

Baichday
SPROJ Report



Moiz Nafey 23100157

Silal Anwar 23100334

Muhammad Arslan Ullah Tarrar 23100225

Mahad Mubashir Beg 23100089

Nashit Iftikhar 23100181

Advisor: Waqar Ahmad
School of Science and Engineering
Lahore University of Management Sciences
Submission Date: 29/04/2023

Acknowledgement and Dedication

We would like to express our sincere gratitude to our instructor, Waqar Ahmad, for his invaluable support, guidance, and encouragement throughout the project. His expertise and insights have been crucial in helping us develop a deeper understanding of the subject matter and achieve our goals. We are also grateful for his willingness to dedicate his time and energy to our project, and for his unwavering belief in our abilities.

We dedicate this project to Lahore University of Management Sciences (LUMS), which has provided us with the education and skills to pursue our academic and professional goals. LUMS has been a home to thousands of students before us and will continue to be a beacon of knowledge and innovation for generations to come. We are proud to be a part of this institution, and we hope that our project contributes to its legacy of excellence.

Certificate

I certify that the senior project titled “**Baichday**” was completed under my supervision by the following students:

and the project deliverables meet the requirements of the program.

Date:

Advisor (Signature)

Date:

Co-advisor (if any)

Table of Contents

Table of Contents	Error! No bookmark name given.
List of Figures.....	Error! No bookmark name given.
1. Introduction.....	Error! No bookmark name given.
a. Introduction	Error! No bookmark name given.
b. Objective and Scope	Error! No bookmark name given.
c. Development Methodology.....	Error! No bookmark name given.
d. Contributions.....	Error! No bookmark name given.
2. System Requirements	Error! No bookmark name given.
a. System Actors	Error! No bookmark name given.
b. Functional Requirements	Error! No bookmark name given.
c. Non-functional Requirements	Error! No bookmark name given.
3. System Architecture.....	Error! No bookmark name given.
a. Architecture Diagram.....	19
b. Architecture Description	21
c. Justification of the Architecture	21
d. Tools and Technologies	Error! No bookmark name given.
4. Requirements Specifications	Error! No bookmark name given.
a. Use Cases	Error! No bookmark name given.
b. Class Diagram	50
c. Sequence Diagrams	52
5. Software Development Methodology and Plan	Error! No bookmark name given.
a. Software Process Selection	62
b. Gantt Chart.....	66
6. Database Design and Web Services.....	Error! No bookmark name given.
a. Database Design.....	Error! No bookmark name given.
b. API Specification	Error! No bookmark name given.
7. System User Interface.....	Error! No bookmark name given.
8. Project Security	Error! No bookmark name given.
a. Project Threats	Error! No bookmark name given.
b. Potential Losses	Error! No bookmark name given.
c. Security Controls.....	Error! No bookmark name given.
d. Static and Dynamic Security Scanning Tools.....	Error! No bookmark name given.
9. Risk Management	Error! No bookmark name given.
Potential Risks and Mitigation Strategies	86
10. Testing and Evaluation.....	Error! No bookmark name given.
11. Deployment Guidelines.....	Error! No bookmark name given.

12. Conclusion	Error! No bookmark name given.
a. Summary	Error! No bookmark name given.
b. Challenges	Error! No bookmark name given.
c. Future	Error! No bookmark name given.
13. Review checklist	102
14. References	Error! No bookmark name given.

List of Figures

Figure 1: Architecture Diagram	20
Figure 2: Use Case Diagram	25
Figure 3: Common Cases	26
Figure 4: Individual Use Cases	27
Figure 5: Class Diagram	48
Figure 6: Sequence Diagram - Signup	49
Figure 7: Sequence Diagram – Login	50
Figure 8: Sequence Diagram - View Home Page.....	51
Figure 9: Sequence Diagram - View Product Description	52
Figure 10: Sequence Diagram - Place bid	53
Figure 11: Sequence Diagram - View Bids	54
Figure 12: Sequence Diagram - Upload Product.....	55
Figure 13: Sequence Diagram - View Profile.....	56
Figure 14: Sequence Diagram - Edit Profile.....	57
Figure 15: Sequence Diagram - Logout.....	58
Figure 16: Figure 16: E/R Diagram	65
Figure 17: Homepage.....	66
Figure 18: Sign In	67
Figure 19: Homepage Signed In	67
Figure 20: My Wallet	68

Figure 21: Add Credit	68
Figure 22: View Product	69
Figure 23: Homepage.....	70
Figure 24: Item History.....	70
Figure 25: Upload Product	71
Figure 26: Item History	72
Figure 27: Footer	72
Figure 28: Feedback	73
Figure 29: Chatbot	73
Figure 30: My Profile	74

1. Introduction

a. Introduction

Baichday is a web application built on 3 tier architecture. It is the auction based platform where users can bid on the products or upload products on the platform to be sold. Through our application, users can bid on a product or sell a product from the same account unlike some other platforms where users have to register a separate account for buying or selling. The initial target users of our application are those users who have limited edition products available or other valuable products that are unique and not readily available in the market for instance a painting, limited edition smartphones, etc.

b. Objective and Scope

The main motivation behind building this auction platform was that currently, there are no such platforms based in Pakistan that are widely known or used. Our main objective is to make people aware about the reliability and advantages of using our platform. Our web application works in real time where the time of each auctioned product is tracked. This will ensure a smooth process and a wonderful experience for the users.

In normal situations, when a user tries to sell their product, they usually meet the customers who try to lower the rate and the negotiations could get pretty intense resulting in loss of either party. Our application will solve this issue by allowing sellers to set the rate of their own choice. If a buyer can afford it, they will bid on the product, otherwise they will not.

Lastly, our platform eliminates the hassle of a middleman since everything is automated and very less fee is charged on each transaction. This methodology is to facilitate our users who can buy products without hassles at relatively lower costs.

c. Development Methodology

Baichday is a web application building using MERN stack. The architecture has 3 layers: Front-End, Back-End, Database. The Front-End was made using Reactjs (a Javascript library). Back-End was implemented with Nodejs and Expressjs. The Database we used was MongoDB.

On Front-End, we divided the pages into components in separate files to be rendered. Similarly, Back-End was divided into multiple folders: controllers (it contains the logic to be implemented when a certain API call is made), routes (contains the defined routes for API calls), and models (it contains the schema of Database).

This Development technique was really helpful since the base code became modular and it was relatively quite easier to implement changes, debugging and testing.

d. Contributions

Our application handles data in real time. This means that it keeps track of time remaining for each auctioned product and automatically decides its future after the time remaining is 0.

Similarly, the transaction handling is also done in real time where if a product's time remaining becomes 0, the money from the highest bidder's wallet is deducted and 95% of it goes to the seller's wallet while the 5% goes to admin's wallet.

Our platform also allows real time chat between users if a transaction between them is successful. The purpose of the chat feature is to allow both parties to communicate and confirm regarding the product details.

2. System Requirements

The system requirements chapter outlines the features and specifications necessary for the system to meet its objectives. We go over the primary actors of our system and then go over the functional as well as non-functional requirements of our project.

a. System Actors

Actor Name	Description
Admin	Admin is the auction marketplace owner, a middleman entity ensuring smooth auction process and verification of products. Admin will provide customer support and track the progress of profits, bids and transactions.
Seller	The seller will be able to upload products on this platform and wait for people to bid on them.
Bidder	The bidder will be able to search for products that are auctioned and bid on them.

b. Functional Requirements

Requirements	
Sr#	Requirement
1	As a bidder, I want to sign up/create an account on the platform.
2	As a bidder, I want to login to the platform.
3	As a bidder, I want to search for the products present on the platform.
4	As a bidder, I want to bid on the products.
5	As a bidder, I want to view the information (bidding status, starting bid, highest bid, image) of a product.
6	As a bidder, I want to view products that I have currently bid on.
7	As a bidder, I want to view all products that I have bid on.

8	As a bidder, I want to check the balance of my wallet.
9	As a bidder, I want to contact customer support.
10	As a bidder, I want to add balance to my wallet.
11	As a bidder, I want to chat with seller..
12	As a bidder, I want to change my name.
13	As a bidder, I want to change my email address.
14	As a bidder, I want to change my password.
15	As a bidder, I want to change my contact info.
16	As a bidder, I want to change my profile image.
17	As a bidder, I want to delete/deactivate my account.

18	As a bidder, I want to logout of my account.
19	As an admin, I want to login to the platform.
20	As an admin, I want to verify products that are to be sold.
21	As an admin, I want to track the profit made.
22	As an admin, I want to view bids on a product.
23	As an admin, I want to add products to featured items.
24	As an admin, I want to view ratings of users.
25	As an admin, I want to ban malicious users.
26	As an admin, I want to logout of my account.
27	As a seller, I want to signup/create an account on the platform.

28	As a seller, I want to login to the platform.
29	As a seller, I want to add a product to the platform to be sold.
30	As a seller, I want to add a minimum bid for my product.
31	As a seller, I want to view the bids on my product.
32	As a seller, I want to view the products I have sold.
33	As a seller, I want to check balance of my wallet.
34	As a seller, I want to view other products on the platform.
35	As a seller, I want to view my chats.
36	As a seller, I want to chat with the bidder after a successful transaction.
37	As a seller, I want to change my name.

