1. Introduction

1.1. Purpose of the System

The purpose of the CampusConnect system is to provide a web-based platform that connects members of the Bilkent University community for various purposes. The system aims to facilitate buying and selling second-hand items, posting lost and found items, making donations, and borrowing or renting items within the university community. It also incorporates features such as messaging functionalities to enable communication between users for item inquiries and negotiations. The system is designed to be user-centric, with an intuitive and easy-to-learn interface that enhances the user experience. It supports a diverse user base, including students, faculty members, and staff, and accommodates different user preferences and accessibility needs by allowing access through web browsers on computers and mobile devices. To ensure the security and reliability of the system, it integrates with reputable payment processors for secure transactions, implements data privacy and encryption measures, and has mechanisms to handle reported issues promptly. Additionally, the system undergoes regular backups and maintenance by a dedicated team to ensure continuous support and improvement. Comprehensive documentation is provided to support users, administrators, and developers, offering clear guidance on how to effectively use and manage the system. Overall, the purpose of the CampusConnect system is to foster a safe, efficient, and user-friendly environment for the Bilkent University community to connect and engage in various activities.

1.2. Design Goals

Maintainability and security are the main design goals chosen for the CampusConnect project due to their critical importance in ensuring the long-term success and reliability of the system. By focusing on maintainability and security, the CampusConnect project aims to create a system that can evolve, adapt, and withstand potential security threats while providing a seamless and secure user experience. These design goals ensure the long-term success of the platform, meeting the diverse needs of the university community and fostering a positive and secure online environment.

1.2.1. Maintainability

The CampusConnect website aims to prioritize maintainability while considering its trade-offs with performance. The system is designed to undergo modifications and enhancements as the platform evolves to meet the dynamic needs of the university community. The use of an object-oriented approach and modular structure facilitates future modifications, such as bug fixes and feature enhancements, without disrupting the user experience. Additionally, the system incorporates a data management strategy that ensures efficient database design to maintain system performance as the user base expands and the system accumulates information.

1.2.1.1. Maintainability vs Performance

In the development of CampusConnect, our core objective is to prioritize maintainability alongside low-latency responses and consistent performance. To directly address maintainability, we've implemented a modular architecture, comprehensive documentation, strict adherence to coding standards, and robust version control. These practices ensure ease of updates, modifications, and repairs over the system's lifecycle. While resilience and fault tolerance mechanisms are in place, we emphasize these directly support maintainability, contributing to the system's adaptability and long-term sustainability.

Altogether, the CampusConnect website strikes a balance between maintainability and performance. It prioritizes maintainability to facilitate future modifications and enhancements, while also ensuring a responsive and robust system that meets the diverse and dynamic needs of its university community users.

1.2.2. Security

The CampusConnect website prioritizes security and privacy to ensure the protection of user information and transactions. The system incorporates various measures to address security issues and maintain data privacy. User authentication and authorization are implemented to control access to data and the system, preventing unauthorized access. Data encryption is employed to ensure data integrity and protect sensitive information. Additionally, the system requires users to open their accounts with their Bilkent University email address and Bilkent ID to prevent multiple account operations and track poor behavior or foul language. The user agreement includes a clause allowing the monitoring of personal chats by administrators to reduce scams and ensure user security.

1.2.2.1. Security vs Usability

It is worth noting that there may be trade-offs between security and privacy measures and usability. Stricter security measures, such as complex authentication processes or frequent data encryption, may introduce additional steps or delays that could impact the user experience. Balancing security and privacy with usability requires careful consideration and user feedback to ensure that the system remains secure while providing a seamless and user-friendly experience.

While the system is designed to prioritize security, the CampusConnect website strives to achieve a balance between privacy, security, and usability as a whole. It guarantees a simple and effective user experience while placing a high priority on the security of user data and transactions. Consistent user input, updates, and monitoring are necessary to improve security measures over time.

