CS 319 TERM PROJECT

Section 3
Group 3B
UNICLAPP

Design Report

Project Group Members:

1. Ayberk Yaşa	21801847
2. Fatih Kaplama	21802755
3. Görkem Ayten	21802399
4. Mert Atakan Onrat	21802520

Supervisor: Eray Tüzün

1. Introduction

1.1 Purpose of the System

This project is a web-based student club application. This application basically allows the clubs and students at Bilkent University to communicate more interactively. In this application, clubs can open their own accounts and make event announcements from here. Students, on the other hand, can follow these clubs using their own accounts and be informed about these announcements easily. In addition, the application offers students the opportunity to evaluate the clubs and the events organized by the clubs. In this way, each event and club will have a point and the clubs will be ranked.

1.2 Design Goals

1.2.1 Performance

A common problem with websites is that they run slowly when instant traffic increases. It is undesirable for this project to run slowly in such a situation. Therefore, all factors that can slow down the response time of the system are avoided. The project is deployed on the fastest and most scalable server. External libraries, which may have a negative impact on the working speed of the project, are used as little as possible. In other words, up to 15000 users, which is the predictable number of simultaneous users, it is designed to keep the loading time of the pages under 1 second and the response time of the buttons around 0.1 seconds.

1.2.2 Usability

Since student clubs usually announce their events via email, it is difficult for students to follow these events. In addition, students have to check emails one by one to find information about an event. This situation makes their job significantly more difficult and is undesirable. The application interface, buttons and texts are chosen quite simply for students to do these operations more easily. Students can easily find out which club has which event in just a few steps. In this way, even students who are introduced to the application for the first time can easily adapt to this environment without difficulty. In addition, there is a tutorial on the application interface that explains what features the application has and how these features can be used.

1.2.3 Reliability

The program is designed and expected to work and behave in a way that the number of times the errors or bugs occur in low percentage while users are using the application. The unwanted system problems are minimized to ensure that the users can use the application fluently. Users must access their upcoming and past events information %99 of the time without failure and users who close the browser without

properly logging out of their account will be automatically logged out. By this way the reliability of the program is increased and users are expected not to stop using the application immediately after encountering constant errors.

1.2.4 Scalability

Since this project is a student and student club interaction application, many students and club officials may log into the website during the day. That's why scalability is very important for this project. The server will be reserved in such a way that the bandwidth limit is not exceeded until the daily total, not simultaneous, of 50000 visitors.

