BILBOARD

Team-T4 BugBunny

Use Cases & Non-Functional Requirements & Tech. Stack

DILARA MANDIRACI
BURAK DEMIREL
YUSUF TORAMAN
SILA ÖZEL
EREN HAYRETTIN ARIM

Fall 2023

1. Introduction

This project is a web application that can be accessed only by Bilkent staff for four main operations: buying or selling second-hand products, donating and borrowing things, and posting lost item notices. Additionally, we added a complaint system so that the students could write their complaints freely, and their complaints would not be left unseen.
The students can sell their secondhand products to other students, make donations if they think someone might need that item, post lost notices without having to worry about what to do when they see a lost ID on the road, borrow things that they need for a short time, and finally they can share their complaints.
As can be seen from the above operations, our university does not have an official media for these purposes. With our web application, we will provide secure media for Bilkent students to do these exchanges with peace of mind.

2. Non-Functional Requirements (NFRs)

2.1. Usability

To make our application easy to use, we tried to create a simple user interface design. Here are the things that we have planned:

- As the user enters our application, they will encounter an About page where we inform the user about our application's goals and modules. The user can use the buttons on the navigation bar to navigate to login, register, or about the page.
- Our application will be a responsive web application that can be used on many devices.

2.2. Security

The main purpose of this application is to create a secure media for Bilkent students to exchange items. Also, we are storing some information about the students (such as their mail address, password, etc.) in a secure way so there won't be any data leaks.

- We will store the user's password encrypted.
- The user can access secondhand sales, donation, borrow, lost and found modules, and complaint system only if they are authenticated.

- Only Bilkent staff can use this application. So, they must register with their Bilkent mail address. A verification code will be sent to verify the user's mail address.
- Each user can only have one account so this will reduce the frauds.
- In our complaint system, we will use tools that check inappropriate words so that the users cannot misuse this system.

2.3. Maintainability

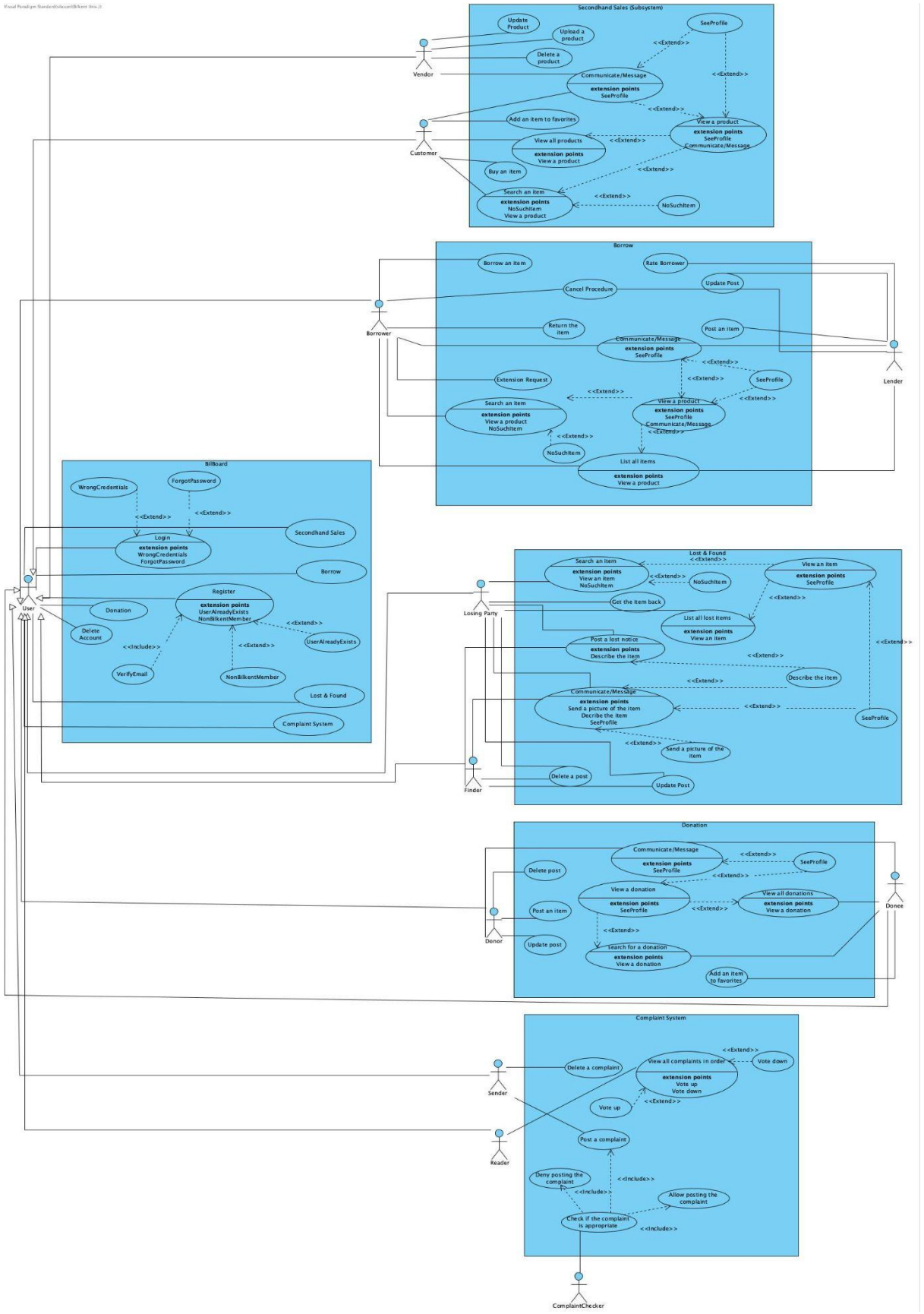
We have decided to implement our frontend and backend in a way that will make our application more maintainable and easy to debug and extend.