2. High-Level Software Architecture

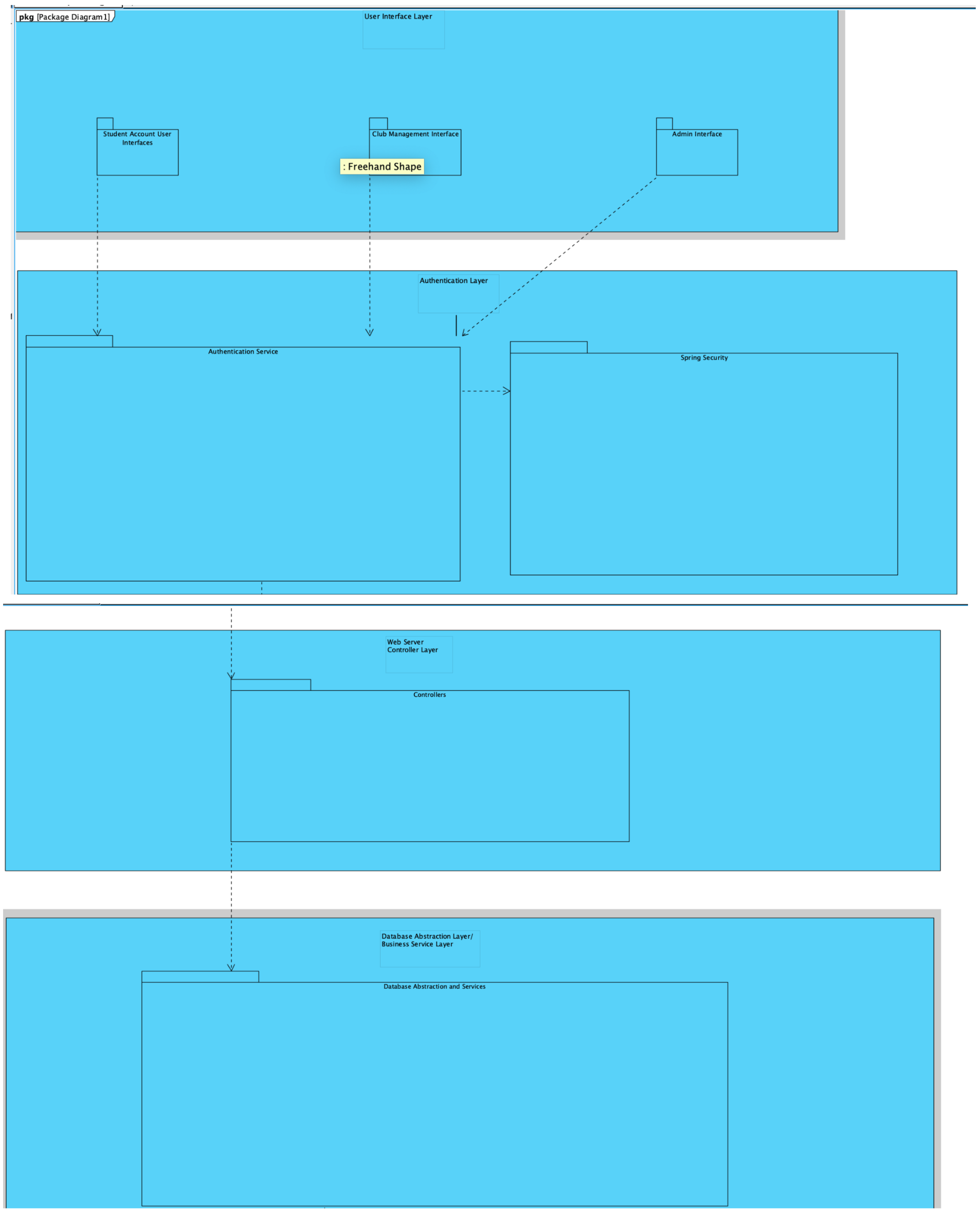
2.1. Subsystem Decomposition

CampusConnect architecture has 6-layers:

User Interface Layer: The part where the user interacts with the program and the interface is created.
Authentication Layer: Where authentication check is done to determine the user type and check if the user is valid.
Web Server Controller Layer: Since our application is done on Spring Boot, these layers were created accordingly. The Controller layer handles incoming HTTP requests, orchestrates user input, and returns the appropriate responses. It acts as the beginning point for interactions with the external.
Database Abstraction / Business Layer: The layer contains the business logic of the application. It processes data, executes business rules, and performs computations. This layer acts as a bridge between the Controller and Data layers, ensuring that business operations are correctly executed.
Database Layer: Often realized as repositories, interacts with the database. It handles data storage, retrieval, and updating. This layer abstracts the underlying data source, allowing the rest of the application to interact with the database without needing to know about specific database details.

