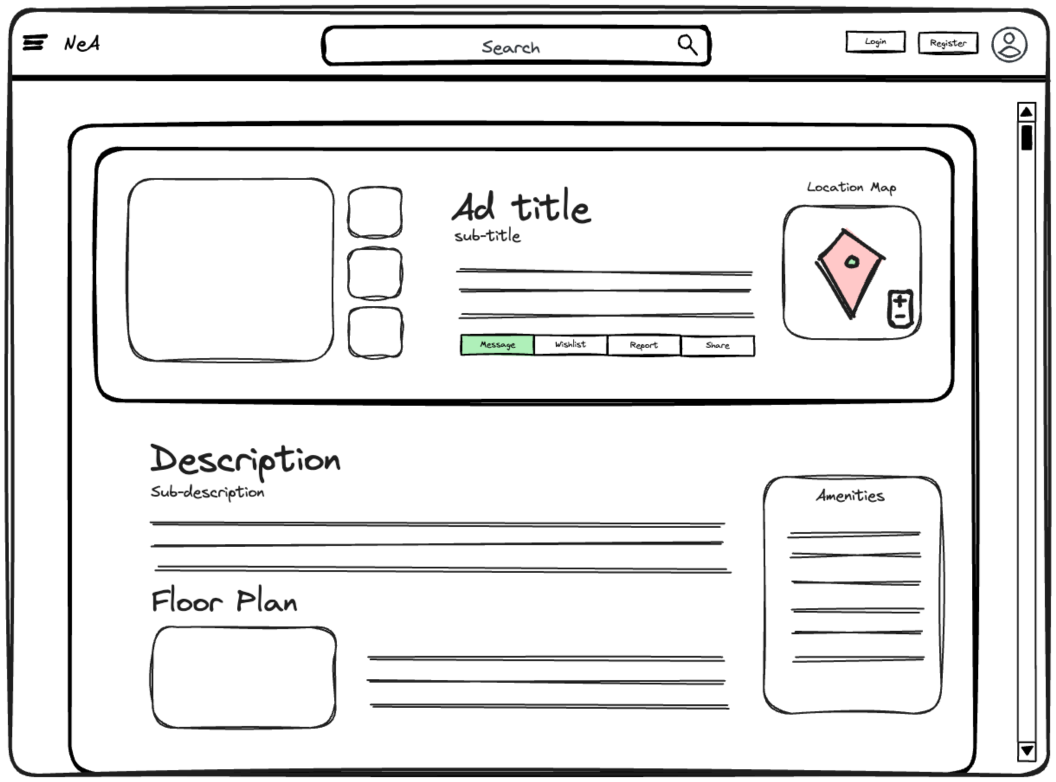


Fig 3.9: Property Details Page