38	As a seller, I want to change my email address.
39	As a seller, I want to change my password.
40	As a seller, I want to change my contact info.
41	As a seller, I want to change my profile image.
42	As a seller, I want to delete/deactivate my account.
43	As a seller, i want to logout of my account.

c. Non-functional Requirements

Sr#	Requirements
1	The system should not utilize more than 1 GB of memory at any time during its execution.
2	The system should not fail more than three times every 24 hours. In case of a failure, the system should restore to normal operations within 5 minutes of failure.
3	The system should be secure and should support payment gateways. All passwords need to be encrypted and stored in a database.
4	The webpages should have a minimum delay and load within 2-3 seconds.
5	Auction process should be fast and simple. Customers should be informed of the latest status of their bid, such as "Your bid has been placed," "Your item is up for auction," etc.
6	The website will conduct checks to ensure that all data entered into the system is correct. This information includes user phone numbers, email addresses, and a

	<p>backup phone number if they have one. Furthermore, if an item is already auctioned, the system will ensure that no further bids are placed on it.</p>
7	<p>MongoDB database should not timeout and should be updated within 2-3 seconds.</p>
8	<p>Website should be highly responsive and behave in accordance with user actions. The website should be able to easily modify the items to whatever the admin wishes them to be.</p>
9	<p>Website should be consistent across all web pages.</p>
10	<p>The website should have a simple layout to make it easy for the user to navigate without training. This will be accomplished by including toolbars, a home page, and other widgets that will direct the user</p>
11	<p>The back-end and front-end of the website shall be synchronized at all times and provide the users with real-time information.</p>
12	<p>The website shall be SEO compliant to catch traffic online.</p>
13	<p>The server will be up and running at all times, allowing customers to access all website functions within 3 seconds of submitting a request.</p>

14	Any device with a web browser and a reliable internet connection should be able to operate the system.
15	Maintenance of the system should be cost-effective, and any faults that appear should be easily repairable. We'll check the website to make sure that it is up and running correctly on commonly used web browsers
16	Daily backups of the entire website, including the databases, will be made so that if the site collapses, we'll have a backup.
17	Selenium will be used to test the website because the majority of our use cases are highly testable, we will use component testing before launching. Our Selenium script will automatically test all functionalities with acceptable and unacceptable cases. Moreover, we will conduct our own separate tests to get user input.

3. System Architecture

The system architecture chapter outlines the architectural design of the software system, including the patterns and technologies used for its development and deployment. The chapter explains the two main architectural patterns adopted in the system, which are the client-server and model-view-controller patterns. The chapter further highlights the benefits and limitations of the chosen architecture and tools, including the development, testing, and deployment technologies.

a. Architecture Diagram

Draw a diagram of the system architecture.

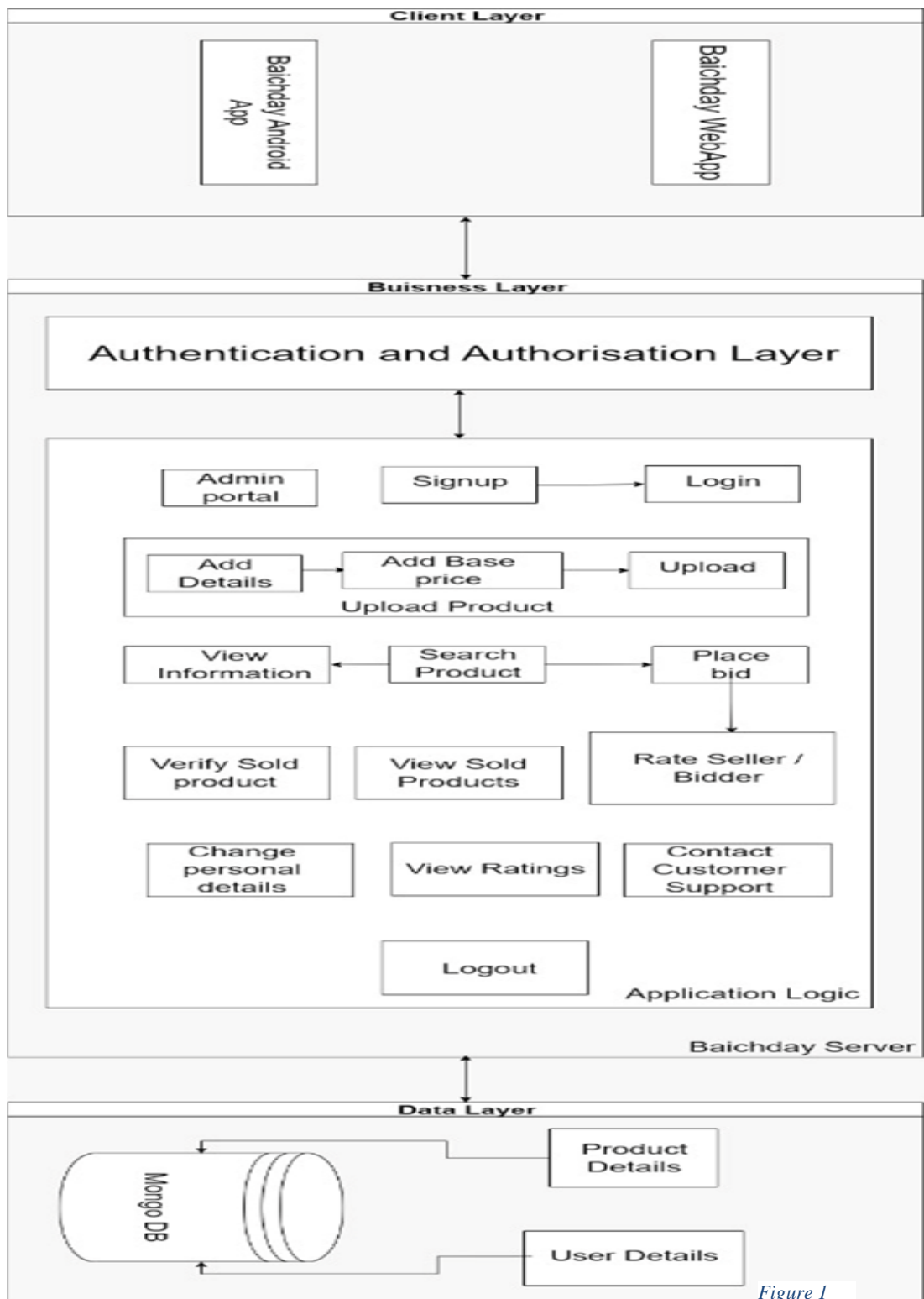


Figure 1

Figure 2

b. Architecture Description

Our system is based on two architectural patterns which are the client-server and the model-view-controller.

The client server pattern is a very famous architectural model which is often used for web applications and websites. There are two types of client server models; thin client (logic heavy client) and thick client (logic heavy server). Our software will rely on the thin client server architectural model. This model in our system symbolizes the divide between the front end and backend of our software. The frontend client will be light in terms of logic and will defer to the backend server in terms of computations. On the other hand, the backend will be responsible for most of the heavyweight computations. The backend will connect to the database and retrieve data for the frontend to display.

The frontend on the other hand has the architecture of the Model-View-Controller which is a three-tier architectural pattern. The Model-View-Controller divides the program logic into three interconnected components that are responsible for handling their own responsibilities. Our frontend in this pattern is divided into three portions namely, the model, the view and the controller. The View is the components and the screens that will be shown to the user while the controller will be the segment in contact with the backend. It will receive the data and pass it on to the model for processing. The model will convert it into usable forms for the components of the view to render.

c. Justification of the Architecture

Pros of using Client Server Architecture and Model-View-Controller:

1. Firstly, the centralized network has complete authority over all the processes which is very necessary in our case since we are the trusted third party between the seller and the bidder.
2. The users can access the application at any time.
3. It provides a good user interface and makes it easier to navigate and find files.
4. Resources can be easily shared across various platforms.
5. The reason for using MVC is that it makes it easier to navigate between different views.
6. It allows for a faster development process since different developers can work on different views at the same time.
7. Modification in one view does not affect the whole front end.
8. MVC is a good choice for web applications as view and controller are naturally separated.

Cons of using Client Server Architecture and Model-View-Controller:

1. If the sever goes down for any reason the whole architecture is affected.
2. It has high operation cost as the view could be overburdened with update requests if the model keeps on going with frequent changes.
3. Views like graphic displays can take a longer time to make.
4. Traffic congestion can take place if there are too many users at one time which will slow the whole network.
5. Duplicated code is used between common views.

d. Tools and Technologies

Development:

- Coding Platform – VS Code (version 1.72.2)
- Front-End – Reactjs (version 18.2.0)
- Back-end – Nodejs and Expressjs (version 18.10.0)
- Database – MongoDB (version 6.0)
- API Testing – Postman (version 9.31.0)
- System Testing – Cypress (version 12.9.0)
- Version Control Git – GitHub (version 2.9.3)
- Browser for development tool and testing – Google Chrome (version 106.0.5249.103)

Deployment:

- Front-End – Netlify (version 12.0.8)
- Back-End – Heroku (version 7.60.1)
- Database – MongoDB Atlas (version 6.0)

4. Requirements Specifications

The Requirements Specifications chapter outlines the use cases and user scenarios for a platform that facilitates bidding and auctioning of products. The use cases are categorized based on the roles of the users, which include bidders and sellers. The chapter also includes descriptions of the steps involved in each use case and any alternate or exception paths that may occur. Additionally, the chapter presents a class diagram and sequence diagrams to illustrate the system's functionality.