- We will have a decoupled frontend and backend code. So, they will only communicate through data. That way, a change in the backend code will not require much change in the frontend code and vice versa. Also, debugging will be easier that way because the concerns are separated.
- Also, we will use the MVC (Model View Controller) design pattern in this application, which will separate the backend into more maintainable parts. Debugging will be much easier since each part will have a certain purpose.

3. Tech Stack

Backend: Django, Django REST
Frontend: React JS, HTML CSS, Bootstrap
Database: PostgreSQL

4. Use Case Diagram



BilBoard Package

1. **Name:** Login
2. **Participating Actor:** Bilkent Member
3. **Entry Condition:** User opens the web application and clicks the “Login” button.
4. **Exit Condition:** User either logs in successfully or runs across a problem.
5. **Flow of Events:**
   5.1. The user enters credentials designated as 'required' (email, password) to log in.
   5.2. If credentials are incorrect,
      5.2.1. This use case extends the “Wrong Credentials" case. The related warning message, displayed for the user.
   5.3. Else if the credentials are correct, the user logs in successfully.
   5.4. If the user cannot remember the credentials, the user uses the forgot password option and changes his/her password again via a verification sent to the user’s specified email.

1. **Name:** Register
2. **Participating Actor:** Bilkent Member
3. **Entry Condition:** The user opens the web application and clicks on the register button to create a new account.
4. **Exit Condition:** The user successfully creates an account, navigates to another page, or cancels the registration.
5. **Flow of Events:**
   5.1. User provides necessary details, such as email, password, and any other required personal information.
   5.2. Else if the email is already registered:
      5.2.1. The system displays a "User already exists" message, and directs the user to the login screen.
      5.2.2. Else if the extension of email is not correct, indicates that the user is not from Bilkent.
      5.2.3. The system displays a “Your email address should include Bilkent extension” message.
   5.3. Else all details are valid:
      5.3.1. A verification email is sent to the user's email address for confirmation.
      5.3.2. The user's account is created.

