D1: Use Cases & Non-Functional Requirements & Tech. Stack

S3T11

BilFind

Serhat Merak **(22002414)**
Mirza Özgür Atalar **(22003455)**
Kaan Türkoğlu **(22102105)**
Ege Karaahmetoğlu **(22102622)**
Orhun Aysan **(22103005)**

Table of Contents

Use Cases

Login

Use Case Name	Login	
Actors	Buyers, Sellers, Admins	
Description	This use case is provided when the users want to sign in the system.	
Reference		
Typical Course of Events	**Actor Action** **Step 1:** This use case is initiated when a user tries to enter the system. The user enters the required Bilkent email and password.	**System Response** **Step 2:** The system checks whether the email is in the database. **Step 3:** Check whether the user matches an email-password combination. **Step 4:** The authentication token is returned to the user. **Step 5:** The user is redirected to the home page.
Alternate Courses	**Step 1:** If a problem occurs during validating the user, the login page is reloaded so that the validation process starts over. **Step 2:** The client is redirected to the register page if the email is not registered to the system. **Step 3:** The client sees an error if the provided password is wrong.	
Pre-condition	-	
Post-condition	If the login is successful, the user will be redirected to the home page.	
Assumptions	None	

Logout

Use Case Name	Logout	
Actors	Buyers, Sellers, Admins	
Description	This use case is provided when the users want to exit from the system.	
Reference		
Typical Course of Events	**Actor Action** **Step 1:** This use case is initiated when a user wants to exit from the system.	**System Response** **Step 2:** The authentication token is deleted from the user. **Step 3:** The user is redirected to the login page.
Alternate Courses	None	
Pre-condition	The user had already signed in to the system.	
Post-condition	If the login is successful, the user will be redirected to the login page.	
Assumptions	None	

Register

Use Case Name	Register	
Actors	Buyers, Sellers	
Description	This use case is shown when the users want to sign up for the system.	
Reference		
Typical Course of Events	**Actor Action** **Step 1:** The user who wants to join the BilFind is shown the register page. **Step 2:** The user enters the necessary information so that he/she can be validated. **Step 4:** The user submits the code taken from the email.	**System Response** **Step 3:** The system checks whether the new user is a member of Bilkent University by sending an email to him/her. **Step 5:** If the provided code matches the sent code, the user is saved to the database **Step 6:** The authentication token is returned to the user. **Step 7:** The user is redirected to the home page
Alternate Courses	**Step 1:** If a problem occurs during validating the user, the registration page is reloaded so that the validation process starts over. **Step 2:** The user sees an error if the provided email is not a Bilkent Email.	
Pre-condition	The user must be a member of Bilkent University.	
Post-condition	User will be redirected to the home page	
Assumptions	The user is a member of Bilkent community.	

Ban User

Use Case Name	Ban User	
Actors	Admin	
Description	This use case occurs when the admin removes a user from the system if needed.	
Reference		
Typical Course of Events	**Actor Action** **Step 1:** This use case happens when the admin decides to delete a user from the system. **Step 5:** Admin sees a success message.	**System Response** **Step 2:** The admin authentication token and user specifications are sent to the backend **Step 3:** The admin authentication token is validated. **Step 4:** The user is deleted from the database, and a successful response is returned to the user.
Alternate Courses	**Step 1:** If a problem occurs during validating the admin, the user page is redirected to the login screen to fix this issue.	
Pre-condition	The user must be an admin.	
Post-condition	Admin will be shown a message.	
Assumptions	None	

Forgot Password

Use Case Name	Forgot Password	
Actors	Buyers, Sellers, Donators, Receivers	
Description	This use case occurs when a member of BilFind forgets their password.	
Reference		
Typical Course of Events	**Actor Action** **Step 1:** This use case is initiated when a user clicks the “Forgot Password” option on the login screen. **Step 2:** The user enters his/her Bilkent email and sends it to the Backend **Step 5:** The user clicks the link from his/her email. **Step 6:** Enter the new password for his account and send it to the backend. **Step 8:** The user can log in with the new password.	**System Response** **Step 3:** Member details are fetched from the database to check whether the user exists. **Step 4:** An email containing a reset password link is sent to the user. **Step 7:** New password and reset password token is validated and saved to the database**.**
Alternate Courses	**Step 1:** The email verification code is re-sent if a problem occurs while validating the user. **Step 2:** If no member has the provided user id, a pop-up occurs to inform the user. After that, the user is redirected to the login page.	
Pre-condition	The user is already on the system, so an email can be sent to validate and allow the user to get a new password	
Post-condition	None	
Assumptions:	The user is already registered to BilFind.	