a. Use Cases

1. Common use cases for all users:

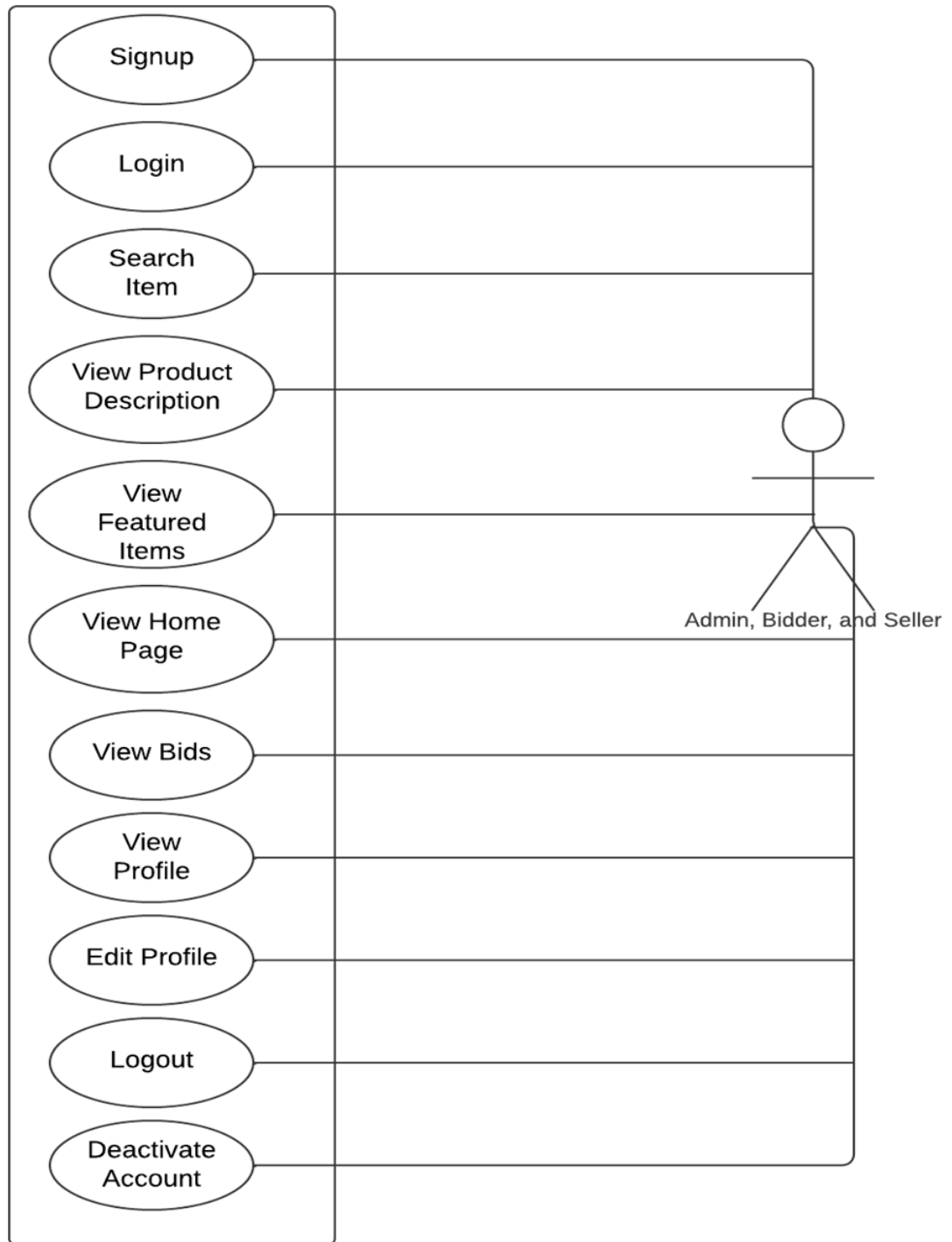


Figure 2: Use Case Diagram

2. Common use cases for bidder and seller:

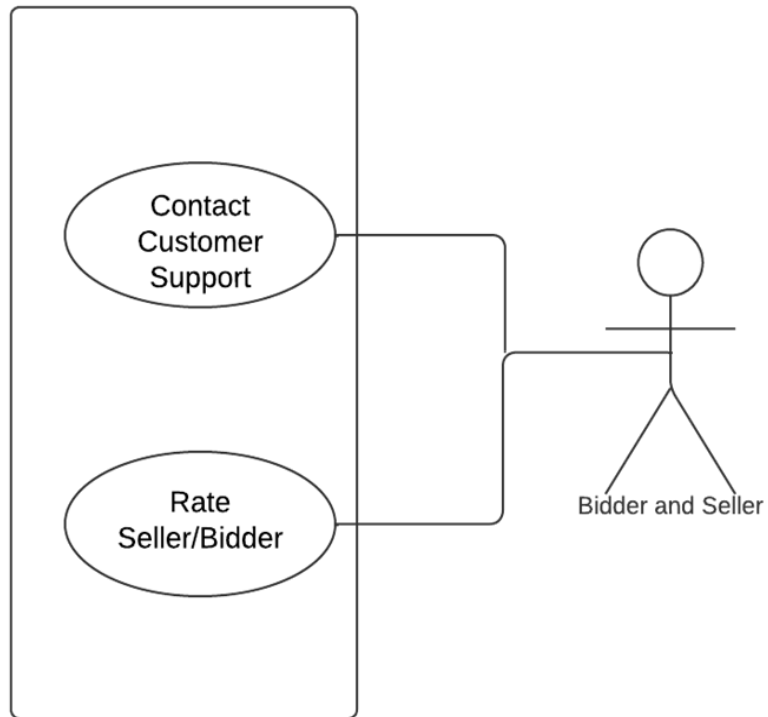


Figure 3: Common Cases

3. Individual use cases for users:

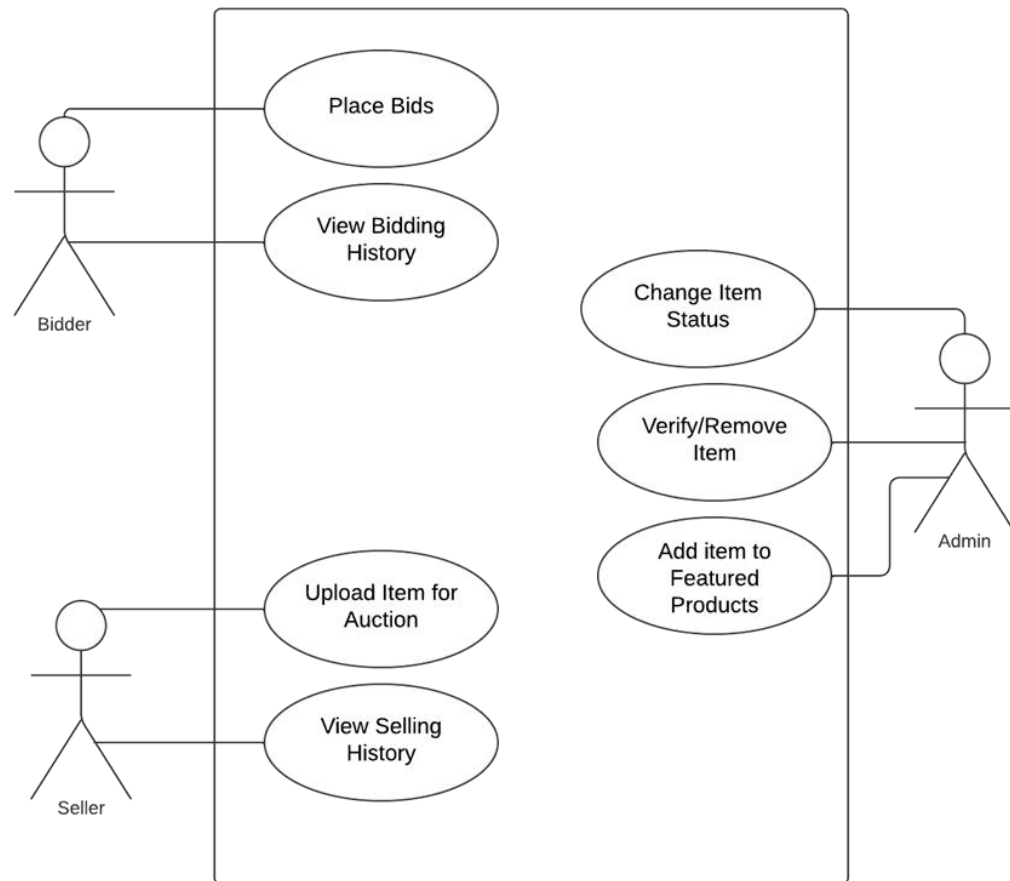


Figure 4: Individual Use Cases

Use Case Descriptions:

1. Sign up

Identifier	UC-001
Purpose	Allows the bidder/customer to sign up on the platform
Pre-conditions	User had a valid email address and phone number
Post-conditions	User account is created on the platform
Step #	Typical Course of Action
1.	The bidder/seller visits the website and selects the option to sign up on the platform.
2.	The bidder/seller is prompted to enter his name, age, location, email and phone number
3.	The bidder/seller confirms to sign up.

4.	If the email and phone number are valid and have not been already used to sign up, they are sent a verification code on their email and phone number
5.	The bidder/seller is prompted to enter the verification code they received on their phone number and email address to complete the signup
6.	If the verification code is correct, the bidder/seller will be directed to create a password for their account.
7.	Once they set a password, they are directed to the login page.
8.	The use case ends.
Step #	Alternate Courses of Action
	None
Step #	Exception Paths
1.	In step 4, if the email or phone number are not valid or are already in use, an error message is displayed, and they are prompted to re-enter a valid email address/phone number

2.	In step 5, if the bidder/seller enters an incorrect verification code, an error message is displayed, and they are prompted to enter the correct verification code again

2. Login

Identifier		UC-002
Purpose		Allows the customer to login onto our platform
Pre-conditions		The <i>signup</i> use case has been successfully completed
Post-conditions		The user can have access to the features available on our platform
Step #	Typical Course of Action	
1.	The bidder/seller opens the sign in page.	
2.	The bidder/seller is prompted to enter their email and password	
3.	If the email and password is correct, they are allowed access to the platform and use its features	
4.	The use case ends	

Step #	Alternate Courses of Action
1.	none
Step #	Exception Paths
1.	In step 3, if an incorrect password is entered, an error message is displayed and the user is prompted to re-enter their correct password or allows them to avail to option of resetting their password by clicking “Forgot Password” option
2.	In step 3, if a bidder/seller tries to sign in with an email that has not been registered on our platform, they are given an error message and are prompted to enter a correct email or sign up with the current email

3. View Home Page

Identifier		UC-003
Purpose		Allows the bidder/seller to view the platforms home page
Pre-conditions		The <i>SignIn</i> use case has been fulfilled
Post-conditions		The bidder/seller is able to view the information on the platform's homepage
Step #	Typical Course of Action	
1.	The bidder/seller clicks on the home tab at the navigation bar on the top of the website	
2.	The bidder/seller is then directed to the platform's homepage	
3.	The bidder/seller is then able to see the information available on the homepage of the platform	

4.	Use case ends.
Step #	Alternate Courses of Action
1.	In step 1, the bidder/seller can click on the icon of the platform as well instead of clicking on the home tab. They will then be directed to the homepage as well
Step #	Exception Paths
1.	In step 8, if there are not sufficient funds, then an error message is displayed and execution proceeds to step 9.