2.2. Subsystem Decomposition Diagram

Below is the Subsystem Decomposition Diagram of CampusConnect and a higher resolution of this diagram can be found under the name Subsystem Decomp.vpp on our GitHub CampusConnect repository.


pkg [Package Diagram1]
User Interface Layer
Student Account User Interfaces
Club Management Interface
: Freehand Shape
Admin Interface
Authentication Layer
Authentication Service
Spring Security
Web Server Controller Layer
Controllers
Database Abstraction Layer/ Business Service Layer
Database Abstraction and Services

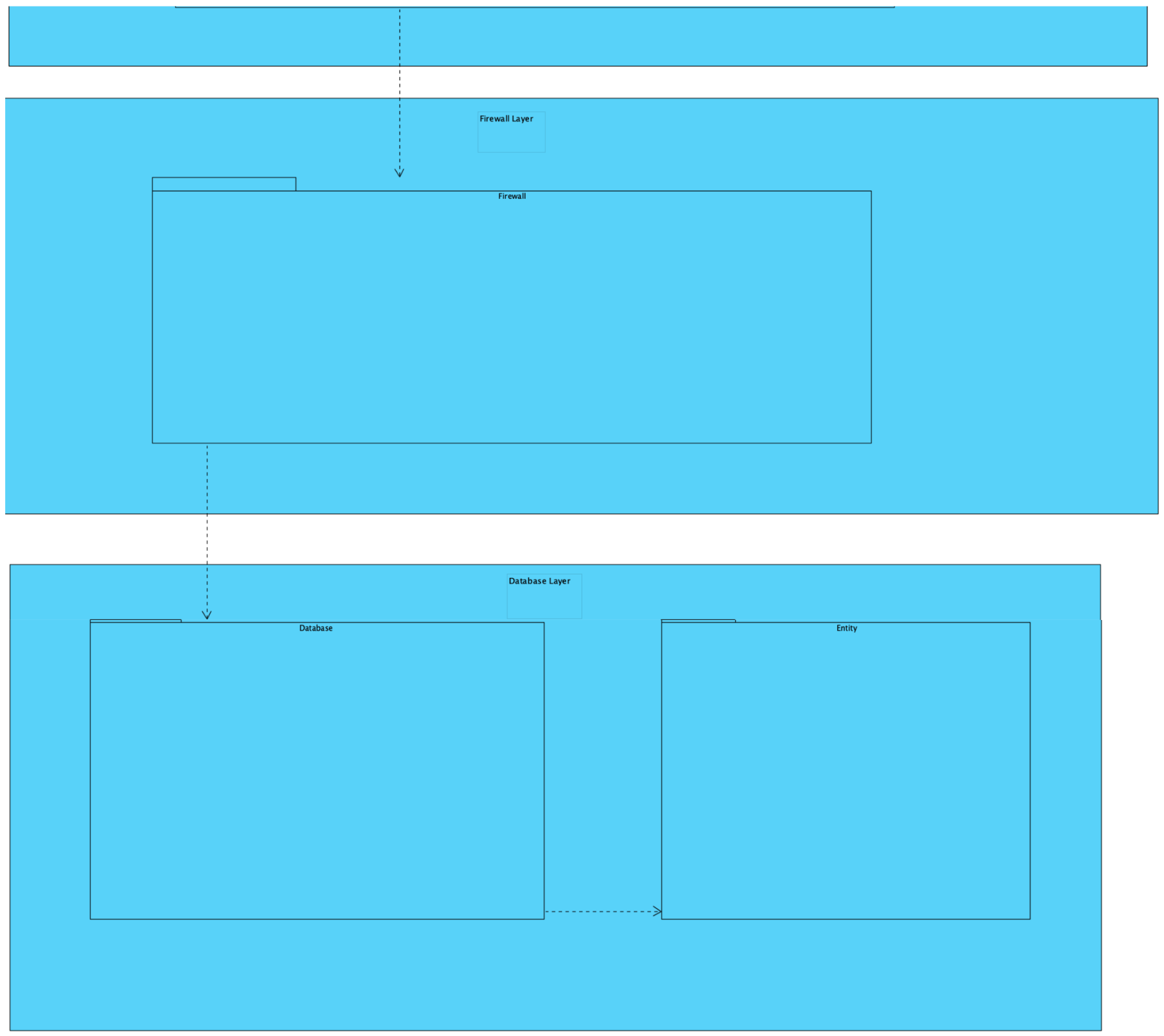


Figure 1. Subsystem Decomposition Diagram

2.3. Hardware/Software Mapping

On our frontend we will use React with JavaScript to manipulate HTML5 and CSS3 expressions. Additionally, we will use Bootstrap5 to create the look we aim for. This structure should be supported by browsers such as Google Chrome, Microsoft Edge, and Mozilla Firefox. Our frontend technologies are very common in modern websites so it should be expected to operate with minimal problems which we aim to fix if any occur.