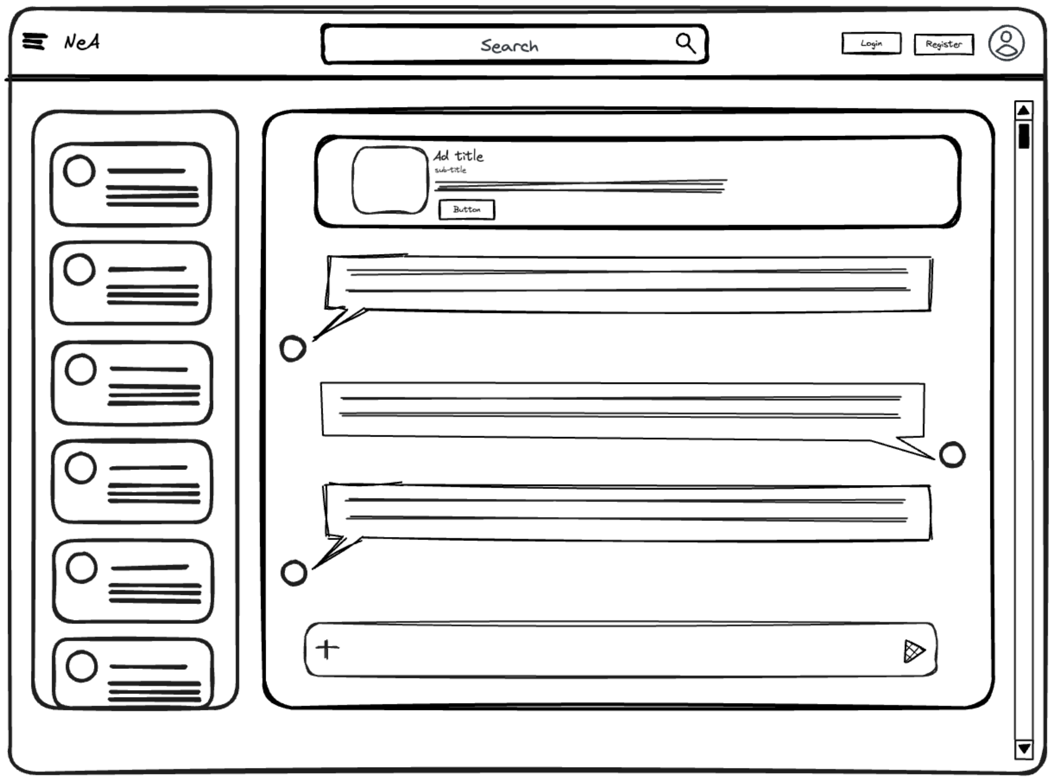


Fig 3.10: Real-time chat