Edit Profile

Use Case Name	Edit Profile	
Actors	Buyers, Sellers	
Description	This use case describes the process of editing the user profile	
Reference		
Typical Course of Events	**Actor Action** **Step 1:** This use case is initiated when a user clicks on “edit profile“ button which is on the profile page. **Step 2:** After the editing process is over, the new information is posted to the database. **Step 6:** This use case concludes when the user is done with editing the profile according to the needs.	**System Response** **Step 3:** Member’s personal information and security token is validated. **Step 4;** The database is updated with the given data **Step 5:** Success response is returned to the user
Alternate Courses	**Step 1:** If a problem occurs during validating the user, the user page is redirected to the login screen to fix this issue.	
Pre-condition	The user has to be signed in.	
Post-condition	None	
Assumptions	The user is a member of BilFind	

Search Post

Use Case Name	Search Post	
Actors	Buyers	
Description	This use case describes the process of buyers searching for a desired post using the search bar or the filters given.	
Reference		
Typical Course of Events	**Actor Action** **Step 1:** This use case is initiated when a user uses the search bar or filters in the search screen **Step 5:** This use case concludes when the user sees the list of posts.	**System Response** **Step 2:** Member's personal information and security token is validated. **Step 3:** Posts are listed according to the given filters and provided text taken from the user **Step 4:** The list of posts is returned to the client from the backend.
Alternate Courses	**Step 1:** If a problem occurs during validating the user, the user page is redirected to the login screen to fix this issue. **Step 2:** If there is no post based on the given filters, the user sees a message indicating it.	
Pre-condition	The user has to be signed in.	
Post-condition	The list of post is shown on the webpage.	
Assumptions	None	

Report Post

Use Case Name	Report Post	
Actors	Buyers	
Description	This use case describes the process of reporting an inappropriate post.	
Reference		
Typical Course of Events	**Actor Action** **Step 1:** This use case is initiated when a user finds a post inappropriate and reports it. **Step 5:** This use case concludes when the user completes the report process.	**System Response** **Step 2:** The user's personal information and security token is validated. **Step 3:** Post details are fetched from the database. **Step 4:** Report details are posted to the database, and a notification is sent to the admins.
Alternate Courses	**Step 1:** If a problem occurs during validating the user, the user page is redirected to the login screen to fix this issue. **Step 2:** If there is no post related to the report, the user sees a message indicating it.	
Pre-condition	The user has to be signed in.	
Post-condition	The user is redirected to the post's page.	
Assumptions	None	

View Post Details

Use Case Name	View Post Details	
Actors	Buyers	
Description	This use case describes the process of buyers viewing the post details listed on the page.	
Reference		
Typical Course of Events	**Actor Action** **Step 1:** This use case is initiated when a user clicks on the post card listed on the page. **Step 5:** This use case concludes when the user is redirected to the post detail page with the desired post.	**System Response** **Step 2:** Member's personal information and security token is validated. **Step 3:** Post details are fetched from the database. **Step 4:** The resulting post object is returned to the client.
Alternate Courses	**Step 1:** If a problem occurs during validating the user, the user page is redirected to the login screen to fix this issue. **Step 2:** If there is no post with the provided user id on the post detail page, the post not found label is indicated for the user.	
Pre-condition	The user has to be signed in, and the given post id must be valid.	
Post-condition	Post details are shown on the webpage.	
Assumptions	None	

Destroy Post

Use Case Name	Destroy Post	
Actors	Admin	
Description	This use case occurs when an admin wants to delete a post.	
Reference		
Typical Course of Events	**Actor Action** **Step 1:** This use case is initiated when an admin wants to delete an inappropriate post **Step 2:** Post id and admin authentication token are sent to the backend. **Step 5:** Admin sees a success message	**System Response** **Step 3:** The admin authentication token is validated. **Step 4:** The post is deleted from the database, and a successful response is returned to the user.
Alternate Courses	**Step 1:** If a problem occurs during validating the admin, the user page is redirected to the login screen to fix this issue.	
Pre-condition	The user should be an admin	
Post-condition	None	
Assumptions:	The user is already registered to BilFind.	