On our backend, we will use Java Spring Boot and MySQL. Spring Boot will be used to create the server. Server communication will be provided with Java. Our data will be kept in the MySQL database. Our servers and database will be hosted on AWS servers. Any data that will be sent to the server side will be first with the React validation engine library to reduce the server usage. This will reduce the workload of the server. We will use the AWS free tier which should be enough in our product's initial trials since it is hard to estimate university members' interest in a secondhand sale site. However, we will use a queue system at the login to reduce the server load. And if the queues are longer than expected we can upgrade the server tier.

2.4. Persistent Data Management

CampusConnect will store many different types of data since it has many functionalities. Since then data management has been a very important issue. We will store the user information and their relatable items depending on the item they have posted such as SecondHand Sale, Lost&Found, or BorrowableItem. The items and users will be stored in different tables and will be linked with link tables. So we will use a relational database. Objects will be stored as JSON to increase the performance. Since the user information is private and a major security issue password will be stored after being hashed with the Bcrypt. We will store all of our data on MySQL database since it is reliable and one of the most commonly used databases which makes it less prone to conflicts and errors.

2.5. Access Control and Security

Security is an important issue for CampusConnect since it stores private information about the users. Permission of the users is limited for both the functionality and the security based on their roles. This is the first layer of access control and security. There are two major roles in our system which is CampusConnectUser will be referred to as users and administrators will be referred to as admins. Even though this site is for all of the university members since there were no permission differences between students and teaching staff their titles will not be referred to as different roles they will be both under the user class. Their titles will be visible through their profiles. Only an additional role to this Club Representative can be assigned to any existing user which will give permission to edit the club's home page, add new donation requests, and edit pre-existing donation requests. Other than that they will have similar permissions like an ordinary user. All users will be able to see all the pre-existing listing in the system but they can only edit or remove the listings or post they own. All the listings will be stored in a database and they will be sent by the server on request. Chats between users will also be stored in the database and will be only available to users who own it. On the other hand admin role is much more flexible they will be able to edit and remove any listing from any user. However, chat history between any user will also be unavailable for the admins since this violates privacy policies. Chat history will be accessible through the database only on special occasions such as suspicion of violence or such.

Users' private information like their addresses, emails, ID numbers, or such will not be displayed on any part of the system. Each user will be allowed to update their own address or their payment information. This data will not be accessible to any other user. Individual posts and listings will only contain the name of the author of the post. Donation Requests and club-related posts will not contain anything related to the author's identity. To reduce the attention on the club representatives in any case of misunderstanding. The activity log will be kept to see any user actions for security issues.

Outside participants to the system will be only allowed to see the login page which allows either logging to the system or registering with a Bilkent ID. This way university members will be protected against outside threats and the site can be considered as a private trade market.

Passwords, addresses, or payment information of the users will be hashed before sending to the server. For hashing, we will use Bcrypt. The database's firewall should also provide a reasonable amount of security.

	CampusConnect User	Admins	Seller/Author (owner of the listing)	Club Representative
Login	**X**	**X**	**X**	**X**
Renew Password	**X**	**X**	**X**	**X**
Edit Profile	**X**	**X**	**X**	**X**
See		**X**	**X**	**X**

SecondHand Sale Listings	**X**			
Add a new SecondHand Sale Listing	**X**	**X**	**X**	**X**
Edit a SecondHand Sale Listing		**X**	**X**	
Delete a new SecondHand Sale Listing		**X**	**X**	
Edit any SecondHand Sale Listing		**X**		
Delete any SecondHand Sale Listing		**X**		
See Lost&Found Post	**X**	**X**	**X**	**X**
Add a new Lost&Found Post	**X**	**X**	**X**	**X**
Edit a Lost&Found Post		**X**	**X**	
Delete a Lost&Found Post		**X**	**X**	
Resolve a Lost&Found Post			**X**	
Edit any Lost&Found Post		**X**		
Delete any		**X**		

