Non-Functional Requirements

1) Performance:

The system should respond to user requests within 2 seconds on average.
The system should support a minimum of 1,000 simultaneous users.
It should be capable of handling at least 100 transactions per minute during peak hours.

2) Scalability:

The system should be easily scalable to accommodate an increasing number of users and products without significant performance degradation.
The web app should support horizontal scaling and load balancing to distribute traffic effectively.

3) Availability:

The system should be available 24/7, with planned downtimes not exceeding 1 hour per month.
It should have automated failover and redundancy mechanisms to ensure high availability.

4) Security:

User data must be stored securely and comply with relevant data protection regulations.
Implement secure user authentication and authorization mechanisms to protect user accounts and sensitive data.
Regular security audits and penetration testing should be performed.

5) Compatibility:

The web app should be compatible with major web browsers (Chrome, Firefox, Safari, and Edge) and various devices (desktops, tablets, and smartphones).
Usability:
The user interface should be intuitive and user-friendly, requiring minimal training for users to navigate the system.

6) Maintainability:

The design should follow the Object-Oriented Programming understanding and the codebase should be well-documented and follow coding standards for easy maintenance to allow additional features in the future.

Use Case Name	Login	
Actors	Buyers, Sellers, Donators, Receivers	
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 username and password. **Step 5:** This use case concludes when the user is directed to the homepage.	**System Response** **Step 2:** The system checks whether the username-password combination is in the database. If not, it asks the user to re-enter the username and password as well as the option to sign up. **Step 3:** Member details are fetched from the database. **Step 4:** The member is directed to the homepage.
Alternate Courses	**Step 1:** If a problem occurs during validating the user, the login page is reloaded so that the validation process starts over.	
Pre-condition	-	
Post-condition	If the login is successful the homepage will be shown and if not the user will be redirected to the login page again.	
Assumptions	None	

Use Case Name	Register	
Actors	Buyers, Sellers, Donators, Receivers	
Description	This use case is shown when the users want to sign up to 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:** User enters the necessary information so that he/she can be validated.	**System Response** **Step 3:** After step 2, the system checks whether the new user is a member of Bilkent University by sending an email to him/her. If not, the system denies the registration process and the user is redirected to the registration page once more. **Step 4:** In case of a successful registration process, member details are posted to the database. **Step 5:** The member is directed to the homepage.
Alternate Courses	**Step 1:** If a problem occurs during validating the user, the registration page is reloaded so that the validation process starts over.	
Pre-condition	The user must be a member of Bilkent University.	
Post-condition	If the registration is successful the homepage will be shown and if not, the user will be redirected to the registration page again.	
Assumptions	The user is a member of Bilkent community.	

Use Case Name	Search Product
Actors	Buyers
Description	This use case describes the process of buyers searching for a desired product using the search bar or the filters given.
Reference	

Typical Course of Events	**<u>Actor Action</u>** **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 products.	**<u>System Response</u>** **Step 2:** Member's personal information and security token is validated. **Step 3:** Products are listed according to the given filters and provided text taken from the user **Step 4:** The list of products 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 product 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 products is shown on the webpage.	
Assumptions	None	

Use Case Name	View Product Details
Actors	Buyers
Description	This use case describes the process of buyers viewing the product details listed on the page.
Reference	

Typical Course of Events	**<u>Actor Action</u>** **Step 1:** This use case is initiated when a user clicks on the product card listed on the page. **Step 5:** This use case concludes when the user is redirected to the product detail page with the desired product.	**<u>System Response</u>** **Step 2:** Member's personal information and security token is validated. **Step 3:** Product details are fetched from the database. **Step 4:** The resulting product 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 product with the provided user id on the product detail page, the product not found label is indicated for the user.	
Pre-condition	The user has to be signed in, and the given product id must be valid.	
Post-condition	Product details are shown on the webpage.	
Assumptions	None	

Use Case Name	Put Product
Actors	Sellers
Description	This use case describes the process of putting new products to the market
Reference	

Typical Course of Events	**<u>Actor Action</u>** **Step 1:** This use case is initiated when a user clicks on the create new product (+) button. **Step 2:** The user selects put the information and photos of the product. **Step 5:** This use case concludes when the user is redirected to the product detail page which he created currently.	**<u>System Response</u>** **Step 3:** Member's personal information and security token is validated. **Step 3:** Product details are pushed to the database. **Step 4:** 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 product id must be valid.	
Post-condition	Details of new product is shown on the webpage.	
Assumptions	None	

Use Case Name	Remove Product
Actors	Sellers
Description	This use case describes the process of removing the existing product from market by its owner

Reference		
Typical Course of Events	**Actor Action** **Step 1:** This use case is initiated when a user clicks the delete product button. **Step 5:** This use case concludes when the user is redirected to the product list page after deleting successfully	**System Response** **Step 2:** Member's personal information and security token is validated. **Step 3:** Product details are fetched from to the database and ownership of the product is validated. **Step 4:** Data about products 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 product id, the user sees an error.	
Pre-condition	The user has to be signed in, and that user must own the given product id.	
Post-condition	Details of new product is shown on the webpage.	
Assumptions	None	

Use Case Name	Add to Favorites
Actors	Buyers
Description	This use case describes the process of users saving a product as a favorite.