1. **Name:** Secondhand Sales
2. **Participating Actor:** Customer, Vendor
3. **Entry Condition:** After the user receives an access token via login, the user goes to this page by clicking the Secondhand sales button**.**
4. **Exit Condition:** User navigates from the navigation bar to another module, switches to profile section or leaves from the web application.
5. **Flow of Events:**
   5.1. Secondhand sales subsystem has its own use cases and it is explained in the previous sections.

1. **Name:** Complaint System
2. **Participating Actor:** Sender, Reader, Complaint Checker
3. **Entry Condition:** After the user receives an access token via login, the user goes to this page by clicking the Complaint button**.**
4. **Exit Condition:** The user moves from the navigation bar to another module, switches to profile section or leaves from the web application.
5. **Flow of Events:**
   5.1. The Complaint System subsystem has its own use cases and it is explained in the previous sections.

1. **Name:** Lost & Found
2. **Participating Actor:** Losing party, Finder
3. **Entry Condition:** After the user receives an access token via login, the user goes to this page by clicking the Lost & Found button.
4. **Exit Condition:** The user moves from the navigation bar to another module, switches to the profile section or leaves the web application.
5. **Flow of Events:**
   5.1. The Lost & Found subsystem has its own use cases and it is explained in the previous sections.

1. **Name:** Borrow
2. **Participating Actor:** Borrower, Lender
3. **Entry Condition:** After the user receives an access token via login, the user goes to this page by clicking the Borrow button**.**
4. **Exit Condition:** The user moves from the navigation bar to another module, switches to profile section or leave from the web application.
5. **Flow of Events:**
   5.1. The Borrow subsystem has its own use cases and it is explained in the previous sections.

1. **Name:** Donation
2. **Participating Actor:** Donater, Donee
3. **Entry Condition:** After the user receives an access token via login, the user goes to this page by clicking the Donation button**.**
4. **Exit Condition:** User moves from Navbar to another module, switches to profile section or leaves from the web application.
5. **Flow of Events:**
   5.1. The donation module acts between the donation and the Donor. It has a product functionality similar to Second Hand sales and Borrow modules. It has basic features such as User View, Update, Upload, Post.

Second-Hand Sale Package

1. **Name**: Update Product
2. **Participating Actor**: Vendor
3. **Entry Condition**: Navigating to profile, selecting to the product and clicking to the edit button.
4. **Exit Condition**: Clicking the “Save” or “Cancel” button or navigating another page.
5. **Flow of Events**:
   5.1. Vendor selects the product they wish to update and clicks the edit button.
   5.2. Vendor modifies the necessary product details.
   5.3. Vendor saves the changes, and the system updates the product details.

1. **Name:** Upload a Product
2. **Participating Actor:** Vendor
3. **Entry Condition:** Clicking the add product button.
4. **Exit Condition:** Clicking the “Publish” or “Back” button
5. **Flow of Events:**
   5.1. Vendor decides to upload a new product.
   5.2. Vendor clicks the add product button.
   5.3. The vendor provides the necessary details such as product name, description, price, images, etc.
   5.4. The vendor clicks the publish button and the system lists the new product.

1. **Name:** Delete a Product
2. **Participating Actor:** Vendor
3. **Entry Condition**: Entering the product page and clicking the “Delete” button
4. **Exit Condition:** Navigation to another page or removing the product successfully.
5. **Flow of Events:**
   - **5.1.** Vendor decides to remove a listed product from the platform.
   - **5.2.** The vendor selects the product they wish to delete.
   - **5.3.** Vendor clicks the delete button.
   - **5.4.** Vendors confirm their intention to delete the product by clicking the confirm button.
   - **5.5.** The system removes the product from the platform.

1. **Name:** See Profile
2. **Participating Actor:** Vendor, Customer
3. **Entry Condition:** Clicking the profile image.
4. **Exit Condition:** Navigation to another page.
5. **Flow of Events:**
   - **5.1.** User (either vendor or customer) wants to view their profile.
   - **5.2.** The user clicks the profile image to view their profile.
   - **5.3.** The system displays the user's profile details.