2. High-level Software Architecture

2.1 Subsystem Decomposition

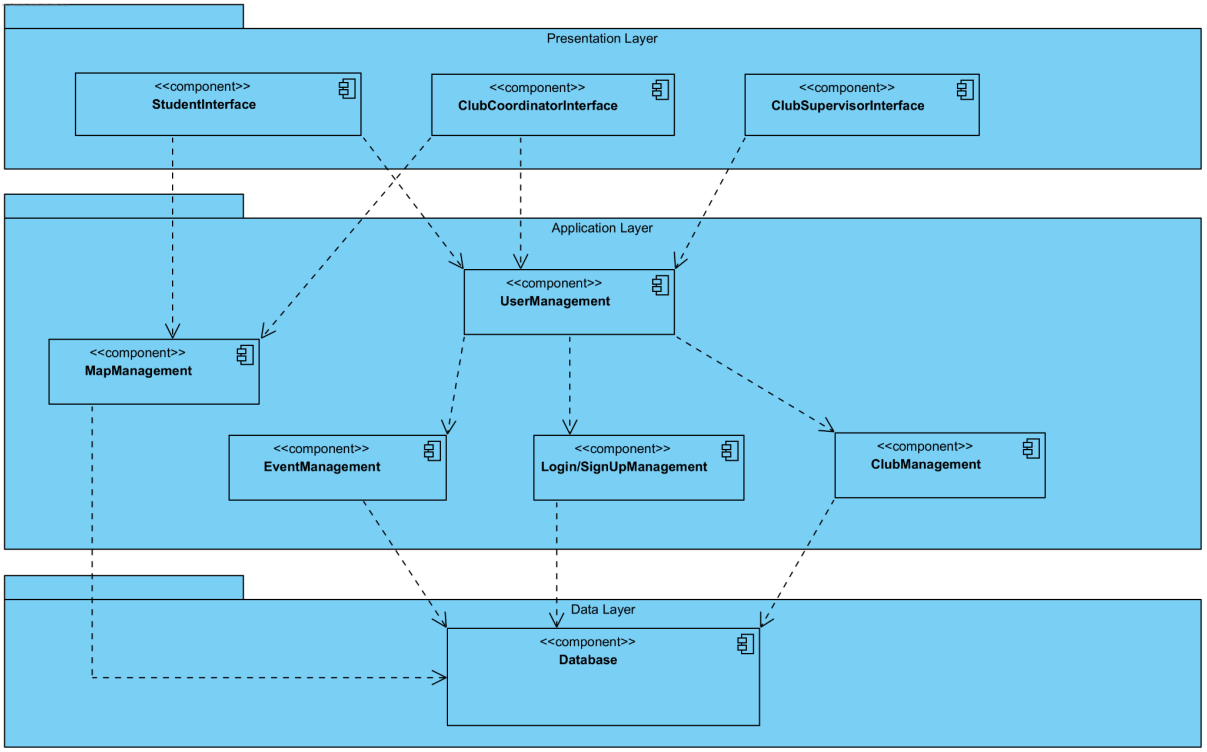

Figure 1. Subsystem Decomposition of the System

In this project, the three layer architecture consisting of presentation, application, and data layer is decided to be used. The Data Layer consists of the Database subsystem containing objects that need to be stored in the project. This subsystem is called by four of the subsystems in the Application Layer, which are MapManagement, EventManagement, Login/SignUpManagement, and ClubManagement.

The Application Layer contains five different subsystems which are UserManagement, MapManagement, EventManagement,

Login/SignUpManagement, and ClubManagement. All these subsystems in this layer are responsible for the operations that process the data obtained from the Database subsystem or each of the subsystems in the Presentation Layer. Moreover, each subsystem is constrained to certain functionalities. Thus, thanks to all subsystems, a structure containing all the functionality on the back-end is created. Lastly, the subsystems in this layer call the Database subsystem.

The Presentation Layer consists of three different weakly coupled subsystems containing the components of the user interface. The reason why it is divided into three different subsystems is that there are three different user types in the project and each of these user types accesses a different interface. Therefore, each subsystem contains all the pages that the user who is specific to that subsystem can access, and the functionality that those pages offer to the user. Lastly, the subsystems in this layer call the subsystems in the Application Layer.

2.2 Hardware/Software Mapping

This student club manager application does not need any hardware components to run successfully on the web browsers. However, since this project is a web-based application, a web browser is required on users' hardware systems such as computers and phones. This web browser can be Google Chrome, Microsoft Edge, Mozilla Firefox, Opera, or Safari [1-5]. For Google Chrome, the browser version should be 96.0.4664.45 or later. For Microsoft Edge, the browser version should be 96.0.1054.34 or later. For Mozilla Firefox, the browser version should be 94.0.2 or later. For Opera, the browser version should be 81.0.4196.54 or later. For Safari, the browser version should be 15.0 or later. Moreover, the version of Django to be used for the back-end is 3.2 and Vue.JS to be used for the front-end has the version of 2.x [6-7].

2.3 Persistent Data Management

In this project, a specific database is needed since clients send a request to the server to access the specific information such as student clubs, events or profile. These data should be accessed quickly. In this project, PostgreSQL [8] will be used. The reason PostgreSQL is chosen, it is an object relational database that is convenient to apply OOP principles. Moreover, PostgreSQL has various functionalities such as storing arrays. Thanks to PostgreSQL, the project's database will be more functional, speed and safe. Project's main objects such as Student, ClubCoordinator, ClubSupervisor, Club, Event, etc. will be stored in this database that is deployed in a cloud server by using AWS [9]. Clients can make some operations such as sending GET, PUT, POST and DELETE requests to the server. It is important to provide such operations to client.

2.4 Access Control and Security

Our web application has access control and it provides security. It is provided access control by giving permission to the logged user with the matching used type. In this way, a user cannot use the properties of a user type that they do not belong to. Also, to improve the security of the web application, there will be a good password policy. Users must create a password of at least 8 characters including lowercase, uppercase, special characters and numbers. These improve the security and the access control of the application.

2.5 Boundary Conditions

2.5.1 Initialization

Since this is a web-based application, no installation is required. All that is required for the application to work is to be connected to the internet. Therefore, the operation of the application is quite simple. This application is an account based application, so users must have an account to use this application. When a user enters the application page for the first time, registration and login screens appear. In order for the user to log in to the system, he or she must first register with the application and this user must be registered in the database. Then the user can use the application by logging in. If the user closes the application, the user's session is automatically closed, so the user has to log in again each time.

2.5.2 Termination

In this application, when the termination event occurs, the first of the events that will occur is to log the user out of the system. Therefore, when the user starts the termination event, he must log in to the system again in order to use the application. However, events such as registering for an event, creating an event, following a club, etc., performed by the user within the application, will not be lost when the termination event occurs, because these data will be saved in the database and the database will be in constant interaction with the application.

