CS-319

Deliverable 4

Team 06

Group Members

Muhammad Rowaha - 22101023

Maher Athar Ilyas - 22001298

Mehshid Atiq - 22101335

Ghulam Ahmed - 22101001

Ismail Özgenç - 22001648

Date: 18/11/2023

1. Design Goals

1.1. User-friendliness:

The foremost design goal is prioritizing usability, ensuring a seamless and intuitive user experience. Users should be able to explore the platform effortlessly. One of the fundamental features implemented to fulfill this objective is that users will have persistent login sessions, eliminating the need to log in anew upon each return to the platform. Moreover, the user interface will be designed for easy navigation so that all core functionalities, including item posts, should be accessible within three clicks. This emphasis on usability aims to create a user-friendly and welcoming platform, encouraging active engagement. Furthermore, by allowing users to subscribe to specific product types (tags) and receive real-time updates regarding purchases and rentals, the app aims to cater to their individual preferences and interests. Moreover, all real-time updates will be provided using WebSockets that automatically update the state of the UI. This means that the user does not have to reload the page to check for new messages or notifications. All these design choices enhance the user experience by providing personalized and relevant information to the user in the most effortless way.

1.2. Reliability:

The secondary design goal focuses on reliability, ensuring the robustness and stability of the CampusConnect platform. Users should be able to rely on the system for consistent performance and trustworthy transactions. The system should be engineered to proactively mitigate the occurrence of simultaneous purchases or rentals of the same product by multiple users. To achieve this, the platform introduces specific states for products and forum posts. These states, like hidden, available, rented, etc, ensure that only items that are available should be visible to the users. This prevents inconsistent search results and forum views across different users. Moreover, we have converted all complex transactions into custom atomic operations and then implemented transactional logic over them. This allows us to maintain a stable state for the database even when a particular transaction (action) fails after being partially completed. CampusConnect aims to build user trust by prioritizing reliability and providing a dependable platform for second-hand sales, donations, borrowing, and related activities.

2. Design Trade-offs

2.1. User Experience vs Security

The aim of the platform is to prioritize a smoother user experience during subsequent logins. This goal is achieved by opting for a trade-off that involves sacrificing some security measures, such as continuous two-factor authentication (2FA), in favor of a more streamlined user experience. While continuous 2FA undeniably enhances overall security, the decision not to implement it at every login is rooted in the recognition that doing so could introduce unnecessary user friction and potentially disrupt the smooth flow of regular interactions with the platform. That’s why 2FA is specifically implemented during the initial sign-up phase only.