4. View Product Description

Identifier		UC-004
Purpose		The bidders/sellers can view the details of a product
Pre-conditions		The product must have been added successfully by the seller
Post-conditions		The bidder either enters the bidding or returns back to home page.
Step #	Typical Course of Action	
1.	The customer selects the product in question from homepage/search page.	
2.	The customer can view the product page containing details of product	
3.	End of use case	
Step #	Alternate Courses of Action	

1.	In step 6, the customer can then go directly back to home page.
Step #	Exception Paths
1.	None

5. Place Bid on Product

Identifier		UC-005
Purpose		Allows the bidder to place a bid on an item
Pre-conditions		The bidder must be signed in and the product must have been published successfully
Post-conditions		A bid is placed on the product from your account
Step #	Typical Course of Action	
1.	The bidder opens the product description and is able to see details regarding the product	
2.	The bidder then clicks on the “Place bid” button	
3.	The bidder is shown the terms and conditions to agree to before placing the bid	
4.	Once the bidder clicks on “Agree”, a pop up will be displayed	

5.	The pop up will prompt the bidder to enter the price they are willing to offer. Once completed, they click on “Done”
6.	A message pops up that displays the name and description of the product, The bidding amount placed by the bidder and the total number of bids that have been placed on this product to make sure the bidder can have a final review before placing the bid
7.	The bidder reviews all the information and clicks on “Confirm” to place his bid
8.	After the bid is placed, the user is redirected to the product page where he can see his bid as placed
9.	The use case ends.
Step #	Alternate Courses of Action
1.	none

Step #	Exception Paths
1.	In step 7, if the user does not have the required funds in his account, an error message will be displayed, and the user will be prompted to deposit funds in their account in order to place the bid

6. View Bids

Identifier	UC-006
Purpose	Will allow the bidder to view all the bids he has made so far
Pre-conditions	The <i>SignIn</i> use case has been completed successfully
Post-conditions	The bidder is able to view the history of all the bids he has placed
Step #	Typical Course of Action
1.	The bidder clicks on the account icon which will display a drop-down menu
2.	From the drop down menu, the bidder will select “Past Bids”

3.	The bidder will be redirected to the “Past Bids” page
4.	The bidder will be able to view the history of all the previous bids he has placed including the bids that you have won, lost or cancelled.
5.	The use case ends.
Step #	Alternate Courses of Action
1.	In step 2, bidder can select to view their profile and then proceed to select the “Past Bids” tab.

Step #	Exception Paths
1.	none

7. Upload Product for Auction

Identifier	UC-007
Purpose	Allows the seller to upload an item to the platform for auctioning
Pre-conditions	The <i>SignIn</i> use case has been successfully completed
Post-conditions	The product is successfully posted on the platform and is up for auction
Step #	Typical Course of Action
1.	The seller clicks on the “Upload Product” tab on the nav bar
2.	The seller is redirected to a page requiring the product information

3.	The seller is prompted to enter the name of the product, clear pictures of the product, the description, date of purchase, model number if applicable, company name if applicable, duration of bidding and the starting bidding amount
4.	Once completed, the user clicks “Done”
5.	A pop up will appear that will display the details entered by the seller to ensure they did not make a mistake
6.	Once reviewed, the seller clicks “Confirm”
7.	A pop up appears if the product has been successfully posted
8.	The seller is then directed to the product description page of their newly uploaded product
9.	The use case ends.

Step #	Alternate Courses of Action
1.	none
Step #	Exception Paths
1.	In step 7, because of network issues, if the product is not posted successfully, they are given an error message and are prompted to reupload the item.

8. View Profile

Identifier	UC-008
Purpose	Allows the bidder/seller to view their profile
Pre-conditions	The <i>sign in</i> use case is completed successfully.
Post-conditions	The bidder/seller will be able to view their profile details successfully

Step #	Typical Course of Action
1.	The bidder/seller clicks on account icon on the navigation bar
2.	A drop down menu will be displayed. bidder/seller will select “View Account” option from the drop down menu
3.	bidder/seller will be directed to the Account Details page
4.	bidder/seller will be able to see their account name, registered email, registered phone number, profile picture, chosen language and location
5.	The use case ends.

Step #	Alternate Courses of Action
1.	none
Step #	Exception Paths
1.	none

9. Edit Profile

Identifier	UC-009
------------	--------

Purpose	Allows the bidder/seller to edit their profile information
Pre-conditions	The <i>sign in</i> use case is completed successfully.
Post-conditions	The profile details of the bidder/seller are changed successfully
Step #	Typical Course of Action
1.	The bidder/seller clicks on account icon on the navigation bar
2.	A drop-down menu will be displayed. The bidder/seller will select “View Account” option from the drop-down menu
3.	The bidder/seller will be directed to the Account Details page
4.	The bidder/seller clicks on the “Edit Information” button
5.	The bidder/seller will be directed to a page where they will be able to edit their information such as name, email, password, profile picture, location
6.	Once the bidder/seller clicks “Done”, a pop message will be displayed

7.	The message will show the changes made by the user and ask for a confirmation
8.	The bidder/seller will click on “Confirm”
9.	The changes will be saved and the bidder/seller will be redirected to the Account Details page, reflecting the changes made
Step #	Alternate Courses of Action
1.	none
Step #	Exception Paths
1.	In step 9, because of a network failure the changes might not be saved
2.	An error message will pop up informing the bidder/seller about this

3.	The message will prompt the user to make the required changes again and redirect them to Account Details page. The course of action will continue from step 3,
----	--

10. Logout

Identifier	UC-010
Purpose	The user can log out of their account
Pre-conditions	The user must be logged in successfully
Post-conditions	The user is redirected to the landing page and must sign in again for features which require validation
Step #	Typical Course of Action
1.	The user clicks on sign out button in dropdown menu in navbar.
2.	The user is redirected to the home page
3.	End of Use Case.

Step #	Alternate Courses of Action
1.	None
Step #	Exception Paths
1.	In step 1, if the user is not logged in, the logout button will not accessible