Create Post

Use Case Name	Create Post	
Actors	Sellers	
Description	This use case describes the process of putting new posts to the market	
Reference		
Typical Course of Events	**Actor Action** **Step 1:** This use case is initiated when a user clicks on the create new post (+) button. **Step 2:** The user selects the information and photos of the post. **Step 6:** This use case concludes when the user is redirected to the post detail page which he created currently.	**System Response** **Step 3:** Member's personal information and security token is validated. **Step 4:** Post details are pushed to the database. **Step 5:** If successful, the backend returns success as a response.
Alternate Courses	**Step 1:** If a problem occurs during validating the user, the user page is redirected to the login screen to fix this issue.	
Pre-condition	The user has to be signed in, and the given post id must be valid.	
Post-condition	Details of the new post is shown on the webpage.	
Assumptions	None	

Remove Post

Use Case Name	Remove Post	
Actors	Sellers	
Description	This use case describes the process of removing the existing post from the market by its owner	
Reference		
Typical Course of Events	**Actor Action** **Step 1:** This use case is initiated when a user clicks the delete post button. **Step 5:** This use case concludes when the user is redirected to the post list page after deleting successfully	**System Response** **Step 2:** Member's personal information and security token is validated. **Step 3:** Post details are fetched from the database, and ownership of the post is validated. **Step 4:** Data about posts are deleted from the backend.
Alternate Courses	**Step 1:** If a problem occurs during validating the user, the user page is redirected to the login screen to fix this issue. **Step 2:** If the request sender user does not own the given post id, the user sees an error.	
Pre-condition	The user has to be signed in, and that user must own the given post id.	
Post-condition	Details of the new post are shown on the webpage.	
Assumptions	None	

Add to Favorites

Use Case Name	Add to Favorites	
Actors	Buyers	
Description	This use case describes the process of users saving a post as a favorite.	
Reference		
Typical Course of Events	**Actor Action** **Step 1:** This use case is initiated when a user clicks the heart button post button. **Step 4:** This use case concludes when the user sees the red heart icon on the post card.	**System Response** **Step 2:** Member's personal information and security token is validated. **Step 3:** If the post is valid, it is saved for the user's favorite list in the database.
Alternate Courses	**Step 1:** If a problem occurs during validating the user, the user page is redirected to the login screen to fix this issue.	
Pre-condition	The user has to be signed in, and that user must own the given post id.	
Post-condition	The red heart is seen on the card	
Assumptions	None	

Chat

Use Case Name	Chat	
Actors	Buyers, Sellers,	
Description	Users can send private messages to each other.	
Reference		
Typical Course of Events	**Actor Action** **Step 1:** This use case is initiated when a buyer wants to communicate with the post owner or reply to the taken messages. **Step 5:** This use case concludes when the user sees the sent message.	**System Response** **Step 2:** Member's personal information and security token is validated. **Step 3:** The Target user is fetched. If it is valid, the message is saved to the database. **Step 4:** Push notification is sent to the target user
Alternate Courses	**Step 1:** If a problem occurs during validating the user, the user page is redirected to the login screen to fix this issue.	
Pre-condition	The user has to be signed in, and the target user must be valid.	
Post-condition	User sees the sent message to user	
Assumptions	None	