Lost&Found Post				
See BorrowableItem listing	**X**	**X**	**X**	**X**
Add a new BorrowableItem Listing	**X**	**X**	**X**	**X**
Edit a BorrowableItem Listing		**X**	**X**	
Delete a BorrowableItem Listing		**X**	**X**	
Edit any BorrowableItem Listing		**X**		
Delete any BorrowableItem Listing		**X**		
See Club's Home Page	**X**	**X**	**X**	**X**
Edit Club's Home Page		**X**		**X**
Add a Donation Request		**X**		**X**
Edit a Donation Request		**X**		**X**
Delete a Donation Request		**X**		
Edit any Donation Request		**X**		
Delete any		**X**		

Donation Request				
View Profile	**X**	**X**	**X**	**X**
Send Message	**X**	**X**	**X**	**X**

Figure 2. Access Control and Security Matrix

2.6. Boundary Conditions

CampusConnect's boundary conditions encompass the initialization process, ensuring secure user access and server setup, termination procedures allowing independent subsystem shutdowns, and handling various failure scenarios such as AWS issues or internet connection loss, all designed to maintain data integrity and user experience.

2.6.1. Initialization

During the initialization process of CampusConnect, users can access the web application by logging in with their existing accounts. Once logged in, the system fetches relevant data from the database to initialize the application. Users without accounts can still access the application but with limited privileges, such as only being able to view university information pages. Upon logging in, the sidebar displays related page options, and when a user clicks on a page, the corresponding information, such as course lists, student lists, or pre-approval forms, is fetched from the database. Additionally, to set up the server, the maintainer needs to initialize it on AWS and ensure that AWS S3 credentials are provided and properly configured to handle user files. The database also needs to be initialized with the correct credentials passed to CampusConnect. If required, the server's IP address should be provided to the database for secure communication. Furthermore, when a user signs in, an access and refresh token is generated internally, which is later used for information integrity checks.

2.6.2. Termination

In the CampusConnect system, single subsystems are allowed to terminate. This means that individual components or modules within the system can be terminated independently without affecting the overall functionality of the system. This allows for flexibility and modularity in the system's design, as subsystems can be modified, updated, or replaced without disrupting the entire system. When a single subsystem terminates, there is no explicit notification sent to other subsystems. However, the system is designed to handle such terminations gracefully. The termination of a subsystem is typically handled within the system's code, ensuring that any necessary cleanup or data-saving processes are executed before the subsystem shuts down. This ensures data integrity and prevents any adverse effects on other subsystems or the overall system. Updates to the database in CampusConnect are communicated through a structured and standardized manner. The system utilizes appropriate protocols and mechanisms to interact with the database, such as SQL queries or ORM (Object-Relational Mapping) frameworks. When updates are required, the system sends the necessary commands or queries to the database, which then processes and applies the changes accordingly. This ensures that the database remains synchronized with the system's data and reflects any modifications made within the system. Additionally, the system may implement data validation techniques to ensure the accuracy and integrity of the data being communicated to the database.

CampusConnect can be terminated in several ways. Firstly, when a user logs out of the application, the application terminates and the user's data is saved in the database. Secondly, an admin user has the ability to terminate the program for maintenance or security purposes. In this case, the last submitted version of the users' data is saved and the application terminates. Thirdly, in the event of a failure due to an unhandled exception, the system may terminate. However, the backend of CampusConnect is designed to handle these situations gracefully, either by exiting the system smoothly or displaying helpful error messages to the user. Additionally, if a user's access token expires, a request is sent to the authentication service. If the refresh token has not expired, the authentication service refreshes the access token. However, if the access token has expired and the refresh token has also expired, the user will be automatically signed off from the system.

2.6.3. Failure

CampusConnect will be hosted on AWS EC2 servers, and there are no additional web servers available to run the application. Therefore, if any issues arise within AWS, it may result in the failure of the application. In such cases, the program will be restarted using the last saved data stored in the database. Additionally, as CampusConnect is a web application, if a user experiences an internet connection loss, it may also lead to a failure in accessing the application. In this scenario, the user will need to log in again, and the application will start with the last saved data for the user retrieved from the database. This ensures that users can resume their activities seamlessly even in the event of technical disruptions.