2.5.1 Failure

Amazon web service (AWS) will be used so that this application can work remotely and the database can be used and updated continuously. If there is a server-related problem in the application, this issue will be forwarded to the authorized persons. In case of bugs or errors that may occur in the application, the data will never be lost because it is saved in the database.

3. Low-level Design

3.1 Object Design Trade-offs

Maintainability versus Performance: This web application has many layers for almost every part of the project and this makes some disadvantages on performance. However, comparing advantages of maintainability with disadvantages of the performance of the application, the benefits of maintainability are much more.

Security versus Usability: This web application makes changing password in each three months obligatory. Each user has to create a new password different from the last three passwords in three months. Although this feature makes this application more secure, it decreases the usability.

Functionality versus Usability: This web application is designed especially for the students and student clubs of Bilkent University. This web application offers different functionality for each user such as rating for students and leaderboard for student clubs. Therefore, while the number of users grows, the usability of the web application would become complicated for new users.

3.2 Final Object Design

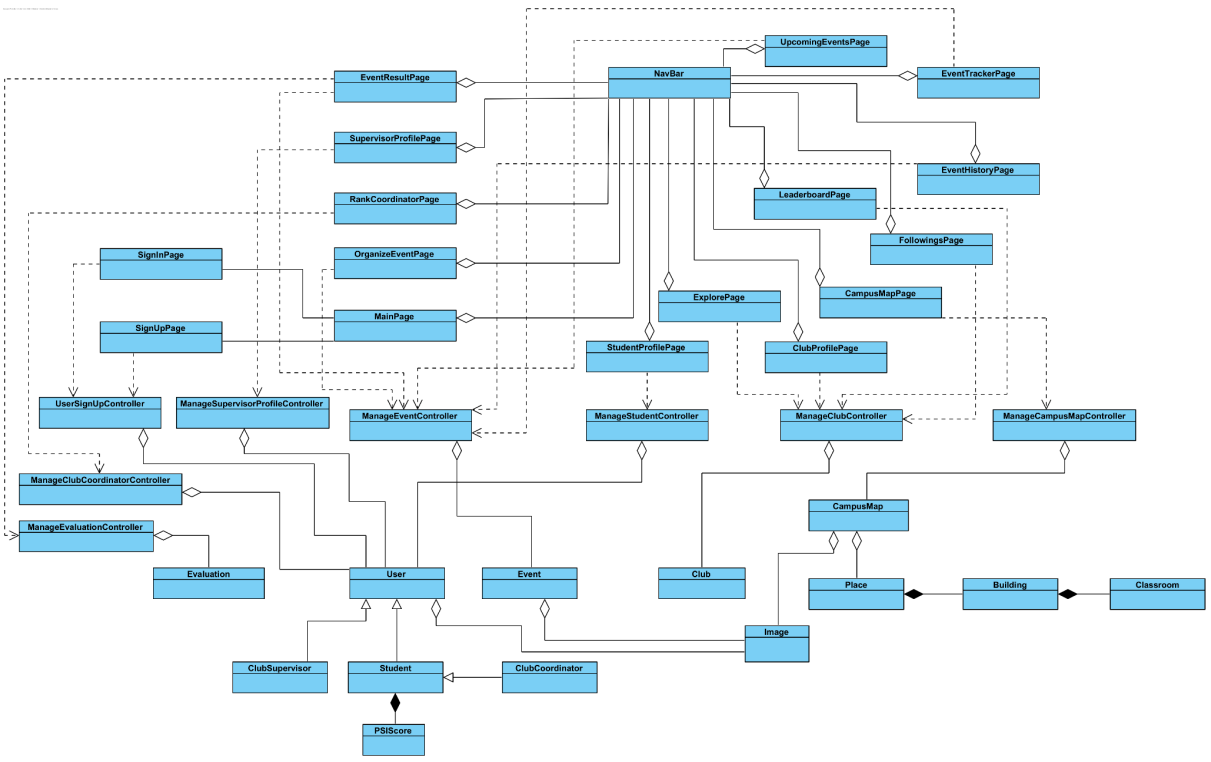

Figure 2. Abstraction of Final Object Design

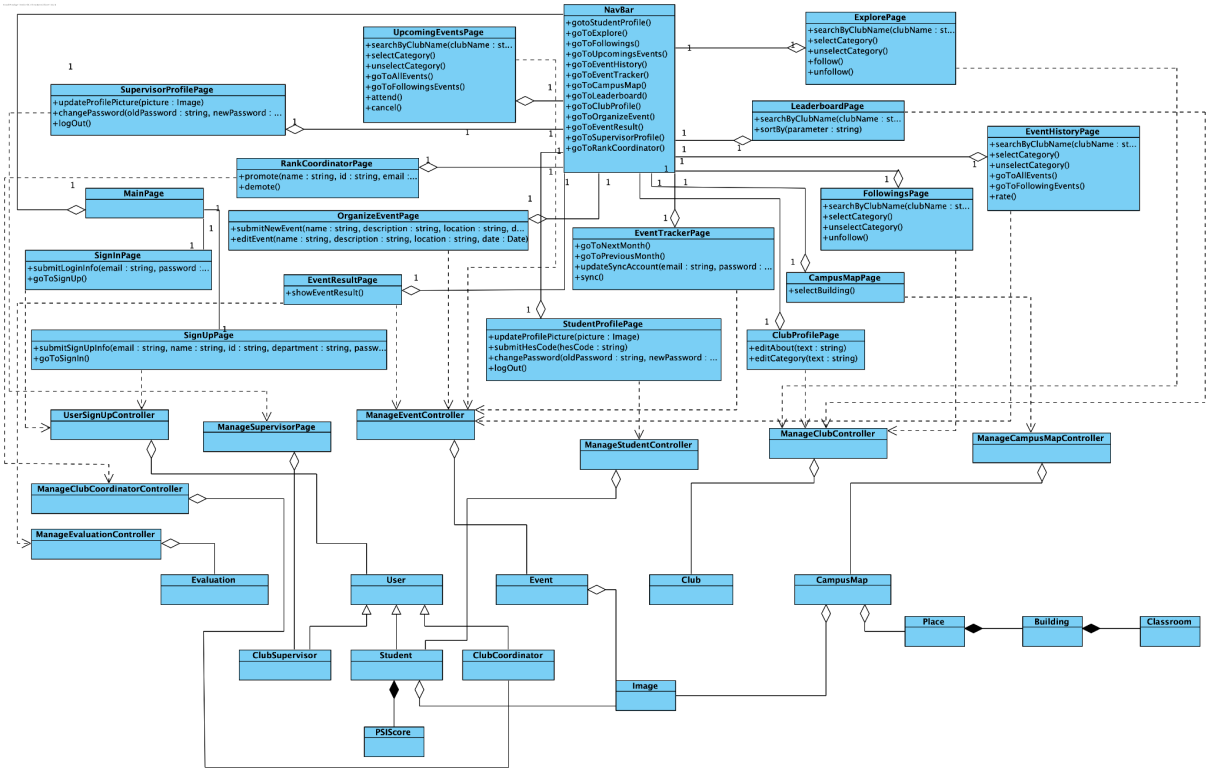

Figure 3. Classes of Boundary Objects

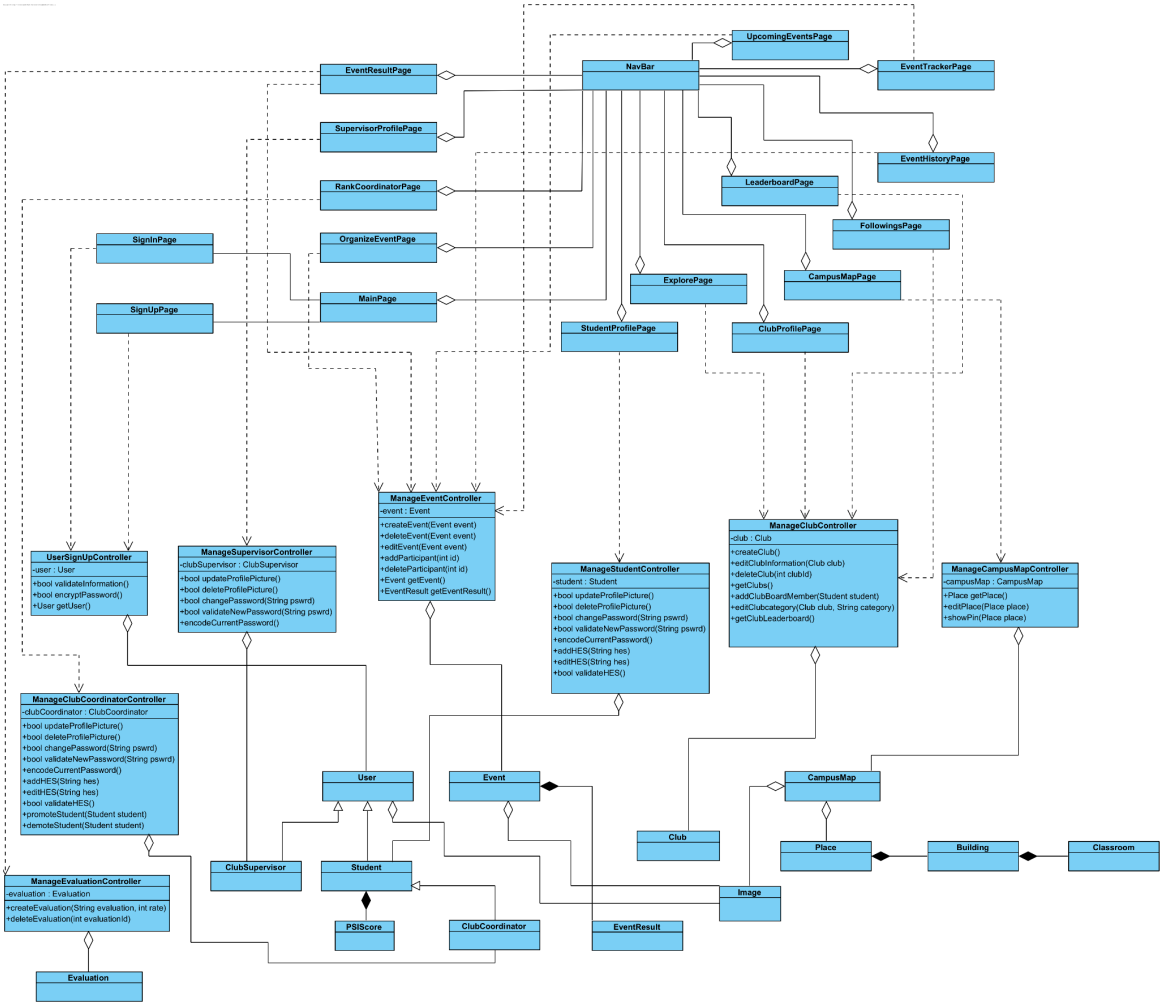

Figure 4. Classes of Controller Objects

