1. Design Patterns

1.1. Strategy Pattern

The infusion of the strategy pattern into the CampusConnect system signifies a pivotal design enhancement crafted to augment its flexibility and extensibility. This strategic integration empowers CampusConnect to dynamically modulate its behavior during runtime by encapsulating distinct algorithms or strategies within interchangeable components. Such a strategic approach facilitates the seamless adaptation of the system to diverse user authentication needs and evolving security requirements, all achieved without necessitating modifications to the core codebase.

To illustrate this concept further, consider the authentication process within CampusConnect. The strategy pattern becomes instrumental in managing various authentication methods, such as username/password and two-factor authentication. Each authentication method becomes encapsulated within a dedicated strategy class, offering the system the capability to dynamically switch between these strategies based on user preferences or evolving security standards. This nuanced approach ensures that CampusConnect remains adept at responding to changing authentication landscapes while adhering to the highest security standards.

The adoption of the strategy pattern for authentication yields a plethora of advantages, including heightened code reusability, maintainability, and scalability. Novel authentication methods seamlessly integrate into the system, and adjustments to existing methods can be executed without imposing disruptive alterations on the overall system architecture. In essence, the strategy pattern serves as a linchpin for future-proofing the CampusConnect system, ensuring it remains agile and adaptive in the face of evolving technological and security requirements.

1.2. Façade Pattern

In the intricate fabric of a full-stack web application like CampusConnect, the Façade pattern emerges as a key orchestrator, seamlessly harmonizing interactions between the front end and the sophisticated backend subsystems, including Java Spring Boot. The strategic integration of the Façade pattern introduces a refined simplicity, presenting a unified interface (facade) to the front end while expertly abstracting away the intricacies of underlying subsystems. This architectural finesse not only expedites development but also elevates the overall user experience. CampusConnect employs the Façade pattern in two strategic realms — firstly, at the web-server layer, where distinct structures known as services encapsulate the implementation of requests at each endpoint. This approach ensures that controllers experience minimal disruption even when there are changes in the implementation. Furthermore, the Façade pattern extends its influence to the database abstraction layer, utilizing repositories to abstract database communication. Repositories adeptly manage interactions with the database, isolating the logic from services. Consequently, changes in the database layer, encapsulated within repositories, exert no influence on the services layer. This bifocal implementation of the Façade pattern enhances not only code reusability, maintainability, and scalability but also fortifies the robust architecture of CampusConnect across its web-server and database realms.

1.3. Observer Pattern

The Observer pattern can be effectively utilized in the CampusConnect system to facilitate communication and synchronization between different components. Specifically, it can be employed to notify the frontend, developed using React, about any changes or updates occurring in the backend, implemented with Java Spring Boot. By implementing the Observer pattern, the system can ensure that relevant frontend components are promptly informed about new data or updates in the backend, enabling them to update their views accordingly. This pattern enhances the overall user experience by ensuring that the front end remains synchronized with the backend, providing real-time information and seamless interactions for users.

2. Class Interfaces

User Interface

<u>Login Page</u>
public login(String id, String pwd) On click sends a login request.
public forgotPassword() On click direct you the forgot password page.
public directReg() On click directs register page

<u>Register Page</u>
public register(String id, email email, String name, String pwd) On click validates bilkent id and email. If id and mail validates user is registered to the system.

<u>Request New Password Page</u>
public requestNewPassword(String id) On click sends an email to the user with the given id it the user exist in the system.

<u>Resest Password Page</u>
public resetPassword(String newPwd) On click new password is set to the user.

<u>Top Nav Bar</u>
public logout() On click session ends and user logs out front the system.
public viewMyProfile() On click user views his/her own profile.
public seeDirectMessages() On click user can see the messages he/she received.
public secondhandsales() On click directed to secondhand sale page.
public borrowing() On click directed to borrowing page.
public donations() On click directed to donations page.
public lostandfound() On click directed to lost and found page.
public viewNotfications(): On click shows all the notifications.

<u>Your Profile Page</u>
public editProfile() On clicks directs to edit profile page.
public changePassword(): On click existing and a new password is requested.
public pwdChange(String exPwd, String newPwd) On click exPwd verified. If ex password is true new password is set.

<u>Edit Profile Page</u>
public uploadPhoto(JPEG image): On click uploads the select JPEG file.
public update(JPEG image, String bio) On click updates the user information.

Others Porfile Page

public sendMessage(): On click directed to direct messages page with an open chat with the owner of the profile.
public reportProfile(): On click a drop down menu of reason for reporting is displayed with an empty box for additional information.
public report(String type, String addInfo) On click report is sent with the information filled.

Donation Page

public searchDonation(String name) On click search the database for the donation with the given name.
public viewDonation(): On click directed to donation request's main page with additional information.
public deleteDonation(String id) On click deletes the donation.
public addDonation() On click directs to add donation page.
public viewClubProfile() On click directed to club's profile.
public updateDonationRequest() On click directed to update donation page.
public List<DonationModel> showAllDonations() On load displays all the donation requests according to their recency.

Club Profile Page

public editClubProfile() On click user is directed to edit club profile page.