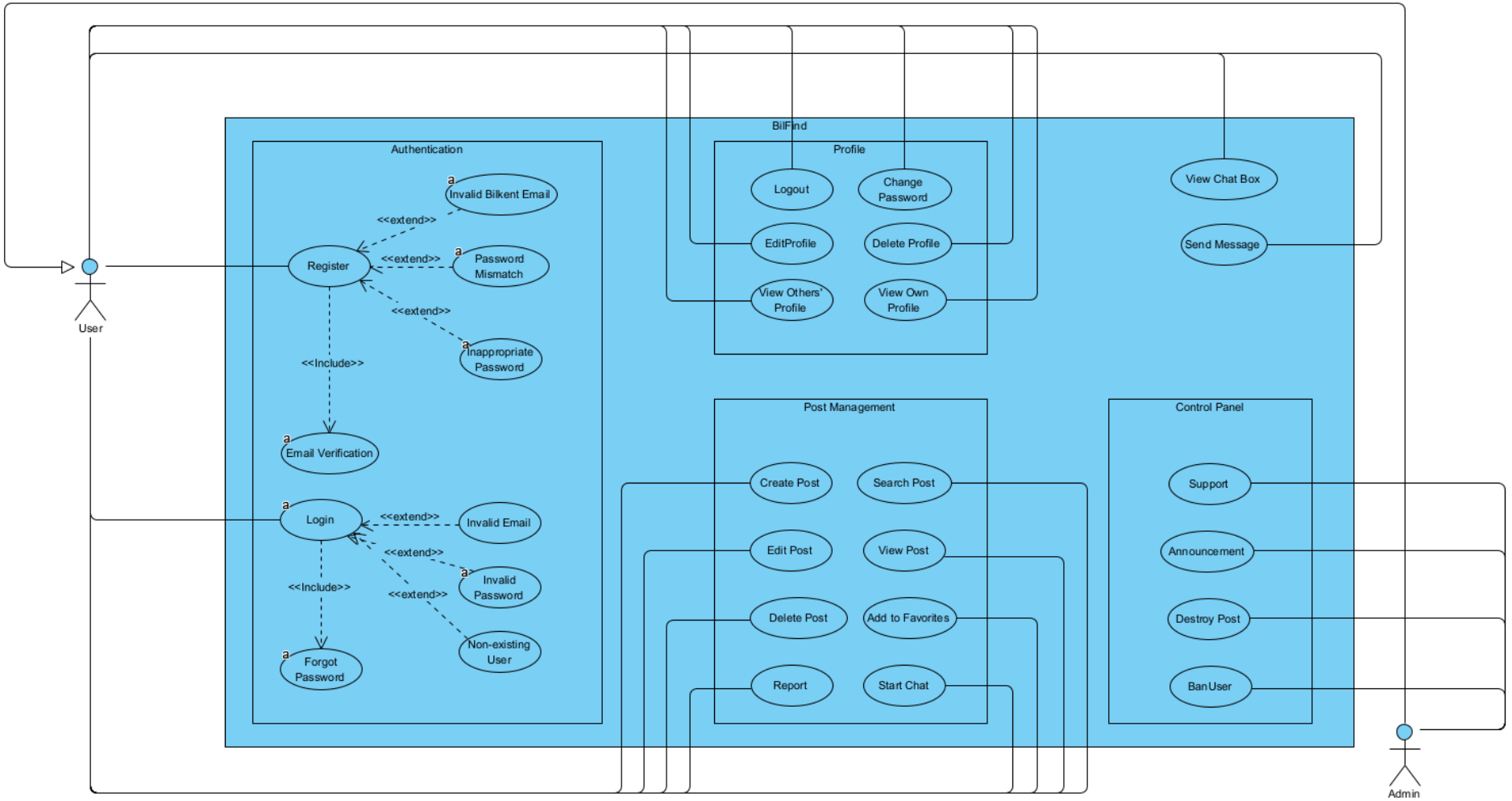
View Profile

Use Case Name	View Profile	
Actors	User, admin	
Description	This use case describes the process of viewing the user profile	
Reference		
Typical Course of Events	**Actor Action** **Step 1:** This use case is initiated when a user clicks on the profile photo on the post card **Step 5:** This use case concludes when the user sees the profile details.	**System Response** **Step 2:** The page is redirected to the profile details screen. **Step 3:** Profile detail request is sent to the backend **Step 4;** If the request is valid, user detail is returned
Alternate Courses	**Step 1:** If a problem occurs while validating the user, the user page is redirected to the login screen to fix this issue.	
Pre-condition	The user has to be signed in.	
Post-condition	None	
Assumptions	The user is a member of BilFind	