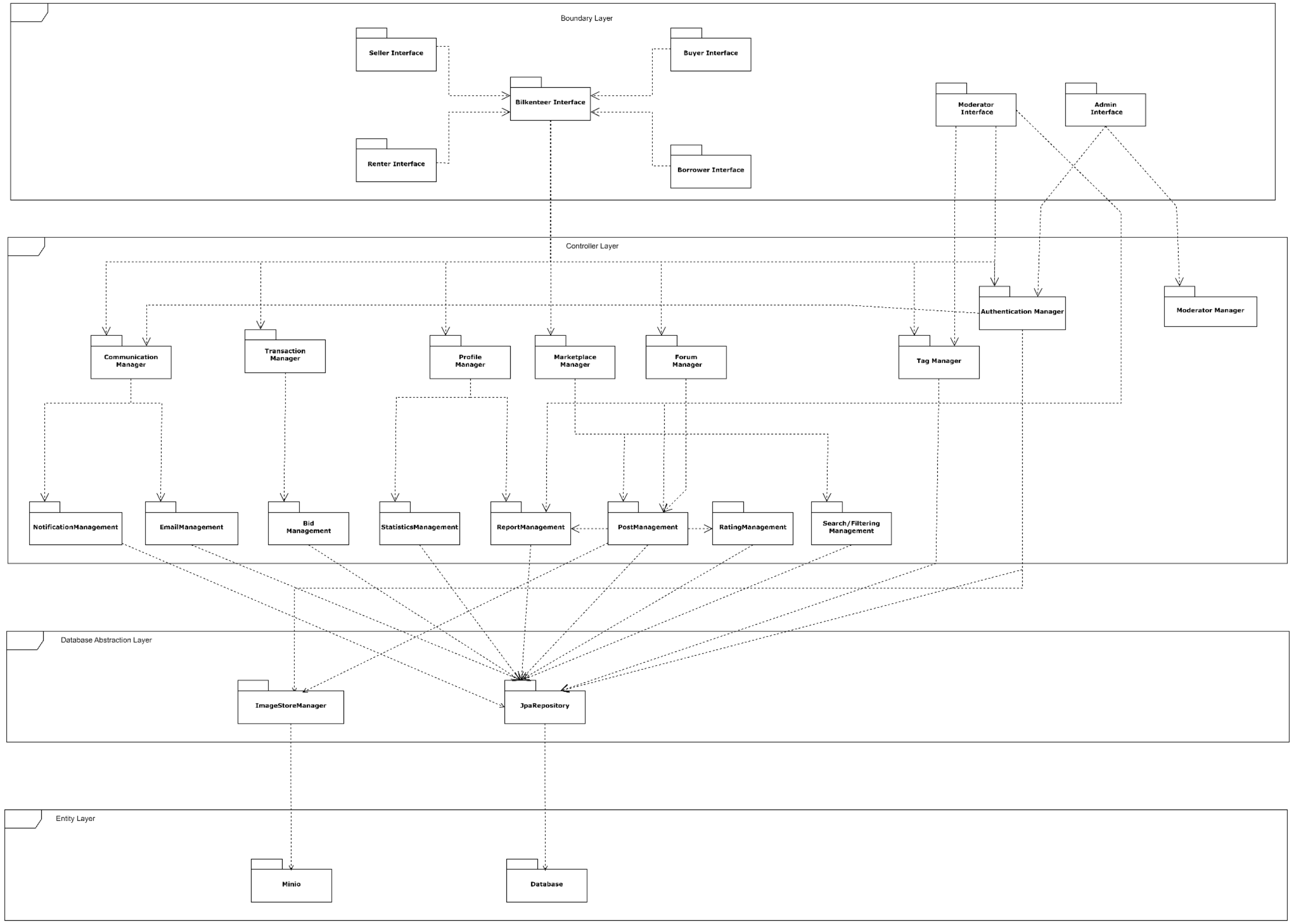
2.2. Performance vs. Security

The design decision for CampusConnect prioritizes performance and fast responsiveness over comprehensive security measures. The platform employs selective data encryption, focusing on sensitive information such as passwords rather than encrypting all user data. While comprehensive encryption enhances security, it can introduce performance overhead. The app balances heightened security measures and optimal system performance by prioritizing encryption for critical data, notably passwords. This trade-off acknowledges the importance of security while considering practical implementation factors.

2.3. Rapid Development vs. Enhanced Functionality

CampusConnect has adopted an advanced search system, specifically Elastic Search, to offer its users comprehensive search capabilities, moving beyond a basic search and filter approach. This strategic decision is a trade-off, emphasizing enhanced functionality at the expense of the speed and simplicity favored in rapid development. Adopting Elastic Search brings numerous advantages: it provides a powerful and feature-rich search experience, can handle complex queries, and offers scalability and high performance. This ensures that the platform can adapt and grow according to user needs. Additionally, it includes advanced features like real-time data indexing, significantly enhancing the overall user experience. However, this approach comes with certain drawbacks in terms of development. The integration and fine-tuning of Elastic Search could extend the development timeline, as it requires specialized expertise and thorough testing. This integration adds a layer of complexity to the development process, necessitating ongoing maintenance and regular updates to the search system. Furthermore, ensuring seamless integration of Elastic Search with other platform components, such as filtering systems, presents additional challenges, potentially impacting the development workflow and resource allocation.

3. Subsystem Decomposition



3.1 Communication Manager:

- Notification Management: Handles the generation and delivery of notifications to users, ensuring timely updates and alerts such as new messages within App.
- Email Management: Manages the email communication system, handling user interactions and “New items under the tag” type reminding emails.

3.2 Transaction Manager:

- Facilitates secure and efficient transaction processing within CampusConnect, ensuring smooth financial interactions such as fee payments, purchases, and other monetary transactions.

3.3 Profile Manager:

- Statistics Management: Gathers, processes, and presents statistical data related to user activities and system usage, providing insights into user behavior and system performance.
- Report Management: Manages the generation of product reports as well as user reports, and forwarding these reports to moderators.

3.4 Marketplace Manager:

- Post Management: Handles the creation, editing, and deletion of posts within the marketplace.
- Search/Filtering Management: Enables users to efficiently search and filter marketplace content by using tags and other criterions to enhance the overall usability and accessibility of the platform.

3.5 Forum Manager:

- Manages the functionality and content of discussion forums within CampusConnect, facilitating communication and collaboration among users on various topics and subjects.

3.6 Tag Manager:

- Organizes and manages the tagging system for content within CampusConnect, allowing users to categorize and search for information more effectively.

3.7 Authentication Manager:

- Handles user and moderator authentication and authorization processes, ensuring secure access to CampusConnect and safeguarding user data.

3.8 Moderator Manager:

- Manages moderator creation, and deletion process.

3.9 JpaRepositoryr:

- Acts as a Data Access Object (DAO), providing an abstraction layer above the database to interact with the data source for creating, reading, updating, and deleting records.

3.10 **ImageStorageManager**:

- Handles image data storage and retrieval, acting as an interface between the application and file storage systems.

3.11 **Seller Interface**

- Provides the UI for functionalities that a seller needs to manage their product listings, sales, and interactions with potential buyers within the marketplace.

3.12 **Buyer Interface**:

- Offers the UI for features required by buyers to browse products, make purchases, and interact with sellers on the marketplace platform.

3.13 **Renter Interface**

- Provides the UI components rental transactions by providing tools and functionalities for users to list, manage, and rent out items, as well as to negotiate terms and conditions.

3.14 **Borrower Interface**:

- Specifically designed for users looking to borrow items, this interface allows them to search for available items for borrowing, initiate borrowing requests, and manage their borrowing history.

3.15 **Moderator Interface**

- Offers the UI for moderators to manage tags, reports, and enforce community guidelines across the platform.

3.16 **Admin Interface**

- Provides administrators UI screens that allow admin to create moderator, view analytics and manage platform.