1. **Name:** Communicate/Message
2. **Participating Actor:** Vendor, Customer
3. **Entry Condition:** Navigating to the messaging page.
4. **Exit Condition:** Navigating to another page or message is successfully sent.
5. **Flow of Events:**
   - **5.1.** User (either vendor or customer) wants to send a message to the other (either vendor or customer) user.
   - **5.2.** The user clicks to the send message button.
   - **5.3.** User is navigated to the message screen.
   - **5.4.** The user composes the message and sends it.
   - **5.5.** The system notifies the recipient of the new message.

1. **Name:** Add an Item to Favorites
2. **Participating Actor:** Customer
3. **Entry Condition:** Entering the product page and clicking the related button.
4. **Exit Condition:** The product is added to the customer's favorites.
5. **Flow of Events:**

5.1. Customer is interested in a product and wishes to save it for later.
5.2. Customer selects a product and enters the product page.
5.3. Customers click the save button to add the product to their favorites.
5.4. The system confirms the addition, and the product appears in the customer's favorites list.

1. **Name:** View All Products
2. **Participating Actor:** Customer
3. **Entry Condition:** Entering to the “Second-Hand” page.
4. **Exit Condition:** Navigating to another page.
5. **Flow of Events:**
   5.1. Customer enters the second-hand page.
   5.2. The system fetches the list of all products from the database.
   5.3. The system displays the products in a list or grid format, showing essential details like product name, image, and price.
   5.4. Customers can scroll through and navigate the list.

1. **Name:** View a Product
2. **Participating Actor:** Customer
3. **Entry Condition:** Clicking on a product.
4. **Exit Condition:** Navigating to another page.
5. **Flow of Events:**
   5.1. Customer selects a specific product from the list or search results.
   5.2. System fetches detailed information about the product from the database.
   5.3. The system displays the product details, including product name, description, images, price, and any other relevant information.
   5.4. Customers choose to take further actions such as adding to favorites, communicating with the vendor, or purchasing the product.

1. **Name:** Buy an Item
2. **Participating Actor:** Customer
3. **Entry Condition:** Sending a message to the vendor.
4. **Exit Condition:** Reaching an agreement with the vendor or not.
5. **Flow of Events:**
   5.1. Customer decides to purchase a product.
   5.2. Customer sends a message to the vendor to deal with.
   5.3. Vendor approves customer’s request.

1. **Name:** Search an Item
2. **Participating Actor:** Customer
3. **Entry Condition:** Searching for a product with a name or filter.
4. **Exit Condition:** Search results are displayed to the customer.
5. **Flow of Events:**
   - **5.1.** Customer inputs search criteria.
   - **5.2.** System searches the database for matching products.
   - **5.3.** Search results are displayed to the customer.

1. **Name:** No Such Item
2. **Participating Actor:** Customer
3. **Entry Condition:** Searching for a product.
4. **Exit Condition:** Searching for another product or displaying to the customer that the item is not found.
5. **Flow of Events:**
   - **5.1.** Customer enters search criteria for the product.
   - **5.2.** The system searches the database for the specified product or criteria.
   - **5.3.** If the product or matching criteria is not found in the database, the system triggers the "No Such Item" scenario.
   - **5.4.** The system displays a message to the customer to indicate that the product could not be found.
   - **5.5.** The system may suggest related products or prompt the customer to refine their search.

Donation Package

1. **Name:** View All Donations
2. **Participating Actor:** Donee
3. **Entry Condition:** Entering the “Donation” page.
4. **Exit Condition:** Navigating to another page.
5. **Flow of Events:**
   - **5.1.** Donee enters the donation page from the main page.
   - **5.2.** The system fetches the list of all donations from the database.
   - **5.3.** The system displays the donations in a list or grid format, showing essential details like product name, image, and price.
   - **5.4.** Donee can scroll through and navigate the list.