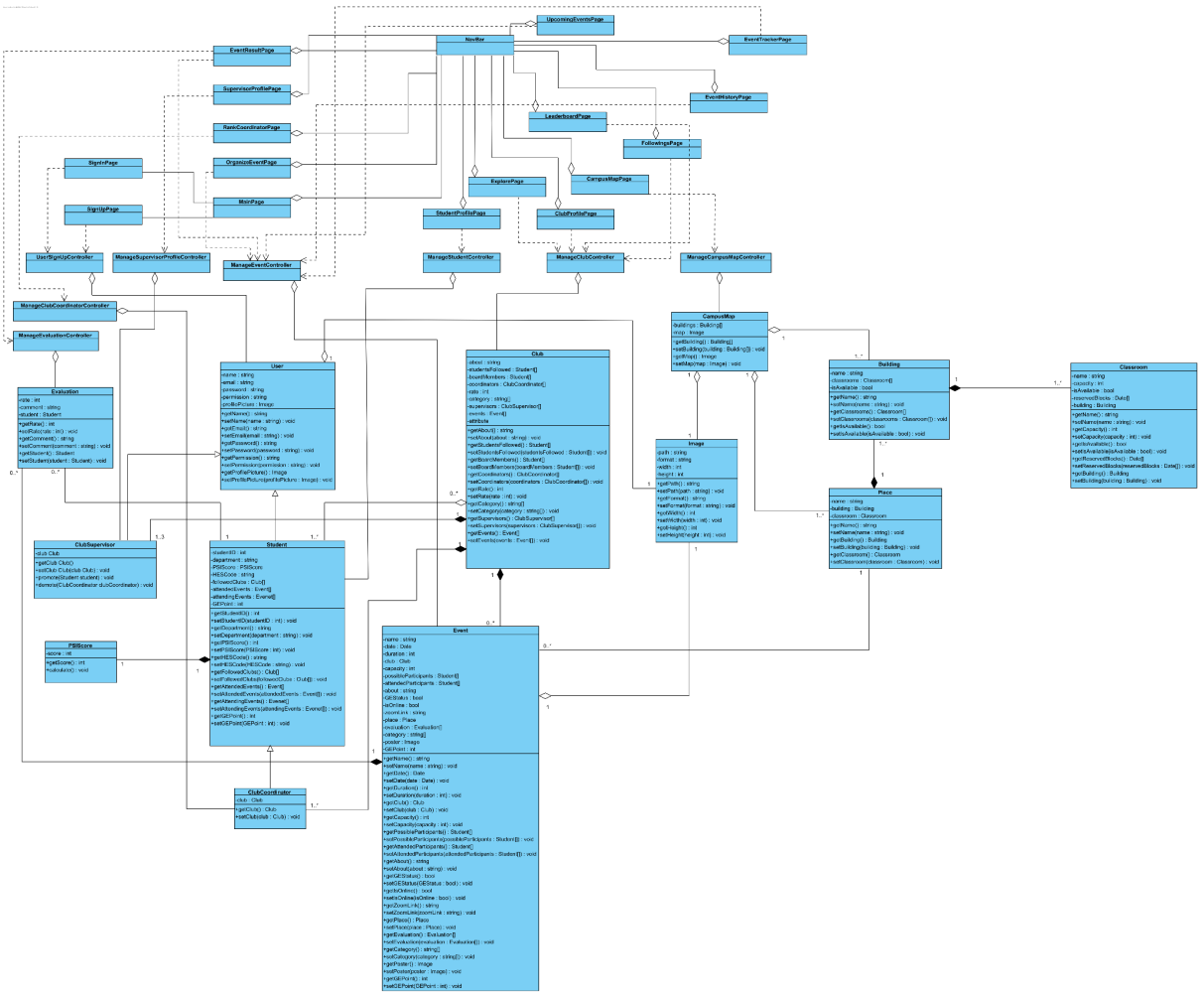

Figure 5. Classes of Entity Objects

3.3 Packages

3.3.1 src.boundary Package

Boundary has no classes since it only creates a namespace.

3.3.2 src.boundary.studentInterface Package

This package contains the classes which are related to the user interface from the student's perspective.

3.3.3 src.boundary.coordinatorInterface Package

This package contains the classes which are related to the user interface from the club coordinator's perspective.

3.3.4 src.boundary.supervisorInterface Package

This package contains the classes which are related to the user interface from the club supervisor's perspective.

3.3.5 src.controller Package

Controller has no classes since it only creates a namespace.

3.3.6 src.controller.userManagement Package

This package includes the controller classes for the operations of the management of the user information.

3.3.7 src.controller.eventManagement Package

This package includes the controller classes for the operations of the management of the event information.

3.3.8 src.controller.clubManagement Package

This package includes the controller classes for the operations of the management of the student club information.

3.3.9 src.controller.mapManagement Package

This package includes the controller classes for the operations of the management of the campus map information.

3.3.10 src.controller.signInSignUpManagement Package

This package includes the controller classes for the operations of the management of the authentication information.

3.3.11 src.entity Package

Entity has no classes since it only creates a namespace.

3.3.12 src.entity.user Package