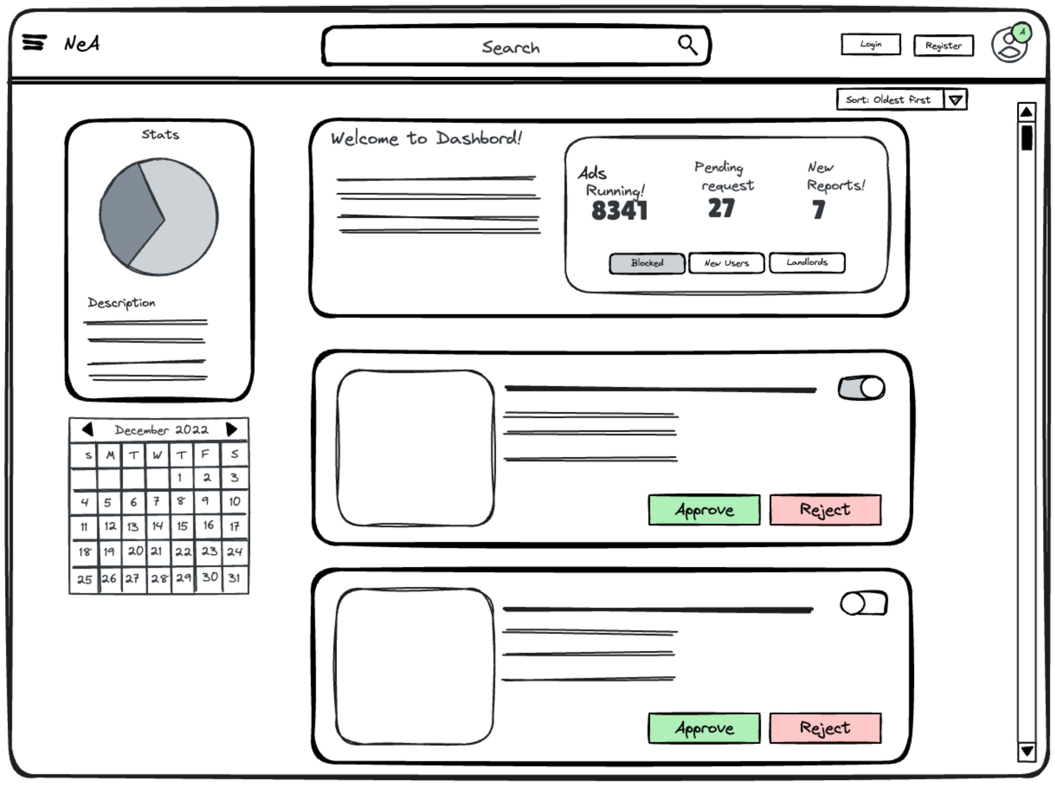
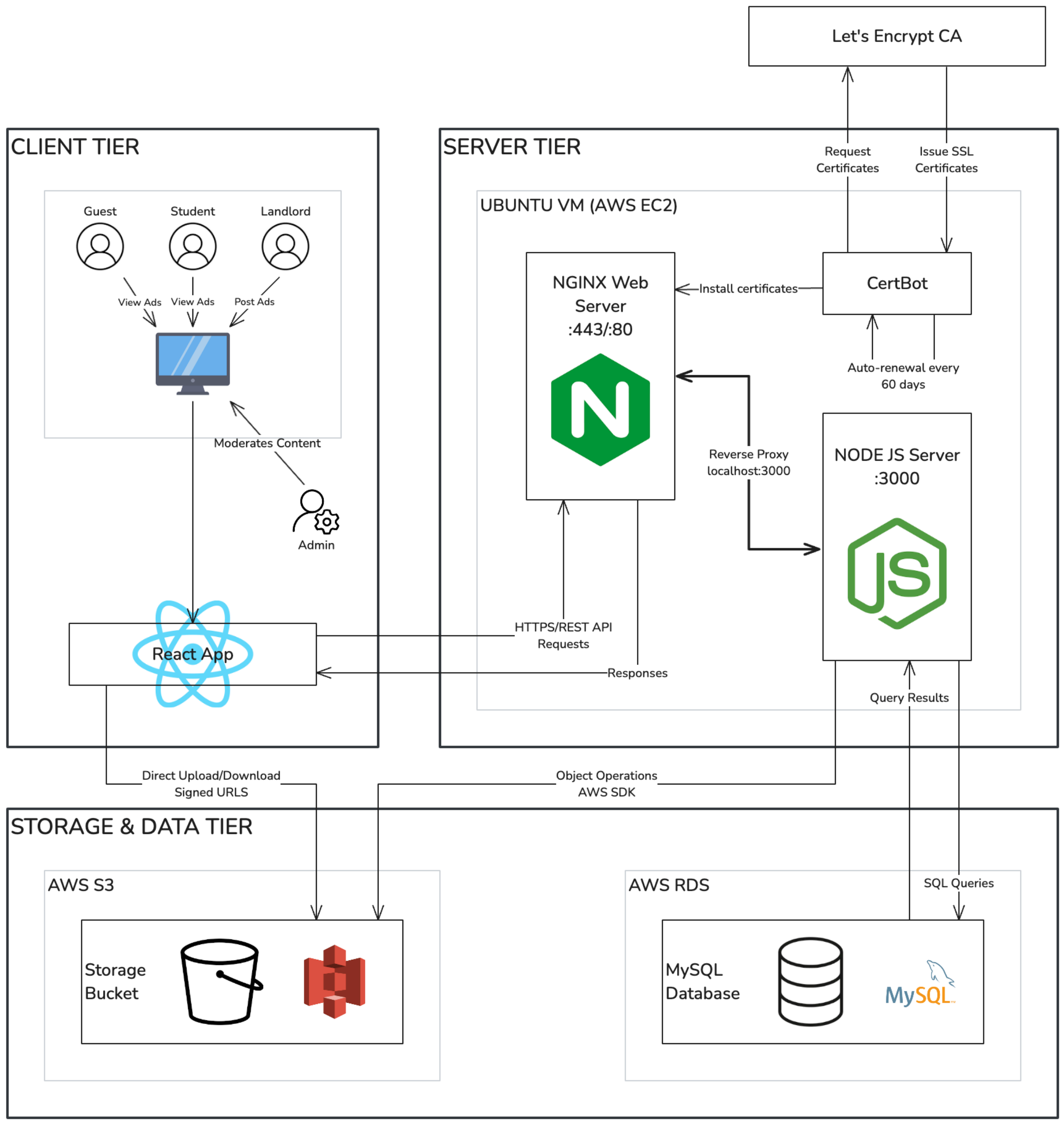