1. **Name:** View a Donation
2. **Participating Actor:** Donee
3. **Entry Condition:** Clicking on a specific donation.
4. **Exit Condition:** Navigating to another page.
5. **Flow of Events:**
   - **5.1.** Donee selects a specific donation from the list or search results.
   - **5.2.** System fetches detailed information about the donation.
   - **5.3.** The system displays the donation details, including the donation name, description, images, and any other relevant information.
   - **5.4.** Donee chooses to take further actions such as adding to favorites or communicating with the Donor.

1. **Name:** Search for a donation.
2. **Participating Actor:** Donee
3. **Entry Condition:** Searching for a donation with a name or filter.
4. **Exit Condition:** Navigating to another page or canceling the search process.
5. **Flow of Events:**
   - **5.1.** Donee inputs search criteria.
   - **5.2.** System searches for matching products with related filters.
   - **5.3.** Search results are displayed to the donee.

1. **Name:** Add an Item to Favorites
2. **Participating Actor:** Donee
3. **Entry Condition:** Entering the product page and clicking the related button.
4. **Exit Condition:** The product is added to the donee’s favorites.
5. **Flow of Events:**
   - **5.1.** Donee enters the product page.
   - **5.2.** Donee clicks the save button to add the product to their favorites.
   - **5.3.** The system confirms the addition, and the product appears in the donee’s favorites list.

1. **Name:** Communicate/Message
2. **Participating Actor:** Donor/Donee
3. **Entry Condition:** By clicking the related message button.
4. **Exit Condition:** Navigating to another page or canceling the procedure.
5. **Flow Of Events:**
   - **5.1.** Donor/Donee is directed to the related messaging page.
   - **5.2.** The system gathers the data of the previous messages, if there are any, from the system.

5.3. The system displays the messages, if there are any, in an appropriate messaging page.

5.4. Donor/Donee either sends another message or just checks previous messages, if there are any.

5.4.1. If Donor/Donee sends a new message then the system updates related segments.

1. **Name:** Update an Item
2. **Participating Actor:** Donor
3. **Entry Condition:** Clicking to the “update an item” button on the “Donation” page.
4. **Exit Condition:** Navigating to another page or canceling the procedure or publishing the updated version of the donation.
5. **Flow Of Events:**
   - **5.1.** Donor opens the intended donation target.
   - **5.2.** System gathers and displays the data of donation on the information page.
   - **5.3.** Donor modifies the properties of the donation by changing related fields. Then save the changes.

1. **Name:** Delete an Item
2. **Participating Actor:** Donor
3. **Entry Condition:** Selecting the intended donation.
4. **Exit Condition:** Navigating to another page or canceling the procedure or deleting the donation.
5. **Flow Of Events:**
   - **5.1.** Lender opens the intended loanable’s page.
   - **5.2.** System gathers and displays the donation’s on an information page.
   - **5.3.** Lender clicks the delete button to start the deleting procedure.

1. **Name:** Post an Item
2. **Participating Actor:** Donor
3. **Entry Condition:** Clicking to “Publish” on “Donation” page.
4. **Exit Condition:** Navigating to another page or cancelling the procedure or publishing the post.
5. **Flow Of Events:**
   - **5.1.** Donors fill up the required fields about donation such as name, image, product health, recommended course(optional), description, etc.
   - **5.2.** System creates the donation on a database and publishes the donation.

Borrow Package

1. **Name:** View All Items
2. **Participating Actor:** Borrower, Lender
3. **Entry Condition:** Entering the “Borrow” page.
4. **Exit Condition:** Navigating to another page.
5. **Flow Of Events:**
   - **5.1.** Borrower/Lender enters the “Borrow” page.
   - **5.2.** System gathers and displays all loans in a list or grid format, showing essential details .
   - **5.3.** System the loans like loanable name, image, anticipated borrowing period etc.
   - **5.4.** Borrower/Lender can scroll through and navigate the list.