b. Class Diagram

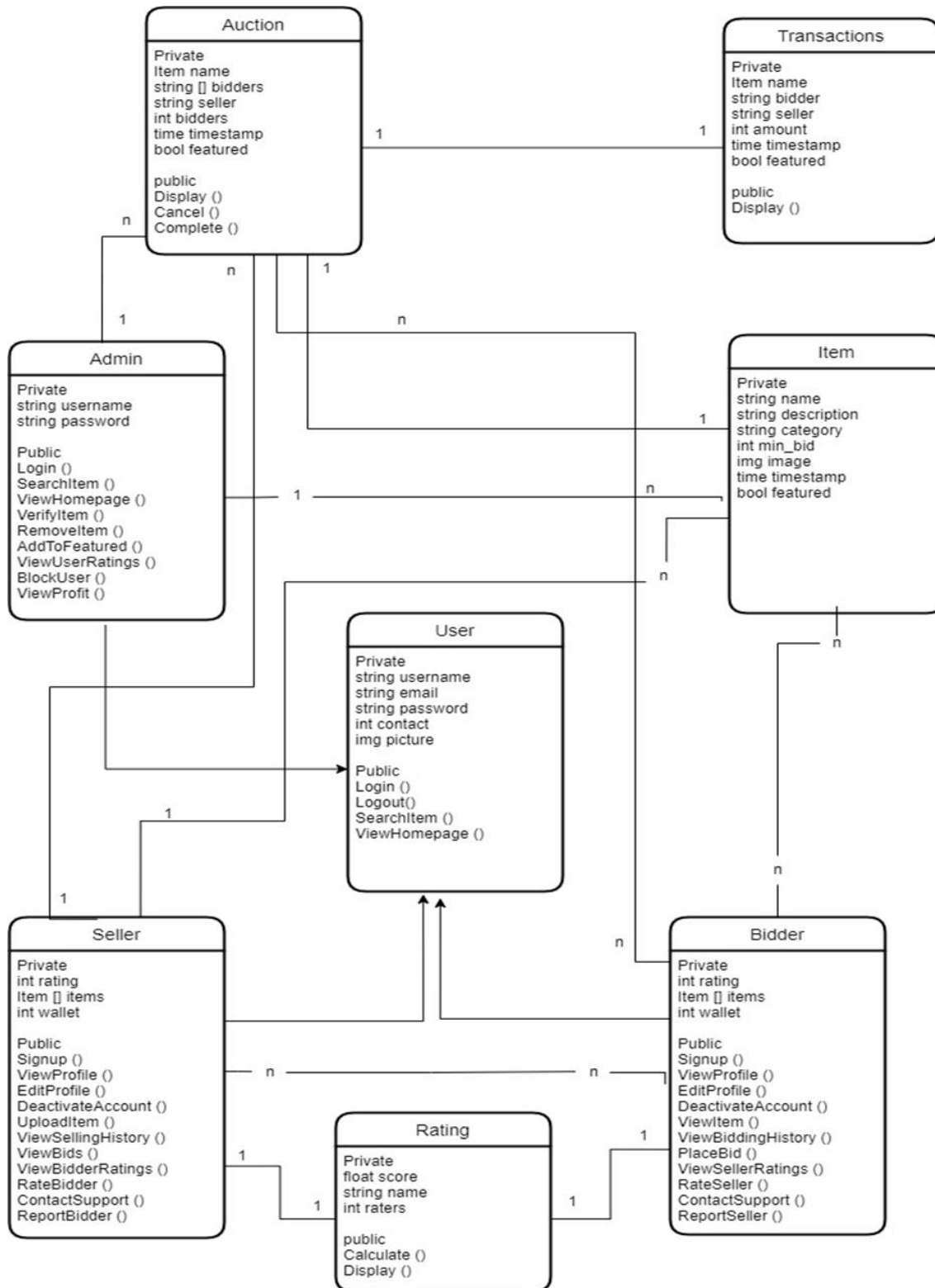


Figure 5:Class Diagram

c. Sequence Diagrams

1. Signup

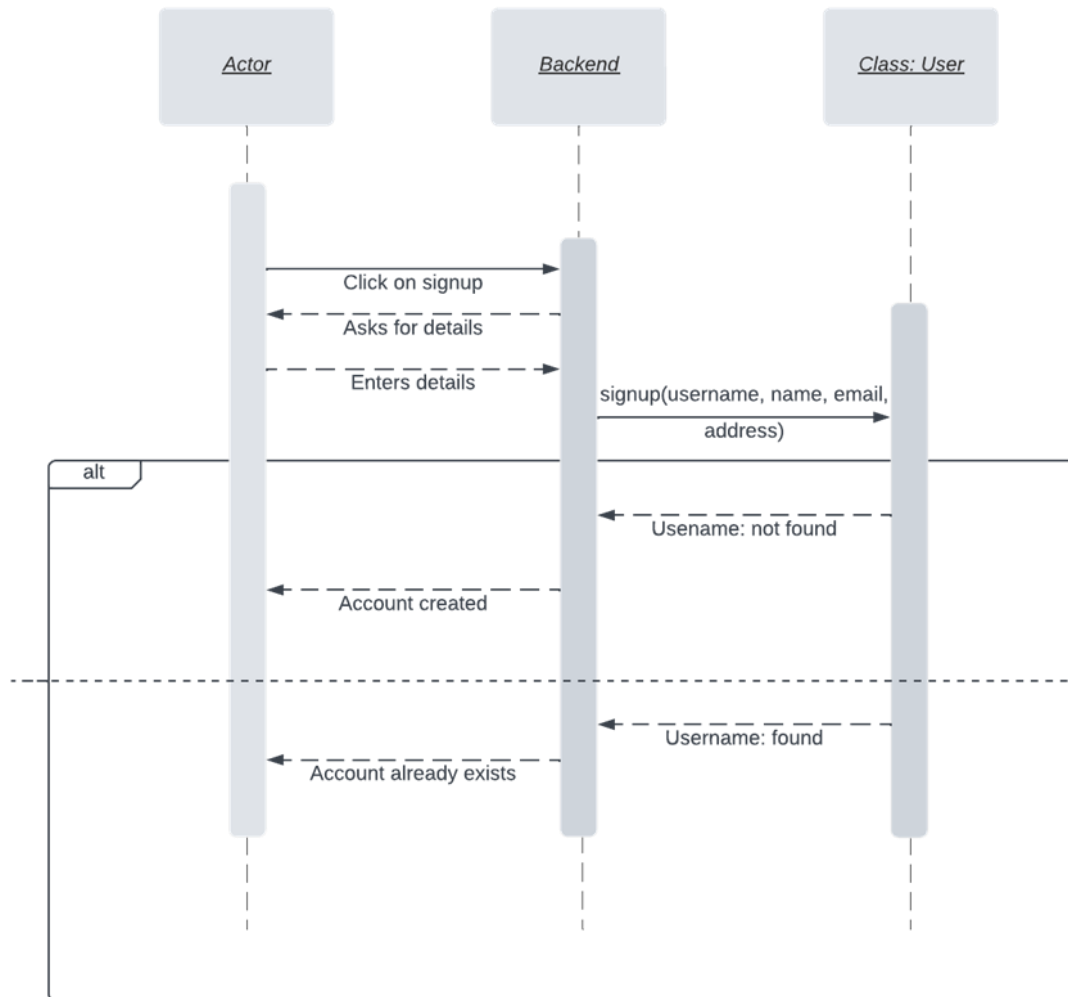


Figure 6: Sequence Diagram – Signup

2. Login

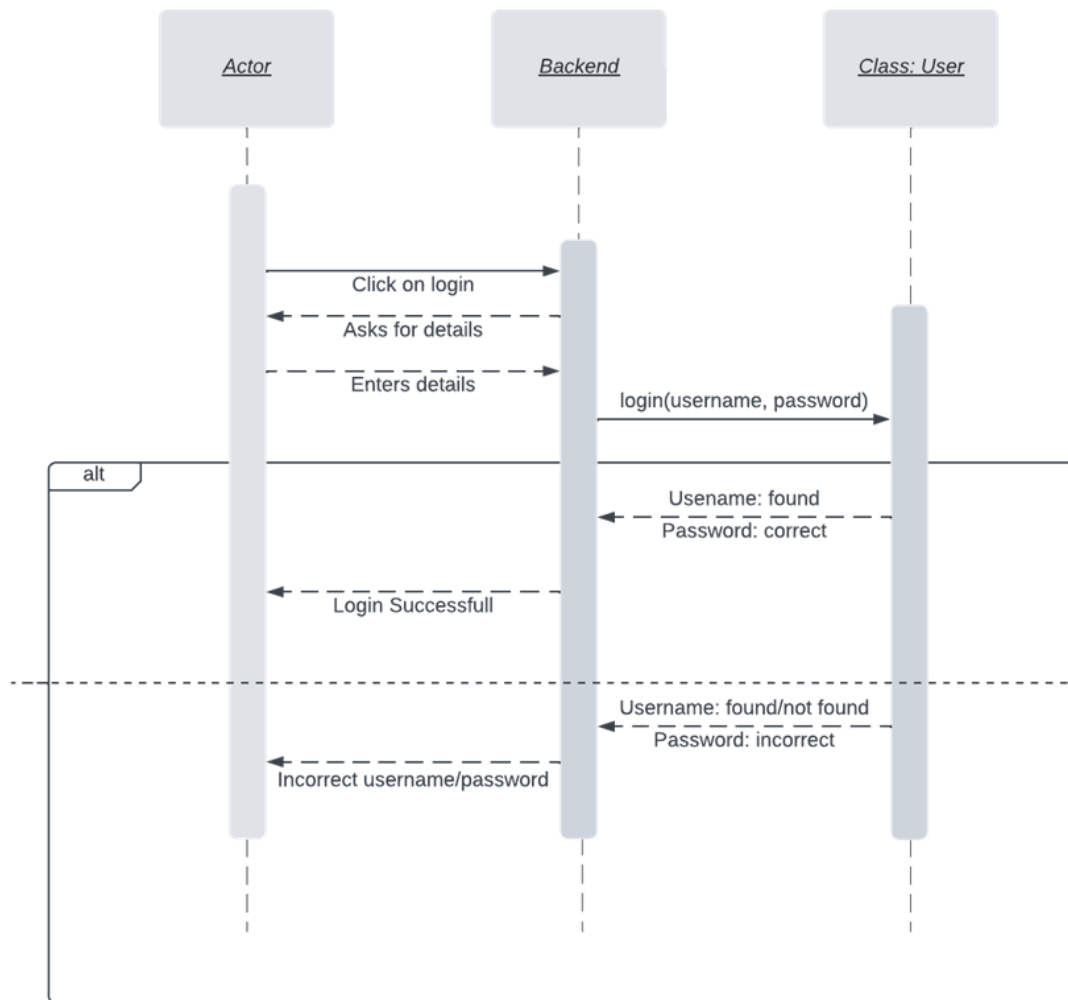


Figure 7: Sequence Diagram – Login

3. View Home Page

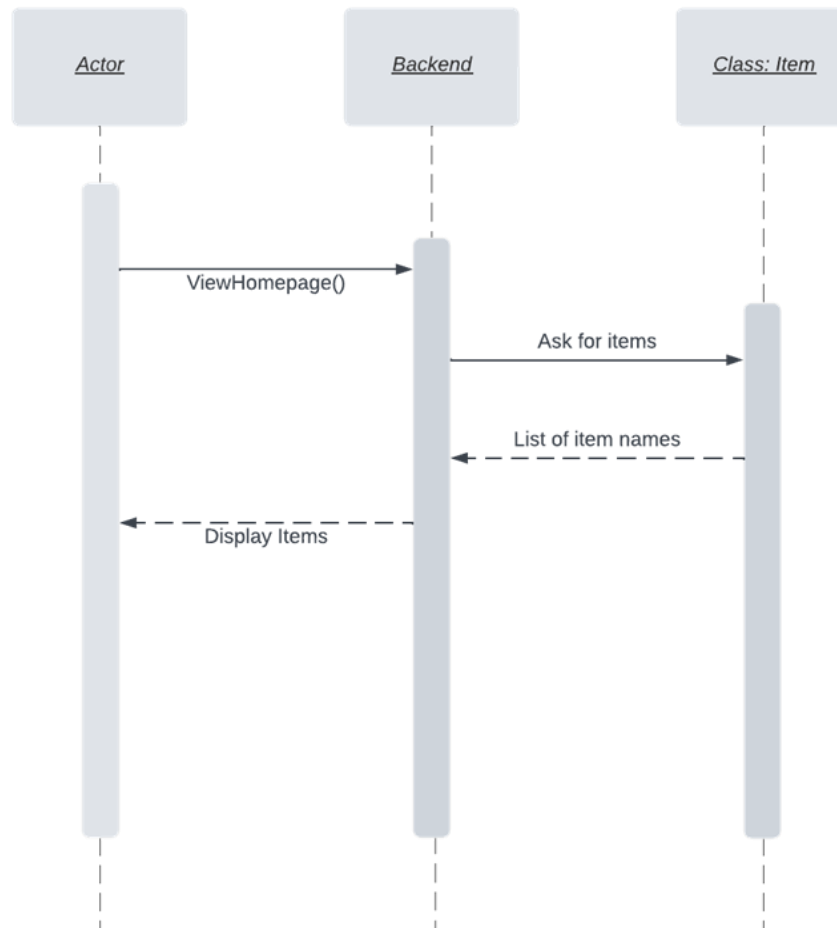


Figure 8: Sequence Diagram - View Home Page

4. View Product Description

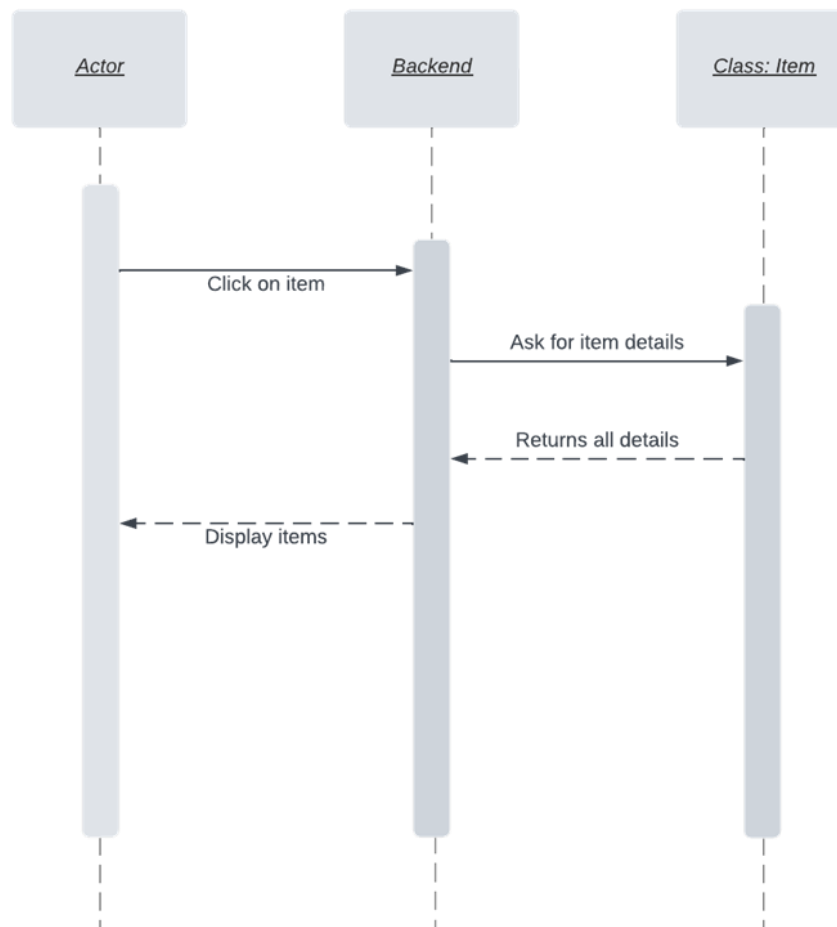


Figure 9: Sequence Diagram - View Product Description

5. Place Bid on Product

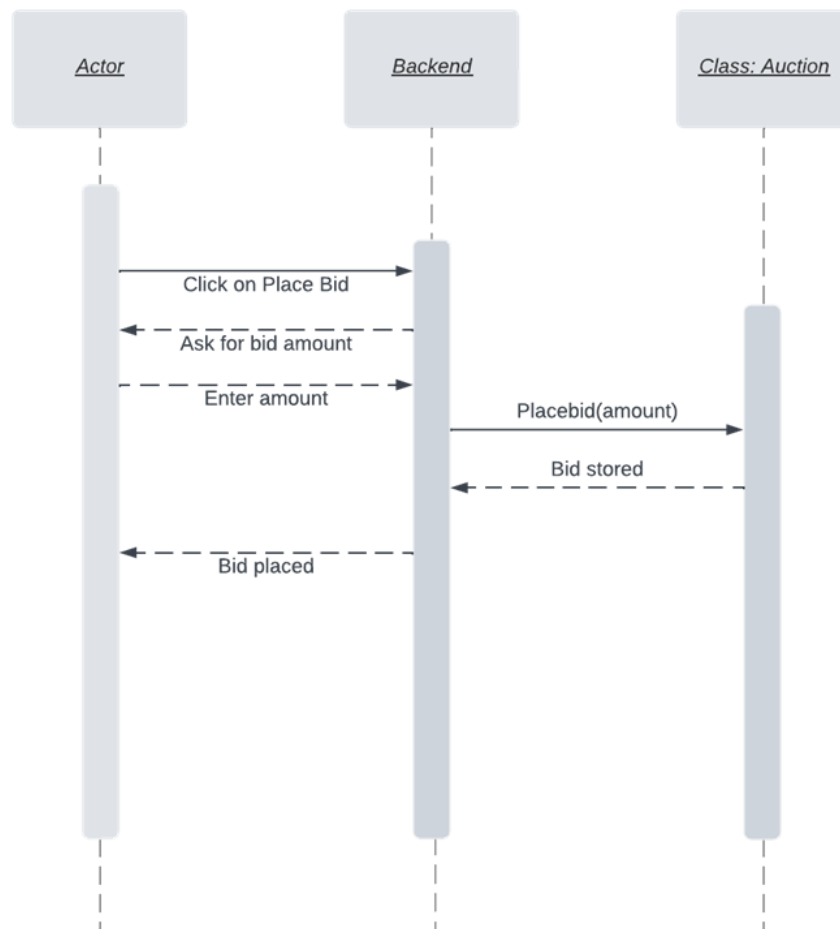


Figure 10: Sequence Diagram - Place bid

6. View Bids

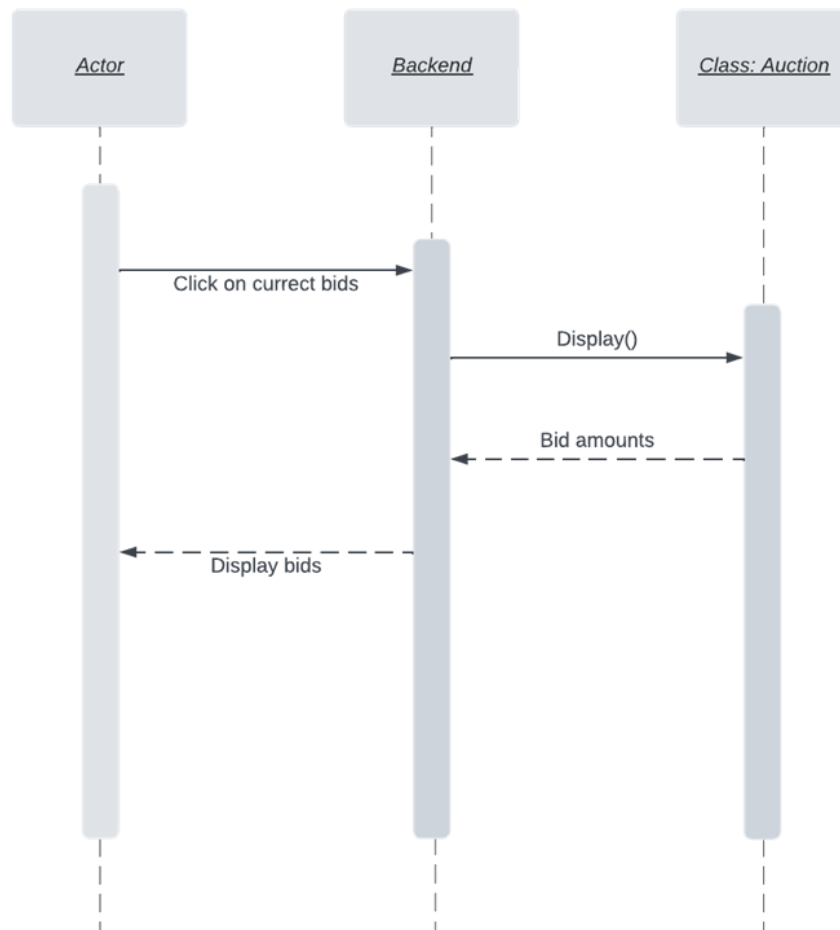


Figure 11: Sequence Diagram - View Bids