Reference		
Typical Course of Events	**Actor Action** **Step 1:** This use case is initiated when a user clicks the heart button product button. **Step 5:** This use case concludes when the user sees the red heart icon on the product card.	**System Response** **Step 2:** Member's personal information and security token is validated. **Step 3:** If the product 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 product id.	
Post-condition	The red heart is seen on the card	
Assumptions	None	

Use Case Name	Take Loan
Actors	Buyers
Description	This use case describes the process of users requesting loans for a product from owners.
Reference	

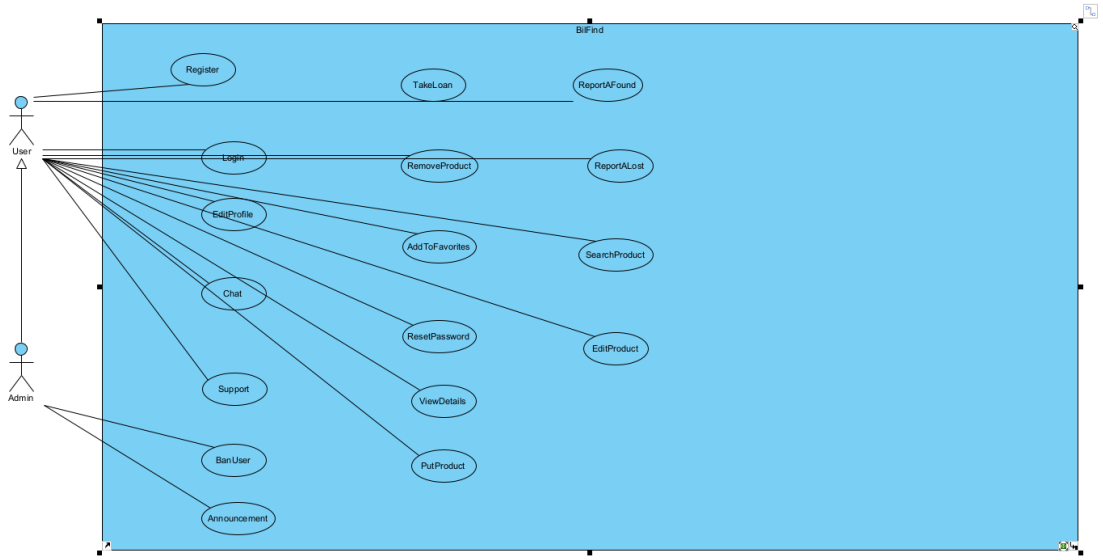
Typical Course of Events	**<u>Actor Action</u>** **Step 1:** This use case is initiated when a user requests a loan from the products allowing it. **Step 5:** This use case concludes when the user is redirected to the message page and sees the custom message sent to the product owner.	**<u>System Response</u>** **Step 2:** Member's personal information and security token is validated. **Step 3:** If the product is valid, a custom message is sent to the product's owner.
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 product id.	
Post-condition	User sees the message that is sent to the product owner.	
Assumptions	None	

Use Case Name	Chat
Actors	Buyers, Sellers,
Description	Users can send private messages to each other.
Reference	

Typical Course of Events	**<u>Actor Action</u>** **Step 1:** This use case is initiated when a buyer wants to communicate with the product owner or reply to the taken messages. **Step 5:** This use case concludes when the user sees the sent message.	**<u>System Response</u>** **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	

Use Case Name	Take Support
Actors	Buyers, Sellers,
Description	Users can send emails to admins when they encounter a problem.

USE CASE DIAGRAM



Tech Stack

We are going to use popular technologies to develop a basic web app with an efficient and sustainable backend structure.

1. **Flutter (Frontend):** We decided to use Flutter because we are planning to develop a mobile application for both iOS and Android while developing a web application with the same code base. With the current developments, without changing the code base a lot, we can transform the one code to all platforms.

2. **NodeJS (Backend):** We are planning to use NodeJs in the backend supported with **TypeScript.** We are going to create the structure with the Express js library. With Express and Typescript, it is possible to use JavaScript with type-safe development and simplifiys routing, handling errors and HTTPS requests.

3. **MongoDB (Database):** Mongodb is a NoSQL database that is free and efficient to store data.