This package contains classes of three user types which are Student, ClubCoordinator, and ClubSupervisor and the related classes to the user classes.

3.3.13 src.entity.event Package

This package contains the creation and management of event objects and other related classes.

3.3.14 src.entity.club Package

This package contains the student club class and the helper classes for this one.

3.3.15 src.entity.map Package

This package contains the classes related to the campus map such as place, building, classroom and the management of all these objects.

3.4 Class Interfaces

3.4.1 Boundary Classes

3.4.1.1 NavBar Class

Explanation: This class includes the main operations buttons of the application.

Methods:
goToStudentProfile(): This method leads the user to enter his/her profile page.

goToExplore(): This method allows users to travel to the exploring clubs page. There are clubs on that page.

goToFollowings(): This method allows the user to go to the page including clubs followed by that user.

goToUpcomingEvents(): This method allows the user to visit the upcoming events page. In that page this user can see the all events or events performed by the club this user is following.

goToEventHistory(): This method opens the event history page on the screen and allows the user to visit this page.

goToEventTracker(): This method allows the user to enter the event tracker page and displays this page on the screen.

goToCampusMap(): This method helps the user enter the campus map page and opens this page on the screen.

goToLeaderboard(): This method opens the leaderboard page and helps the user examine the leaderboard of clubs.

goToClubProfile(): This method displays the club profile page for club supervisor and club coordinator.

goToOrganizeEvent(): This method allows the club coordinator to organize an event. Firstly this method opens the organize event page.

goToEventResult(): This method leads the users to see the specific event result and displays the event result page on the screen.

goToSupervisorProfile(): This method helps the club supervisor enter his/her profile page.

goToRankCoordinator(): This method allows the club supervisor to change the status of a student. The club supervisor can promote a student to club coordinator or can demote a club coordinator to a student.

3.4.1.2 SignInPage Class

Explanation: This class represents the user interface of a page that includes the student identification number and password areas for users to login.

Methods:
submitLoginInfo(email: string, password: string): This method checks the email and password data entered whether there is such a user on the database or not.
goToSignUp(): This method changes the screen from sign in to sign up..

3.4.1.3 SignUpPage Class

Explanation: This class represents the user interface of a page that includes the name, identification number, department password and email areas for the users to create an account.

Methods:
submitSignUpInfo(email: string, name: string, id: string, department: string, password: string): This method submits the information entered by the user to the database and creates a new account on the database.
goToSignIn(): This method changes the screen from sign up to sign in.

3.4.1.4 StudentProfilePage Class

Explanation: This class represents the user interface of a page that includes the profile photograph, name, id, email, department, PSI score, and HES code of the user.