7. Upload Product for Auction

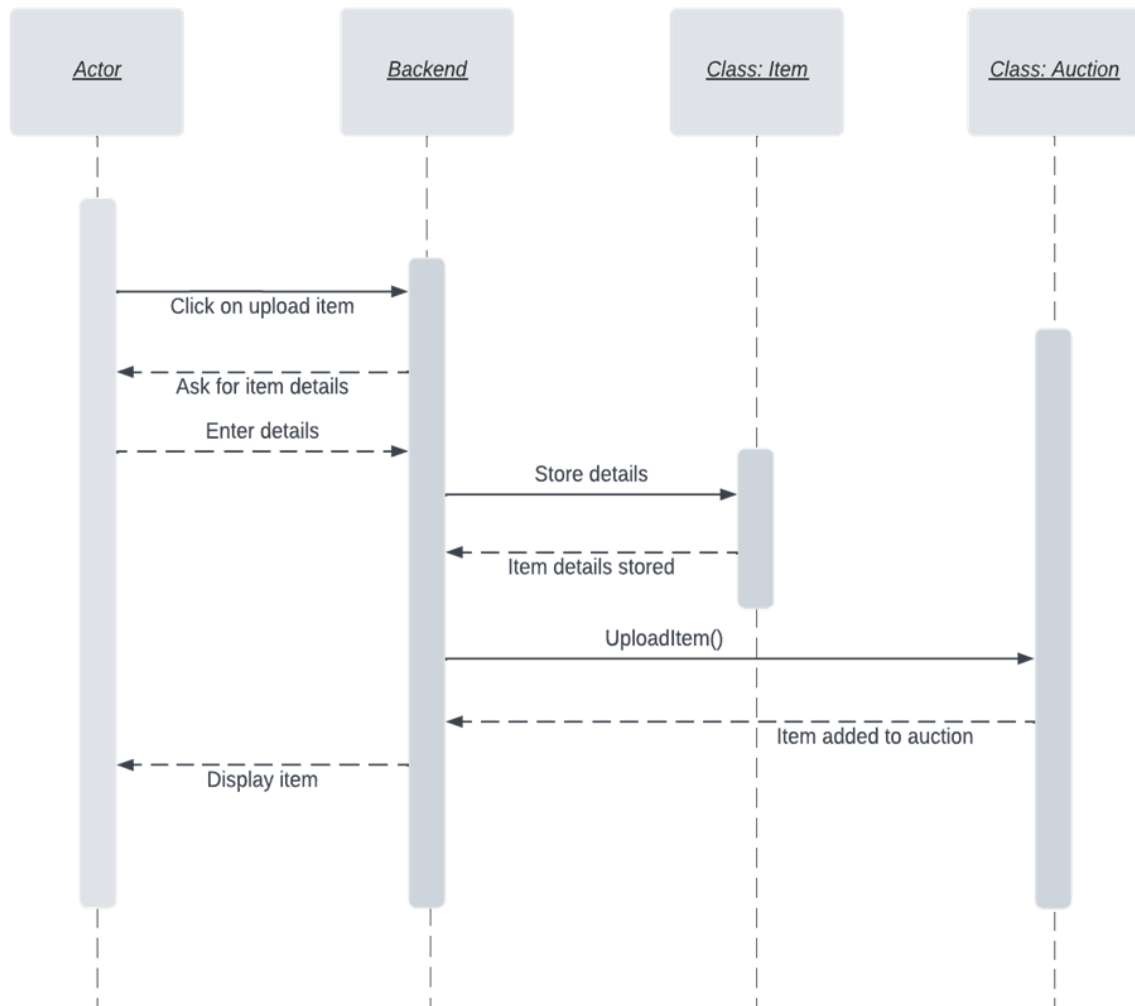


Figure 12: Sequence Diagram - Upload Product

8. View Profile

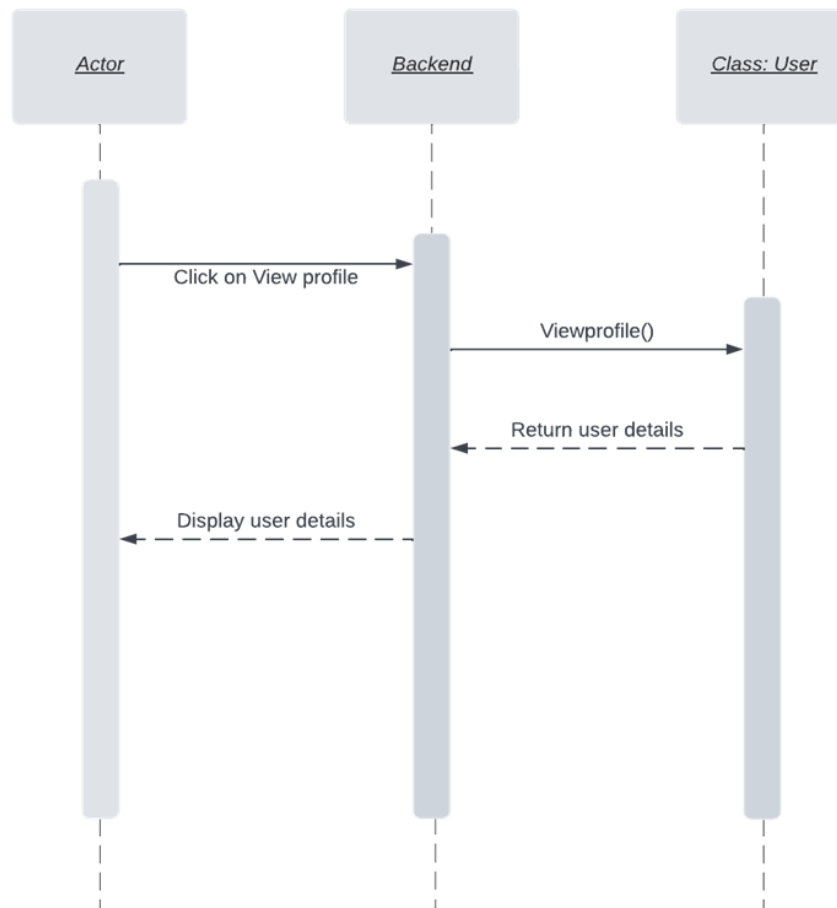


Figure 13: Sequence Diagram - View Profile

9. Edit Profile

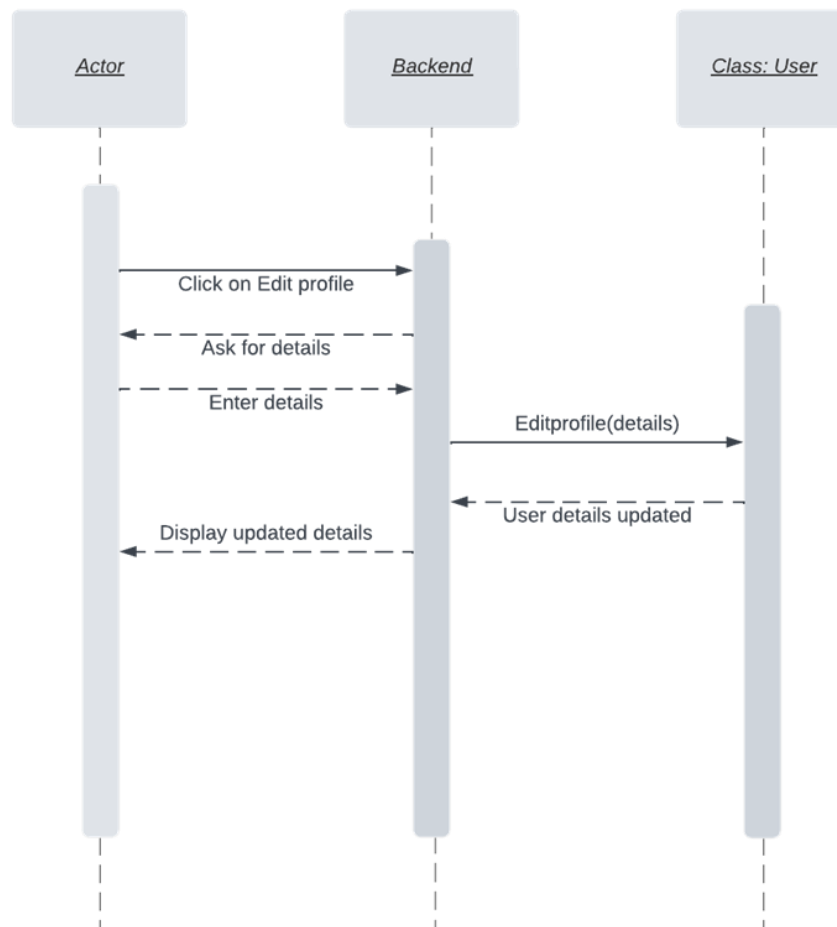


Figure 14: Sequence Diagram - Edit Profile

10. Logout

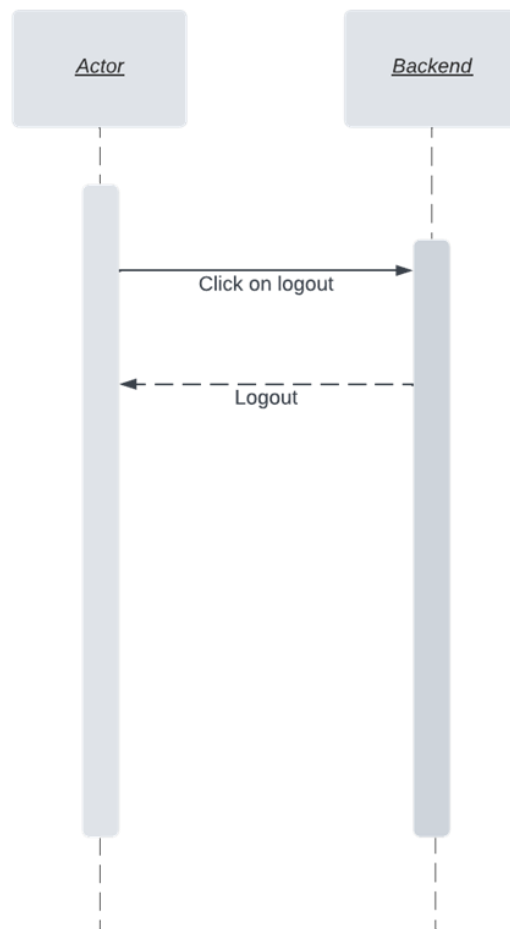


Figure 15: Sequence Diagram - Logout

5. Software Development Methodology and Plan

This chapter provides an overview of the selection process for software development methodologies, with a focus on comparing and contrasting the waterfall model and agile method. The chapter discusses the benefits and drawbacks of each methodology, and explains the decision to use the agile method for the project. Additionally, the chapter briefly touches on the use of a Gantt chart for project scheduling and monitoring.

a. Software Process Selection

Waterfall and Agile models are the two most commonly used software development methods. Since we are going to start the development phase of our project, we need to analyze the pros and cons of both models and choose the one that is best for us. The waterfall model is a sequential process which follows the 5 basic phases of development that are:

1. Requirements
2. Design
3. Implementation
4. Verification
5. Maintenance

Everything in the waterfall model is done step by step and requires documentation at every phase of the development process.