1. **Name:** Borrow
2. **Participating Actor:** Borrower
3. **Entry Condition:** Selecting a specific loanable.
4. **Exit Condition:** Navigating to another page.
5. **Flow Of Events:**
   - **5.1.** Borrower selects and opens the intended loanable’s information page.
   - **5.2.** System gathers and displays the data of loans.
   - **5.3.** Borrower examines and sends a borrow request to the lender.
   - **5.4.** A notification is delivered to Lender’s account.

1. **Name:** Cancel Procedure
2. **Participating Actor:** Borrower, Lender
3. **Entry Condition:** Selecting a borrowed loanable request.
4. **Exit Condition:** Navigating to another page or canceling the current status (borrowed) of the loanable.
5. **Flow Of Events:**
   - **5.1.** Borrower/Lender opens a borrow request affiliated with its account.
   - **5.2.** System gathers and displays the data of loans.
   - **5.3.** Borrower/Lender clicks the cancel button to cancel the borrowing request.
   - **5.4.** A notification is delivered to Borrower’s/Lender’s account.

1. **Name:** Return
2. **Participating Actor:** Borrower
3. **Entry Condition:** By selecting the borrow request will be returned.

4. **Exit Condition:** Navigating to another page or canceling the procedure or returning the loanable.
5. **Flow Of Events:**
   - **5.1.** Borrower opens the intended borrow request affiliated with its account.
   - **5.2.** System gathers and displays the data of loans.
   - **5.3.** Borrower clicks the return button to return the loanable.
   - **5.4.** A notification is delivered to Borrower's/Lender's account.
     - **5.4.1.** Once Lender approves the return request, the system finalizes the borrowing request.

1. **Name:** Extension Request
2. **Participating Actor:** Borrower
3. **Entry Condition:** By selecting the borrow request wanted to be extended.
4. **Exit Condition:** Navigating to another page or canceling the procedure or requesting extension.
5. **Flow Of Events:**
   - **5.1.** Borrower opens the intended borrow request affiliated with its account.
   - **5.2.** System gathers and displays the data of loans on an information page.
   - **5.3.** Borrower first fills up the time period segment then clicks the request button to send the request.
   - **5.4.** A notification is delivered to Borrower's/Lender's account.
     - **5.4.1.** Depending on the Lender's decision, a response notification is sent to Borrower.

1. **Name:** Post an Item
2. **Participating Actor:** Lender
3. **Entry Condition:** Clicking the "post" button on the "Borrow" page.
4. **Exit Condition:** Navigating to another page or canceling the procedure or publishing the post.
5. **Flow Of Events:**
   - **5.1.** Lender fills up the required fields about loanable assets such as name, image, anticipated borrowing period, recommended course, description.
   - **5.2.** System publishes the loans.

1. **Name:** Update an Item
2. **Participating Actor:** Lender
3. **Entry Condition:** Selecting the intended loanable.
4. **Exit Condition:** Navigating to another page or canceling the procedure or updating the post.

5. **Flow Of Events:**
   - **5.1.** Lender opens the intended loanable's page.
   - **5.2.** System gathers and displays the data of loans on an information page.
   - **5.3.** Lender modifies the properties of the loanable by changing related fields. Then save the changes.
   - **5.4.** Once modifications are done, the system updates the properties of the loans.

1. **Name:** Rate Borrow
2. **Participating Actor:** Lender
3. **Entry Condition:** Selecting the intended borrowing request that was finalized.
4. **Exit Condition:** Navigating to another page or canceling the procedure or rating the Borrower.
5. **Flow Of Events:**
   - **5.1.** Lender selects and opens the intended borrowing request's information page.
   - **5.2.** System gathers and displays the data of the borrowing request on an information page.
   - **5.3.** Lender sets a rate from 1 to 5 and saves it.
   - **5.4.** A notification is delivered to Borrower's account.

1. **Name:** Communicate/Message
2. **Participating Actor:** Borrower**/**Lender
3. **Entry Condition:** By clicking the related message button.
4. **Exit Condition:** Navigating to another page or canceling the procedure.
5. **Flow Of Events:**
   - **5.1.** Borrower/Lender is directed to the related messaging page.
   - **5.2.** System gathers the data of the previous messages, if there are any, from the system.
   - **5.3.** The system displays the messages, if there are any, in an appropriate messaging page.
   - **5.4.** Borrower/Lender either sends another message or just checks previous messages, if there are any.
     - **5.4.1.** If Borrower/Lender sends a new message then the system updates related segments.