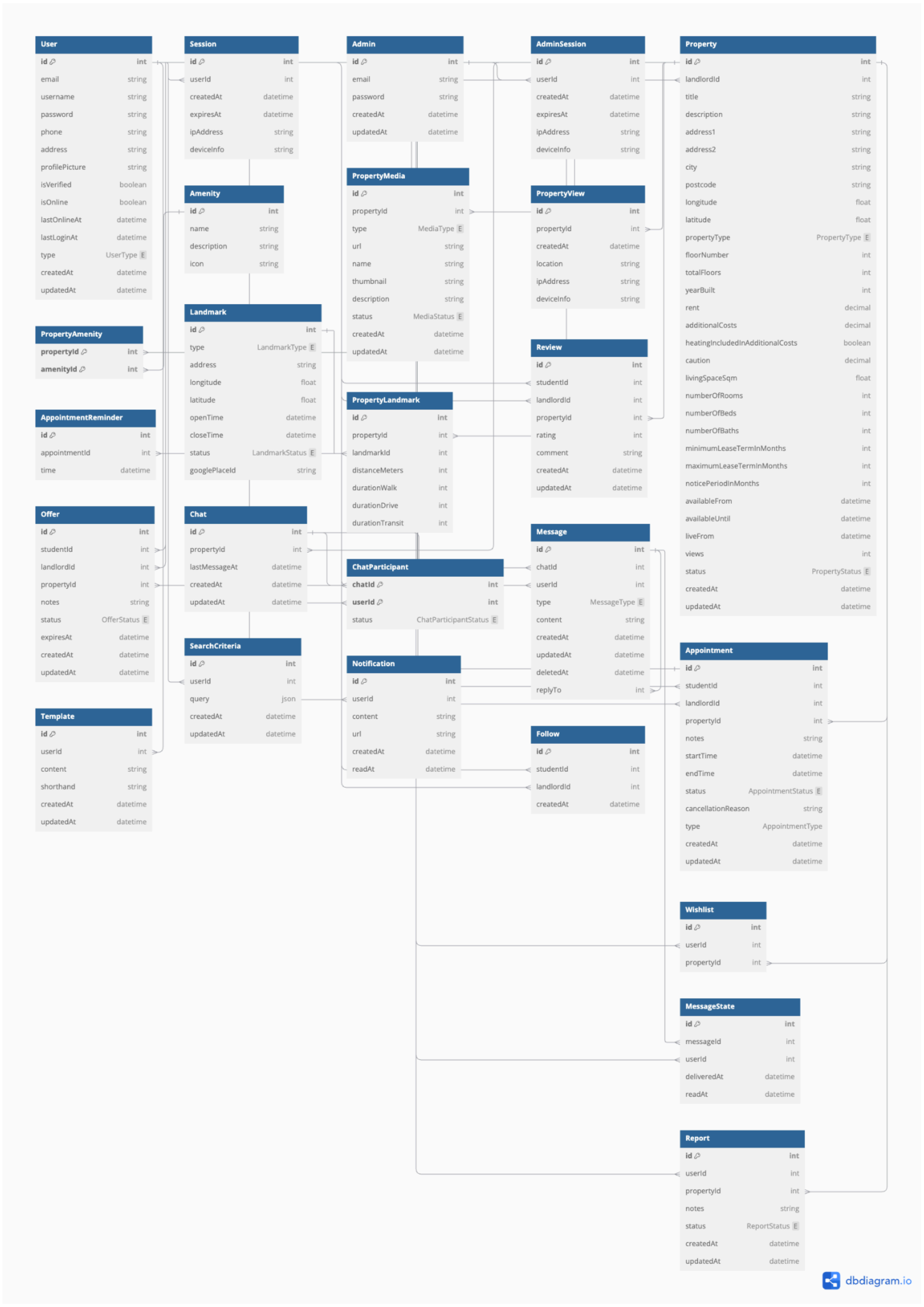
Fig 3.11: Admin Dashboard

4. High level Architecture, Database Organization:

High level Architecture:



Database Schema:



Following are the key entities defined in the above schema:

1. **Users & Authentication**
    a. User table is central with two types: STUDENT and LANDLORD
    b. Authentication handled through Session table
    c. Separate Admin and AdminSession tables for administrative access

2. **Property Management**
    a. Property is a core table linked to LANDLORD users
    b. Properties have detailed attributes (size, rent, location, availability)
    c. PropertyStatus tracks lifecycle (DRAFT → ACTIVE → RENTED etc.)
    d. Enhanced by:
        i. PropertyAmenity for features/facilities
        ii. PropertyMedia for photos/videos
        iii. PropertyView for tracking visits
        iv. PropertyLandmark for nearby points of interest and neighbourhood guide

3. **Interaction System**
    a. Appointment for apartment viewings
    b. Chat system with:
        i. ChatParticipant for user involvement
        ii. Message for communication
        iii. MessageState for read/delivery status
    c. Offer system for rental proposals
    d. Review for apartment/landlord ratings