Methods:
updateProfilePicture(picture: Image): This method allows the user to change his/her profile photograph.
submitHesCode(hesCode: string): This method allows the user to submit the HES code to the database.
changePassword(oldPassword: string, newPassword: string): This method allows the user to change his/her password.
logOut(): This method allows the user to log out and end the session.

3.4.1.5 ExplorePage Class

Explanation: This class represents the user interface of a page that includes a list of the clubs registered on the system.

Methods:
searchByClubName(clubName: string): This method helps the user search the clubs by entering the club's name.
selectCategory(): This method allows the user to select a category of clubs for filter operation.
unselectCategory(): This method is used to cancel the selection when the selected category is clicked again.
follow(): This method allows the user to follow clubs and adds that club to the list of clubs the user follows.
unfollow(): This method allows the user to unfollow the clubs and removes that club from the list of clubs the user follows.

3.4.1.6 FollowingsPage Class

Explanation: This class represents the user interface of a page that includes the list of clubs followed by the student entered this page.

Methods:
searchByClubName(clubName: string): This method helps the user search the clubs by entering the club's name.
selectCategory(): This method allows the user to select a category of clubs for filter operation.
unselectCategory(): This method is used to cancel the selection when the selected category is clicked again.
unfollow(): This method allows the user to unfollow the clubs and removes that club from the list of clubs the user follows.

3.4.1.7 UpcomingEventsPage Class

Explanation: This class represents the user interface of a page that includes the events of the clubs.

Methods:
searchByClubName(clubName: string): This method helps the user search the clubs by entering the club's name.
selectCategory(): This method allows the user to select a category of events for filter operation.
unselectCategory(): This method is used to cancel the selection when the selected category is clicked again.

goToAllEvents(): This method allows the user to see all events performed by the clubs registered on the system.
goToFollowingsEvents(): This method allows the user to see events performed by the clubs the user follows.
attend(): This method allows the user to register the event.
cancel(): This method allows the user to cancel registration of the event.

3.4.1.8 EventHistoryPage Class

Explanation: This class represents the user interface of a page that includes the information of events the user attended.

Methods:
searchByClubName(clubName: string): This method helps the user search the clubs by entering the club's name.
selectCategory(): This method allows the user to select a category of events for filter operation.
unselectCategory(): This method is used to cancel the selection when the selected category is clicked again.
goToAllEvents(): This method allows the user to see all events performed by the clubs registered on the system.
goToFollowingsEvents(): This method allows the user to see events performed by the clubs the user follows.
rate(): This method allows the user to rate the events that the user attended out of 5.

3.4.1.9 EventTrackerPage Class

Explanation: This class represents the user interface of a page that includes the calendar showing the date of events.

Methods:
goToNextMonth(): This method moves the month that appears in the calendar on the page to the next month.
goToPreviousMonth(): This method moves the month that appears in the calendar on the page to the previous month.
updateSyncAccount(email: string, password: string): This method allows the user to update the information of Google Account which is synchronized with the event tracker data.
sync(): This method is called by clicking the sync button and starts the synchronization process.

3.4.1.10 CampusMapPage Class

Explanation: This class represents the user interface of a page that includes maps for the user and users can select the building they want to go to and see it on the map.

Methods:
selectBuilding(): This class allows the user to select a building in the campus of Bilkent University on the drop-down menu.

3.4.1.11 LeaderboardPage Class

Explanation: This class represents the user interface of a page that includes the leaderboard of the clubs. Users can filter this leaderboard.

Methods:
searchByClubName(clubName: string): This class allows the user to search the student clubs by name on the leaderboard.
sortBy(parameter: string): This class allows the user to sort the student clubs by a specified parameter on the leaderboard.

3.4.1.12 ClubProfilePage Class

Explanation: This class represents the user interface of a page that includes the profile page of a club. This page can only be accessed by the club supervisor and the club coordinator.

Methods:
editAbout(text: string): This method is called by only the club coordinator and allows the club coordinator to edit the general information about the student club.
editCategory(text: string): This method is called by only the club coordinator and allows the club coordinator to edit the category of the club.

3.4.1.13 OrganizeEventPage Class

Explanation: This class represents the user interface of a page that allows the club coordinator to organize an event assigned to the club.

Methods:
submitNewEvent(name: string, description: string, location: string, date: Date): This method enables the club coordinator to submit the information about the new event desired to be created.
editEvent(name: string, description: string, location: string, date: Date): This method enables the club coordinator to edit the already existing event's information.

3.4.1.14 EventResultPage Class

Explanation: This class represents the user interface of a page that contains the past event results.