1. **Name:** Search an Item
2. **Participating Actor:** Borrower
3. **Entry Condition:** Searching for a product with a name or filter.

4. **Exit Condition:** Search results are displayed to the Borrower.
5. **Flow Of Events:**
   - **5.1.** Borrower inputs search criteria.
   - **5.2.** System searches the database for matching products.
   - **5.3.** Search results are displayed to the customer.

1. **Name:** Delete Item
2. **Participating Actor:** Borrower/Lender
3. **Entry Condition:** Selecting the intended loan.
4. **Exit Condition:** Navigating to another page or canceling the procedure or deleting the loan.
5. **Flow Of Events:**
   - **5.1.** Lender opens the intended loanable's page.
   - **5.2.** System gathers the loanable's data and displays the loanable's data on an information page.
   - **5.3.** Lender clicks the delete button to start the deleting procedure.

Complaint System Package

1. **Name:** Post a Complaint
2. **Participating Actor:** Sender, ComplaintChecker
3. **Entry Condition:** After logging in to the website, the user goes to the complaint module from the navigation bar and then uses the add product (complaint) button.
4. **Exit Condition:** The user moves from the navigation bar to another module, posts the complaint, cancels the post, or leaves the web application.
5. **Flow of Events:**
   - **5.1.** The user clicks the add button.
   - **5.2.** After clicking the Add button, the user selects the complaint category to create a complaint.
   - **5.3.** Creates a complaint by entering the specified properties.

1. **Name:** Delete a Complaint
2. **Participating Actor:** Sender
3. **Entry Condition:** After logging in to the website, the user goes to the Complaint module.
4. **Exit Condition:** The user navigates to another module, cancels deletion, or deletes the complaint.
5. **Flow of Events:**

5.1. After entering the user profile, the user goes to the Dashboard section.
5.2. Find your complaint in the complaint category in the Dashboard section.
5.3. Press the delete button for the complaint and delete the complaint.

1. **Name:** Check if the complaint is appropriate
2. **Participating Actor:** Complaint Checker
3. **Entry Condition:** Posting a complaint
4. **Exit Condition:** When the check is complete (either allows or denies)
5. **Flow of Events:**
   5.1. The user will create a complaint and try to post it.
   5.2. The ComplaintChecker will automatically check if the complaint is appropriate.
      5.2.1. If it is appropriate it will be posted.
      5.2.2. Otherwise, the ComplaintChecker will deny the post.

1. **Name:** View all complaints in order
2. **Participating Actor:** Reader
3. **Entry Condition:** After going to the Complaint System.
4. **Exit Condition:** User moves from the navigation bar to another module.
5. **Flow of Events:**
   5.1. After entering the complaint system, the user sees the complaints in the vote order.
   5.2. They can choose to upvote and downvote the complaint, which extends the view of all complaints.

1. **Name:** Vote up
2. **Participating Actor:** Reader
3. **Entry Condition:** After reviewing a complaint among the others, when the user clicks the upvote button.
4. **Exit Condition:** The user navigates to another module.
5. **Flow of Events:**
   5.1. When a user sees a complaint that they agree with, they can click on the button to upvote the complaint.
   5.2. If a complaint has more upvotes it will become more visible in terms of ordering the complaints.

1. **Name:** Vote down
2. **Participating Actor:** Reader
3. **Entry Condition:** After reviewing a complaint among the others, when the user clicks the downvote button.

4. **Exit Condition:** The user navigates to another module.
5. **Flow of Events:**
   5.1. When a user sees a complaint that they don't agree with, they can click to the button to downvote the complaint.
      5.1.1. If a complaint has more downvotes it will become more invisible in terms of ordering the complaints.