4. **Supporting Features**
    a. Landmark for points of interest (schools, transport, etc.)
    b. Wishlist for saved properties
    c. SearchCriteria for saved searches
    d. Template for message templates
    e. Notification for user alerts
    f. Follow for student-landlord connections
    g. Report for issue reporting

Media Storage:
The application will use S3 to store all media such as images, files, and videos. The S3 Bucket is organized in the following structure:

```
📁root/
├── 📁chat/
│   └── 📄...
├── 📁user/
│   └── 📄...
└── 📁public/
    ├── 📁property/
    │   └── 📄...
    └── 📁user/
        └── 📄...
```

The **root** folder of the S3 bucket contains private folders like **chat** and **user**. These folders store private files such as chat attachment or private user files. These files can only be accessed via a temporary signed URL that the system generates.

Within the root folder, there is a **public** folder. All content within this folder is public and accessible via long-lived URLs. Users can access content within this folder if they have a URL but not list all the contents of the folder. This folder stores public content like **property** images and **user** profile images.

The above structure is implemented by making the bucket private and adding a policy (like below) to make the public folder accessible:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicRead",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::gdsd/public/*"
        }
    ]
}
```

Search Implementation:
For the vertical prototype search is implemented in a very simple manner by leveraging SQL queries only. We are using an ORM called prisma to manage all database queries so a typical query to search properties with rent between €400 - 800 looks like this:

```
const properties = await prisma.property.findMany({
  where: {
    totalRent :{
      gte: 400,
      lte: 800,
    }
  },
});
```

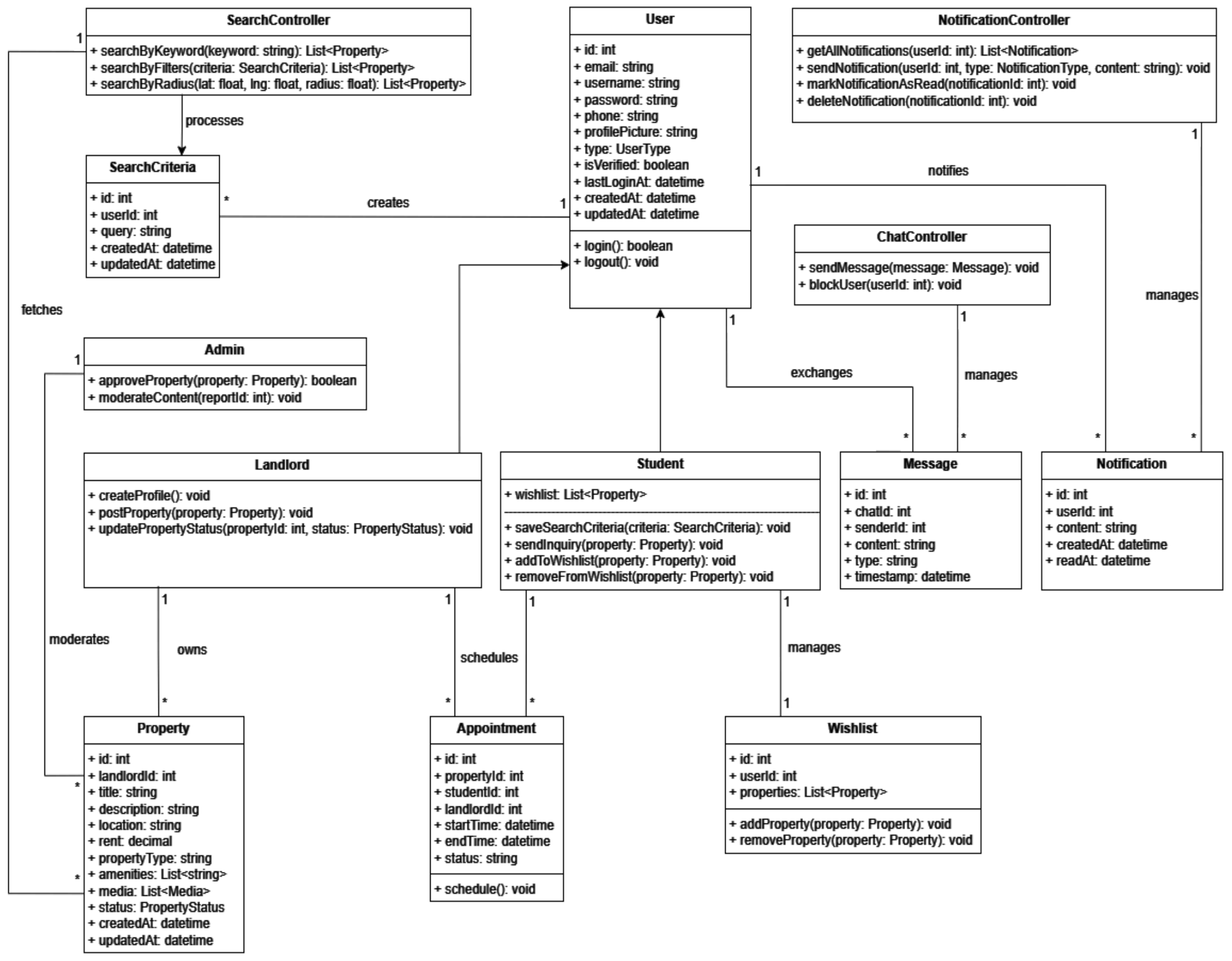
For the P2 implementation we want to develop a custom algorithm that is able to:

1. **Rank search results for users**:
   The algorithm will be able to display a feed that is tailored to each individual user. Each ad will receive a score based on the landlord rating, wishlist count, views, and user search criterion matching index (a score based on saved search filters). We will display the ads with the highest scores first in the results list.

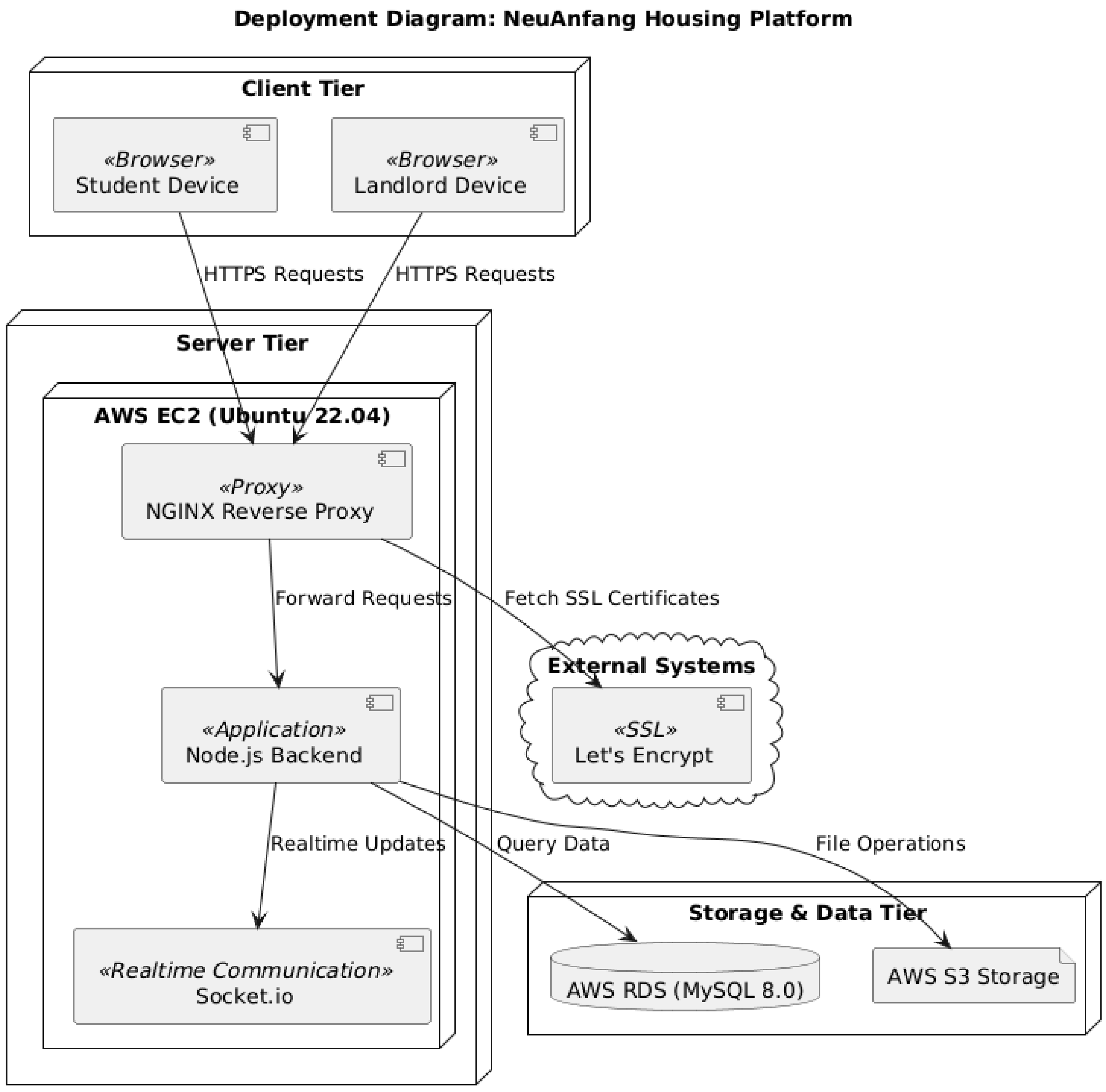
2. **Match students and landlords**:
   When a new ad is created, landlords receive a list of students based on the most recent saved search criteria. We can also obtain preferences through user interactions on the site; however, saved searches will serve as the primary source of data.

5. High Level UML Diagrams:

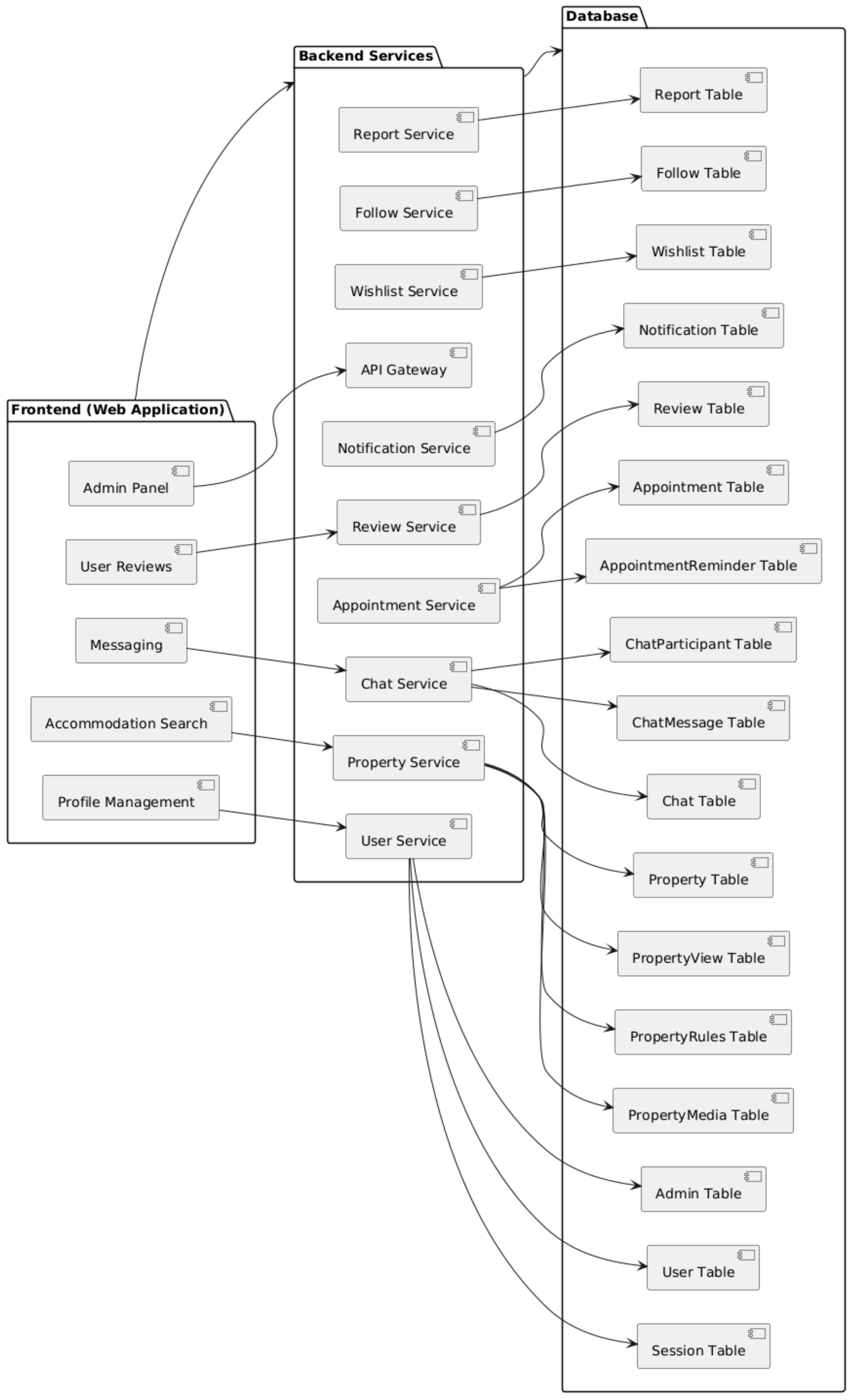
a) UML Class Diagram



b) UML Component and deployment diagram



Component Diagram: NeuAnfang Platform



6. Key risks for project at this time:

1. **Skills risks**
   Risk: Not all the team members are proficient with tools and tech stack required for the project.
   Plan for resolution:
   - Experienced members help others to catch up.
   - Team members dedicate time for self-learning during free hours.
   - Leads provide guidance and share expertise.

2. **Schedule Risks**
   Risk: There is a risk of missing deadlines due to resource constraints or unforeseen challenges.
   Plan for resolution:
   - Commit only to realistic goals.