Second Hand Sale Page

public searchItem(String name) On click search the database for the item with the given name.
public viewSale() On click directed to sale's main page with additional information.
public deleteSale(): On click delete the sale.
public addNewSale() On click direct to add new sale page.
public viewSellerProfile() On click directed to seller's profile
public updateSale(): On click directe to update sale page.
public sendMessage(): On click directed to direct messages page with an open chat with the seller.
public List<SHSItem> showAllSales() On load displays all second hand sales according to their recency.

Second Hand Sale Item Page

public viewSellersProfile() On click directed to author's page.

public sendMessage() On click directed to direct messages page with an open chat with the seller.

<u>Borrowing Page</u>

public searchItem(String name) On click search the database for the item with the given name.
public viewSale() On click directed to sale's main page with additional information.
public deleteSale(): On click delete the sale.
public addNewSale() On click direct to add new sale page.
public viewSellerProfile() On click directed to seller's profile
public updateSale(): On click directe to update sale page.
public sendMessage(): On click directed to direct messages page with an open chat with the seller.
public List<BItem> showAllSales() On load displays all the item borrowing sales according to their recency.

<u>Borrowing Item Page</u>

public viewSellersProfile() On click directed to author's page.
public sendMessage() On click directed to direct messages page with an open chat with the seller.
public requestExtension(int days) On click extension is requested from the seller with selected no of days.
public assignCustomer(String id) On click an assignment request to the user is sent. The borrower should also accept it from the notifications section.

<u>Lost and Found Page</u>

public searchItem(String name) On click search the database for the item with the given name.
public viewPost() On click directed to posts main page with additional information.
public deletePost() On click delete the post.
public addNewPost() On click direct add new lost or and found post page.
public viewAuthorsProfile() On click directed to author's page.
public updatePost() On click directed to update post page
public resolve() On click mark post as resolved and id of the resolver is saved to the database.
public sendMessage() On click directed to direct messages page with an open chat with the author.

public List<LFItem> showAllPosts() On load displays all the lost and fount item posts according to their recency.

<u>**Lost and Found Item Page**</u>
public viewAuthorsProfile() On click directed to author's page.
public sendMessage() On click directed to direct messages page with an open chat with the author.
public resolvePost() On click textBox is opened for the id of the resolver to saved.
public resolve(String id) On click id of the revolver saved to the system.

Entity Classes

<u>**CampusConnectUser**</u>
An entity class that represents user with minimum features.
Attributes

- **private long id:** primary unique key of the user
- **private String name:** user's real name
- **private String email:** user's bilkent email
- **private long bilkentID:** user's bilkent id
- **private String password:** user's hased password
- **private String degree:** user's degree in school; undergraduate, teaching staff, masters
- **private Date birthdate:** user's birthday
- **private JPEG image:** user's profile picture
- **private boolean changePassword(String newPwd, String oldPwd):** validates the old password and if valid sets a new password for the user
- **private boolean resetPassword(String newPwd):** In case of a forgotten password. Password can be reset via email.
- **private boolean uploadImage():** set or replaces the profile picture

<u>**ClubRepresentitive**</u>
An entity class that extends CampusConnectUser. Has additional permission to update club pages and donation request. Contains all other attributes of CampusConnectUser.

Attributes

- **private int clubID:** foreign key for the club

- **private boolean resign()** resign the role of club representetive

Item
An abstract entity class that contains common attributes of all type of items.
Attributes

- **private long id:** primary key for the item object
- **private String name:** name of the item
- **private String category:** category item belongs
- **private JPEG image:** picture of the item
- **private String additionalInfo:** additional information about the item
- **private long userID: i**d of the user that added the item to the system
- **private timeStamp publishDate:** time stamp of the entry of item
- **private boolean uploadPicture(JPEG image)** upload a picture of the item
- **private void editItem(String ..):** edit any attribute of the item other than id.

SecondHandItem
This entity class extends from item class and contains all attributes of it.

Attributes

- **private double price:** price of the item
- **private boolean isNegotiable:** represents if the item is negotiable
- **private void editItem():** overrides the editItem method of parent class to enable the edit of the additional attributes.

BorrowItem
This entity class extends from item class and contains all attributes of it.

Attributes

- **private int minDays:** minimum number of barrowable days
- **private int maxDays:** maximum number of borrowable days
- **private boolean isFree:** represents if the item has a rent fee
- **private double price:** represents the rent fee if it exists otherwşse set to 0
- **private long renterID: i**ncase item is borrowed, borrowers id is saved
- **private int remaining days:** number of days for item to returned to the owner
- private Date endDate: end date of the renting
- **private Date begDate:** beginning date of the renting

- **private void requestExtension(int noOfDays):** if the item is rented, renter can request the return date to postponed
- **private void emergentRequest():** for owner to request the return of the item
- **private boolean evaluateExtension():** in case of an extension the extension is evaluated by the owner
- **private void reportRenter():** if the item is not returned to the owner in item owner can report the renter.
- **private void editItem():** overrides the editItem method of parent class to enable the edit of the additional attributes.