Methods:
showEventResult(): This method provides the club coordinator and supervisor with inspecting the result of the past events.

3.4.1.15 SupervisorProfilePage Class

Explanation: This class represents the user interface of a page that includes the user information of the club supervisor.

Methods:
updateProfilePicture(picture: Image): This method provides the supervisor with a new profile picture for the supervisor's account.
changePassword(oldPassword: string, newPassword: string): This method allows the supervisor to change the password.
logOut(): The website can be logged out using this method.

3.4.1.16 RankCoordinatorPage Class

Explanation: This class represents the user interface of a page that includes the promoting and demoting operations by the club supervisor.

Methods:
promote(name: string, id: string, email: string): This method allows the club supervisor to promote a student to the rank of the club coordinator.
demote(): This method allows the club supervisor to demote a club coordinator to the rank of the student.

3.4.2 Controller Classes

3.4.2.1 UserSignUpController Class

Explanation:
Methods:

3.4.2.2 ManageClubCoordinatorController Class

Explanation:
Methods:

3.4.2.3 ManageSupervisorProfileController Class

Explanation:

Methods:

3.4.2.4 ManageEventController Class

Explanation:
Methods:

3.4.2.5 ManageStudentController Class

Explanation:
Methods:

3.4.2.6 ManageClubController Class

Explanation:
Methods:

3.4.2.7 ManageCampusMapController Class

Explanation:
Methods:

3.4.2.8 ManageEvaluationController Class

Explanation:
Methods:

3.4.3 Entity Classes

3.4.3.1 User Class

Explanation: This class is used for the users using the application such as students, club supervisor, and club coordinator.
Methods:

3.4.3.2 ClubSupervisor Class

Explanation: This class contains the management of club supervisor objects and its operations. This class extends the User class.
Methods:

3.4.3.3 Student Class

Explanation: This class contains the management of student objects and its operations. This class extends the User class.
Methods:

3.4.3.4 ClubCoordinator Class

Explanation: This class contains the management of club coordinator objects and its operations. This class extends the User class.

Methods:

3.4.3.5 Event Class

Explanation: This class includes the fundamental operations of the events objects such as changing the time or getting rate.
Methods:

3.4.3.6 Club Class

Explanation: This class includes the fundamental operations of the club objects such as changing the name, listing followers.
Methods:

3.4.3.7 Place Class

Explanation: This class is used for the place of the event objects and it includes building and classroom attributes.
Methods:

3.4.3.8 Building Class

Explanation: This class contains the management of building objects and its operations. This class extends the Place class.
Methods:

3.4.3.9 Classroom Class

Explanation: This class contains the management of classroom objects and its operations. This class extends the Place class.
Methods:

3.4.3.10 CampusMap Class

Explanation: This class is used for the campus map. It includes the place of buildings.
Methods:

3.4.3.11 Image Class

Explanation: This class is used for the users' and events' profile photographs. This class extends the Python image class.

Methods:

3.4.3.12 Evaluation Class

Explanation: This class is used for the evaluation of the events. It stores the comments and rates of the event made by the users.
Methods:

3.4.3.13 PSIScore Class

Explanation: This class is used for the Personal Sociability Intelligence score of the users. The object of this class is affected by the activities of the user such as participating in events, following clubs and so on.

Methods:

4. Glossary & References

4.1 Glossary

PSI: This term is the abbreviation of Personal Sociability Intelligence. This score indicates how social a person is. This score will be calculated by an algorithm determined by the system. The events that the user participates in, the clubs the user follows, the clubs the user is the board member of will affect the calculation of this score.

4.2 References

[1] “Documentation - Chrome Developers”. https://developer.chrome.com/docs/. [Accessed: Nov 24, 2021].

[2] “Microsoft Edge Documentation”. https://docs.microsoft.com/en-us/microsoft-edge/. [Accessed: Nov 24, 2021].

[3] “Firefox - Mozilla”. https://docs.microsoft.com/en-us/microsoft-edge/. [Accessed: Nov 24, 2021].

[4] “Opera Version History”. https://help.opera.com/en/opera-version-history/. [Accessed: Nov 24, 2021].

[5] “Resources - Safari - Apple Developers”. https://developer.apple.com/safari/resources/. [Accessed: Nov 24, 2021].

[6] “Django”. https://www.djangoproject.com/. [Accessed: Nov 24, 2021].

[7] “The Progressive JavaScript Framework”. https://vuejs.org/. [Accessed: Nov 24, 2021].

[8] “PostgreSQL: The World's Most Advanced Open Source Relational Database”. https://www.postgresql.org/. [Accessed: Nov 24, 2021].

[9] “AWS Documentation”. https://docs.aws.amazon.com/. [Accessed: Nov 24, 2021].