Take Support

Use Case Name	Take Support	
Actors	Buyers, Sellers,	
Description	Users can send emails to admins when they encounter a problem.	
Reference		
Typical Course of Events	**Actor Action** **Step 1:** This use case is initiated when a user clicks the '?' icon. **Step 5:** This use case concludes when the user sees a success message.	**System Response** **Step 2:** Member's personal information and security token is validated. **Step 3:** The message retrieved from the user is saved to the database. **Step 4:** The success response is returned to the user.
Alternate Courses	**Step 1:** If a problem occurs during validating the user, the user page is redirected to the login screen to fix this issue.	
Pre-condition	The user has to be signed in.	
Post-condition	User sees a popup message.	
Assumptions	None	

Use Case Diagram



Non-Functional Requirements

1) Performance:

- **Response Time:** The application should have fast response times for user interactions, with a goal of under 5 seconds for most of the actions both in the frontend and the backend.
- **Scalability:** The system should be scalable to accommodate a growing number of users and items in the marketplace.
- **Load Handling:** The application should be able to handle a large number of concurrent users without degrading performance.

2) Security:

- **Data Security:** User data, especially personal and financial information, will be securely stored and transmitted with the help of modern and secure encryption and database systems such as MongoDB.
- **Authentication:** Secure user authentication will be provided, and only users from Bilkent University will be authenticated via their Bilkent-assigned e-mails. Third-party members unrelated to Bilkent University will not be able to interfere with the user data.
- **Privacy:** User privacy and data protection must be a top priority, complying with relevant data protection regulations.

3) Usability:

The application should have a user-friendly and visually appealing interface, with the goal of making users comfortable upon using the application. Also, in the interface, each app feature should be easily distinguished and provide clear information about themselves.

4) Reliability:

- **Availability:** The system should be available 24/7 with minimal downtime for maintenance.

- **Data Integrity:** All user data, listings, and messages should be stored securely and reliably. Data loss or corruption should be prevented via frequent database backups.

5) Maintainability:

Throughout the implementation, Object-Oriented Programming will be used in order to extend and improve the features in time. Moreover, the codebase will be well-documented and follow coding standards for easy maintenance and for further development. These approaches will also make debugging and fixing bugs easier, as it will facilitate finding errors in the code.

Tech Stack

We decided on our tech stack primarily considering the limited time and the flexibility requirements. We also wanted to use modern technologies in the continuously developing world.

1. **Flutter (Frontend):** In our front-end application, we decided to use Flutter for its flexibility. Our main product will be a web application. However, in today's world, we observed that the mobile application for several needs at Bilkent University would be as efficient as the web application. Therefore, we needed to create a web, Android, and iOS application in a limited time, so Flutter is one of the best options to ensure this aim. It is possible to adopt the written code to both mobile and web applications with only small adjustments. Also, it is obvious that almost every application follows similar components. Therefore, with the very rich open-source library, Flutter was the best option for us to develop our application with the least effort.

2. **NodeJS (Backend):** Our backend will be structured over NodeJs technology supported with multiple libraries. By using **ExpressJs**, we will create a flexible and well-structured RESTful API for our application. Since we will write the backend considering the Model View Controller (MVP) structure, it is quite readable and scalable to write with the Express library. However, using solely JavaScript in the backend may produce some ambiguity among the group members because it would be harder to maintain code as it grows without type-checking. Therefore, we decided to utilize **TypeScript** for our backend project to use static typing and Object Oriented Concepts in the backend architecture.

3. **MongoDB (Database):** In our application, we observed that we need flexible data models because even though we are planning the whole development process, it is likely for data models to

evolve over and over. Therefore, MongoDB’s support for flexibility was the key factor in our decision. Additionally, with the query optimization support, it is easier to scale the datasets while protecting the efficiency. Also, MongoDB is very suitable for startups because it is free up to a certain level.