LostandFoundItem

This entity class extends from item class and contains all attributes of it.

Attributes

- **private String type:** type of the item Lost/Found
- **private long resolverID:** if the case is resolved id of the resolver otherwise empty
- **private String location:** the location of the item that is found or lost
- **private void editItem():** overrides the editItem method of parent class to enable the edit of the additional attributes.
- **private void resolve(String resolverId):** marks the post as resolved and saves the resolver's id

Club

An entity class that represents clubs.

Attributes

- **private int id:** primary key of the club
- **private String name:** name of the club
- **private List<DonationRequest> donationRequests:** list of donation requests related to the club
- **private int nofClubRep:** no of club representetşve assigned for the club
- **private String bio:** biography of the club
- **private JPEG image:** amblem or picture for the club
- **public boolean assignRep(String id)**: bilkent id of the user is given to give the role of the club representative

- **private boolean demoteRep(String id)** demotes the club representative with the given bilkent id to campus connect user. Nothing happens and error message is displayed if the user with the given id does not exist.
- **private void editClub()** edits any club information other than primary key id.

DonationRequest
An entity class that represents donation requests.

Attributes
- **private long id:** primary key of the donation request
- **private String name:** name of the donation request
- **private String donationType:** type of expected from the users
- **private String Location:** location of the donations going to be collected
- **private timeStamp publishDate:** publication date of the donation request
- **private Date endDate:** date of last day of the donation
- **private Date begDate**: date of first day of locations
- **private String addInfo:** additionaol information about donation request
- private JPEG image: image for the donation request
- **private long clubRepresentitiveID:** id of the club representitve that posted the donation request
- **private void edit() :** edit any attribute other than primary keyand publish date

Notification:
This entity class represents notifications such as message notification, donation request , announcements or borrow item request .

Attributes
- **private long id: primary key of the notification**
- **private String message: message of the notification**
- **private String type: type of notifications; direct message/announcement/report**
- **private timeStamp time: time stamp of notification**

Message
An entity class represents single direct message.

Attributes
- **private long id:** primary key of the message
- **private String message:** content of the message

- **private timeStamp time:** time of sent
- **private long senderId:** id of the sender
- **private receiverID:** id of the receiver
- **private boolean isDeleted:** if the message is deleted from the system
- **private void deleteMessage:** marks the message as deleted

DirectMessgaes

An entity class represents the chat between two users.

Attributes

- **private long id:** primary key of the direct message
- **private List<Message> messages:** all messages sent between the users
- **private long contact1ID:** id of the first user
- **private long contact2ID:** id of the second user
- **private void reportMessage()** report a message if violates user terms

Report

An entity class represents user reports about other users.

Attributes

- **private long id:** primary key of the report
- **private String type:** type of violation
- **private String addInfo:** reason for reporting

Controller Classes

LoginController

Attributes

- **private boolean isLoggedIn:** true if user is logged in. used for if the user is not logged in it is automatically directed to the page.

Methods

- **private ActionResult login(String id, String pwd, boolean rememberMe):** requests a login action if the password and id matches user is logged in to the system otherwise an error message is shown in the system.
- **private ActionResult forgotPwd();** Directs to forgot password page
- **private ActionResult register();** directs to register page
- **private void loadScene()** loads the login screen

<u>ProfileController</u>

Attributes

- **private long userID :** id of the logged in user.

Methods

- **private ActionResult editProfile():** if the user views his own profile the method is available and directs to edit profile page.
- **private ActionResult reportProfile():** user can report the other user profile
- **private ActionResult sendMessage():** opens a direct message between two users
- **private ActionResult changePassword():** directs of change password page
- **private void loadScene():** loads the profile of the user. If another profile is viewed editprofile and changePassword method are not available. If the user view its own profile reportProfile and sendMessage methods are unavailable.

<u>EditProfileController</u>

Methods

- **private ActionResult editProfile():** updates the profile according to given information
- **private JPEG upload():** upload an temporary image on client side
- **private void loadSceen():** loads the edit profile page with existing user information

<u>AccountManager</u>

Methods

- **private boolean changePassword(String newPassword, String oldPassword):** sends a password change request
- **private void setImge(JPEG image):** sends the new image
- **private String changeBio(String bio):** send the new biography information

DierctMessageController

Atributes

- **private DirectMessage:** direct message object that contains all the messages between two users.

Methods

- **private ActionResult sendMessage(String message):** new message is created and sent
- **private List<Message> loadAllMessages(DirectMessage dm):** loads all the messages between two users

<u>SecondHandSaleController</u>

Atributes

- **List<SecondHandItem> sales**

Methods

- **private void loadAllSales():** list all existing sales at the database according to their recency
- **private ActionResult viewSale():** directs the items page to seee it more detailed.
- **private ActionResul viewProfile():** directs to the profile of the owner of the post
- **private ActionResult filter():** fileters the sales according to given criteria
- **private ActionResult deleteSale():** if the user is the owner of the sale this option is available and user can delete the post
- **private ActionResult addSale():** directs to add sale page.
- **private ActionResult editSale():** owner of the sale can update the sales information by being directed to edit sale page.