   - Classify additional tasks as lower priority (P2) and handle them accordingly.
   - Allocate buffer time in the schedule to handle unforeseen challenges or changes.
   - Check status in the middle of the week to reassess timelines if needed.
   - Use project management tools for clear deadline visibility

3. **Availability Risks**
   Risk: A team member may become unavailable due to personal or health reasons.
   Plan for resolution:
   - Notify the team in advance to adjust plans.
   - Clear dependencies before the absence to minimise disruption.
   - Ensure key tasks are not dependent on a single individual
   - Maintain clear documentation for tasks to allow others to refer if needed.

4. **Technical Risks**
   Risk: Unforeseen technical issues, such as server failures or software dependencies, may arise.
   Plan for resolution:
   - Refer to documentation, Stackoverflow and other resources for troubleshooting.
   - Involving backend and frontend leads to resolving bottlenecks.
   - Communicate issues promptly and allocate buffer time for potential risks.

5. **Teamwork Risks**
   Risk: Communication issues, misunderstandings, or lack of coordination may disrupt teamwork.

Plan for resolution:

- Plan and discuss tasks thoroughly before starting work.
- Using tools like Trello to maintain transparency
- Ensure all members are on the same page and understand their roles.
- Foster respectful and open communication within the team.
- Encourage team members to take ownership of tasks and support one another when needed.

6. **Legal/Content Risks**

   Risk: Use of images, data, or software in the project may violate copyright or licensing terms which can potentially lead to legal issues.

   Plan for resolution:

   - Verify that all images, data, and software used are either copyright-free or properly licensed.
   - Use dummy data or publicly available datasets for testing and development.
   - Regularly review copyright and licensing guidelines to ensure compliance.

7. Project Management:

How We Managed Tasks So Far:
For M2 and before, we utilized **Trello** for task management. Tasks were first listed in the backlog. Then we assigned them to team members based on their expertise and interests. Each task was categorized as *To Do*, *Doing*, or *Done* to track progress. We also created a *Resources* section in Trello, where team members could access key documentation and important information pertaining to the project.

For team communication:
We used **WhatsApp** for quick updates and **Google Meet** for online meetings.

Each week we had the following structure:

1. **Planning Meeting**: Held at the start of the week to discuss tasks and responsibilities. Preferred offline meetings but also online based on everyone's availability.
2. **Progress/Status Check**: Once a task was completed, team members would notify the group via WhatsApp. If required, we held quick midweek meetings (online/offline) to assess task progress and address any blockers.
3. **Readiness Check**: Held a day before the deadline to review if tasks were completed as required.

Moving forward, we will continue using the above discussed tools and methodologies we have practised so far for team communication and task management. Additionally, we will have clear documentation of individual programming tasks, handle critical tasks in the absence of team members, and define clear interfaces for frontend and backend communication.