Lost & Found Package

1. **Name:** Search an Item
2. **Participating Actor:** Losing party
3. **Entry Condition:** After entering the "Lost & Found" page, clicking to the search box.
4. **Exit Condition:** Click an item or navigate to another page.
5. **Flow Of Events:**
   5.1. Losing party enters the "Lost & Found" page.
   5.2. Losing party searches for an item that they lost.
   5.3. Filtering mechanism shows the matching notices with keywords / tags.
   5.4. Losing party can scroll through and navigate the list.

1. **Name:** View a Notice
2. **Participating Actor:** Losing party
3. **Entry Condition:** Clicking a notice while searching.
4. **Exit Condition:** Exiting the view mode or navigating to the message section with the publisher of the notice or navigating to another page.
5. **Flow Of Events:**
   5.1. Losing party clicks on a specific notice.
   5.2. Users can see the publisher and can text them.

1. **Name:** List all lost items
2. **Participating Actor:** Losing party
3. **Entry Condition:** Entering the "Lost & Found" page.
4. **Exit Condition:** Navigating through another page.
5. **Flow Of Events:**
   5.1. When the user enters the lost & found page, the main screen will show all notices.
   5.2. Users can scroll down and see the list of all lost items.
   5.3. Or, they can search for a specific notice from the search section.

1. **Name:** Post a lost notice
2. **Participating Actor:** Losing party and Finder
3. **Entry Condition:** After entering the “Lost & Found” page, when they click the “Create” button.
4. **Exit Condition:** Navigating through another page, posting the notice or canceling the process by clicking corresponding buttons.
5. **Flow Of Events:**
   - **5.1.** When the user enters the lost & found page, they will see a button on there called something similar, “create a new lost notice.”
   - **5.2.** They can click that button to create their lost post.

1. **Name:** Describe the item
2. **Participating Actor:** Losing party
3. **Entry Condition:** When they click the “create” button.
4. **Exit Condition:** Canceling the process or posting a notice.
5. **Flow Of Events:**
   - **5.1.** Inside the create page, they will see a text field that they can describe the item, time, place, etc.
   - **5.2.** They can post the notice or cancel the process and go back to the main page.

1. **Name:** Communicate/Message
2. **Participating Actor:** Losing party / Finder
3. **Entry Condition:** By clicking the related message button.
4. **Exit Condition:** Navigating to another page or canceling the procedure.
5. **Flow Of Events:**
   - **5.1.** Borrower/Lender is directed to the related messaging page.
   - **5.2.** System gathers the data of the previous messages, if there are any, from the system.
   - **5.3.** The system displays the messages, if there are any, in an appropriate messaging page.
   - **5.4.** Borrower/Lender either sends another message or just checks previous messages, if there are any.
     - **5.4.1.** If Borrower/Lender sends a new message then the system updates related segments.

1. **Name:** Send the picture of the item
2. **Participating Actor:** Losing party / Finder
3. **Entry Condition:** When they click to the share image part on the communication section.
4. **Exit Condition:** Navigating through another page or sending the message or canceling the process by clicking the corresponding buttons.
5. **Flow Of Events:**
   - **5.1.** When the losing party finds a notice about their lost product, they communicate with the finder.
   - **5.2.** After gathering enough information about the item from each other, both sides can share a photo of the item. For example, the losing party can prove that the item is theirs by sharing a previous photo of the item.

1. **Name:** Delete a post
2. **Participating Actor:** Losing party and Finder
3. **Entry Condition:** Entering the “Lost & Found” page.
4. **Exit Condition:** Navigating through another page
5. **Flow Of Events:**
   - **5.1.** When the lost item is found, they both can delete their lost notice.

1. **Name:** Update a post
2. **Participating Actor:** Losing party and Finder
3. **Entry Condition:** entering the “Lost & Found” page.
4. **Exit Condition:** Navigating through another page
5. **Flow Of Events:**
   - **5.1.** When something has changed about the notice or is miswritten, they can update their lost posts.