Pros:	Cons:
1. Since documentation is a significant part of the waterfall model hence it helps everyone in clearly understanding the objectives and also allows new	1. It does not allow for flexibility. Making changes to the requirements is

<p>developers to understand the project quickly if they join at a later stage.</p>	<p>not possible once you move on to the next phase.</p>
<p>2. Helps maintain timelines and makes it easy to make sure that each phase is completed on time.</p>	<p>3. Projects may take longer to deliver as steps are linearly followed.</p>
<p>4. Testing is easy to do as test scenarios are already explained in the functional specification of the requirements phase.</p>	<p>2. If initial requirements are faulty the problem is reflected on the entire project as it heavily relies on the requirements.</p>
<p>5. The outcomes are clearly communicated through documentation and everyone is on the same page.</p>	<p>3. It requires complete knowledge of the field that the project caters to as making changes later on is not possible.</p>
<p>6. Workarounds of potential development issues can be handled during the design stage before actual development begins.</p>	<p>7. It is only suitable for small-scale projects as large-scale projects often require changes.</p>
<p>8. Size and cost of the project is pre-calculated. There is no uncertainty or surprises that can occur.</p>	<p>9. It is difficult to estimate the time and cost for each stage of the model.</p>

The agile method on the other hand is an incremental approach and is also considered as a solution to the drawbacks of the waterfall method. Developers can work on small units on a weekly or monthly basis and it also allows for project priorities to be re-evaluated which is not possible in the waterfall model. It follows the following steps in every sprint:

1. Plan
2. Design
3. Build
4. Test
5. Review
6. Launch

Pros:	Cons:
1. It is a flexible model and allows to make changes even after the initial planning.	1. It might cause unexpected delays in delivery of the project.
2. It also allows to add features that keep you updated with advances in the industry.	3. The project cost may go over budget as changes are being made overtime.
4. It allows clients to give regular feedback to the developers and get the	2. If the initial plan is not stable then the final project may turn out to be different than what was intended.

product they want. It promotes better communication.	
3. The product gets to the market faster as the focus is on developing working deliverables and not perfect deliverables.	
4. It allows for regular testing.	

We will be following the agile methodology for the development of our project. The reasons for choosing agile method are listed as follows:

1. It is a small-scale project hence the agile method allows us to add features overtime.
2. We have a small team of 5 people and all the team members are completely aware of what we aim to achieve hence we don't require rigorous documentation at every stage.
3. Moreover, we are on a short timeline as we will have one and a half semester to complete our project so our goal is to produce working deliverables in shorter time spans instead of trying to develop a perfect deliverable at the end.
4. Since we are new developers with less work experience and are particularly new to this niche so we might need to make changes while developing and would need great flexibility there hence we cannot use the waterfall model.
5. Agile method allows for constant improvement in our project without any restrictions as we aim to deliver our project to the best of our abilities.

6. As agile teams are self-organized and self-managing, they have increased autonomy and authority over their decisions and allows every team member to learn and grow in their respective roles.

7. It also allows us to track and evaluate our progress through the deliverables that we produce.

b. Gantt Chart

The Link for our gantt chart is added below as the size of the gantt chart was not allowing us to add it inside the document. Please open the link below to view our gantt chart.

[Gantt Chart](#)

6. Database Design and Web Services

In this chapter, we will discuss the design of a system's data model using an Entity-Relationship (E/R) diagram, which depicts the various entities involved in the system and their relationships. Additionally, we will provide a list of external APIs that were used in the development of the system.

a. Database Design

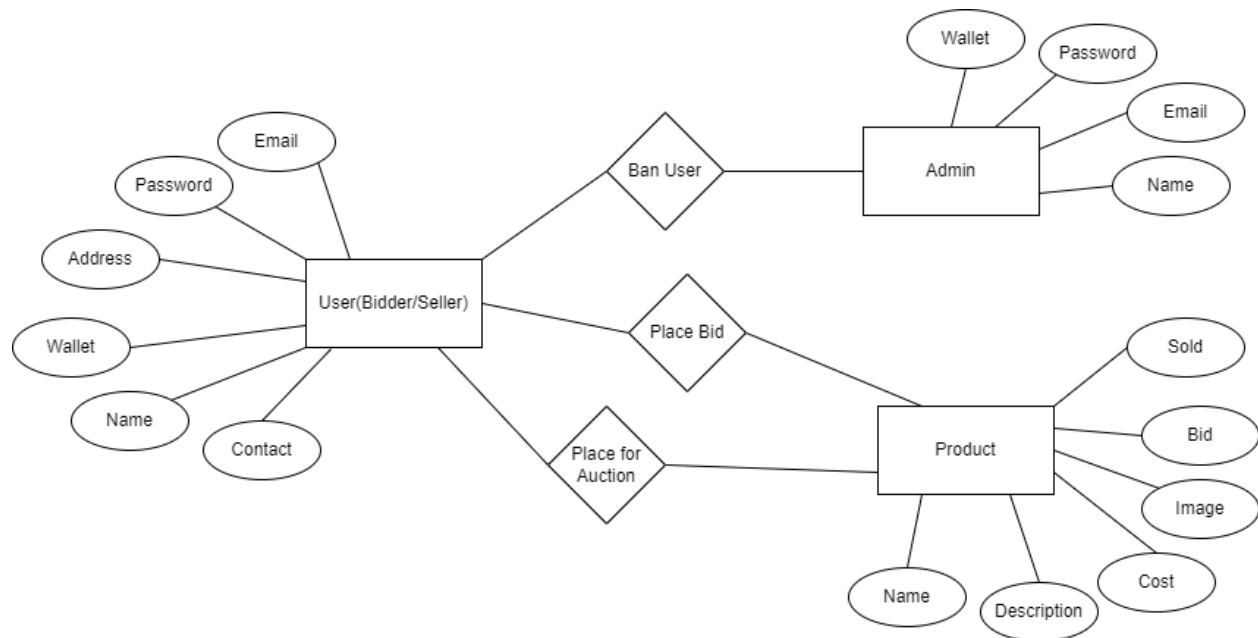


Figure 16: E/R Diagram

b. API Specification

1. Whatsapp API

We used the whatsapp API to integrate the chat feature in our website. It allows our user to have a live conversation with one of our customer service representatives. In case the customer has any queries that need to be taken care of right away such as issues with navigating through the website, they can get help from our representative.

7. System User Interface

First of all, when the user opens our website using the url, they are welcomed to the landing page of our website on which all the products up for auction are displayed. The right side of the page displays promotional advertisements. If a user wishes to bid on a product then they must click on the sign in button on the top right side of the page.

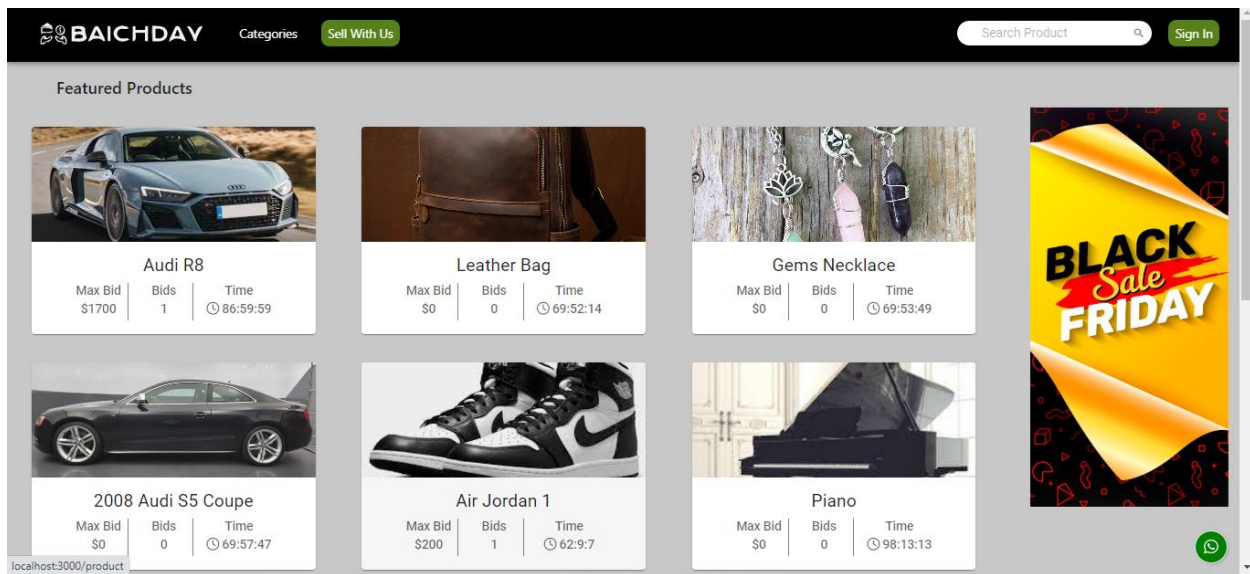


Figure 17: Homepage

The user can sign in on our website using their login credentials. After successfully logging in the user will be navigated to the landing page where they can bid on the products they want. In case a user does not already own an account then they can click on the sign up link shown below and create an account.

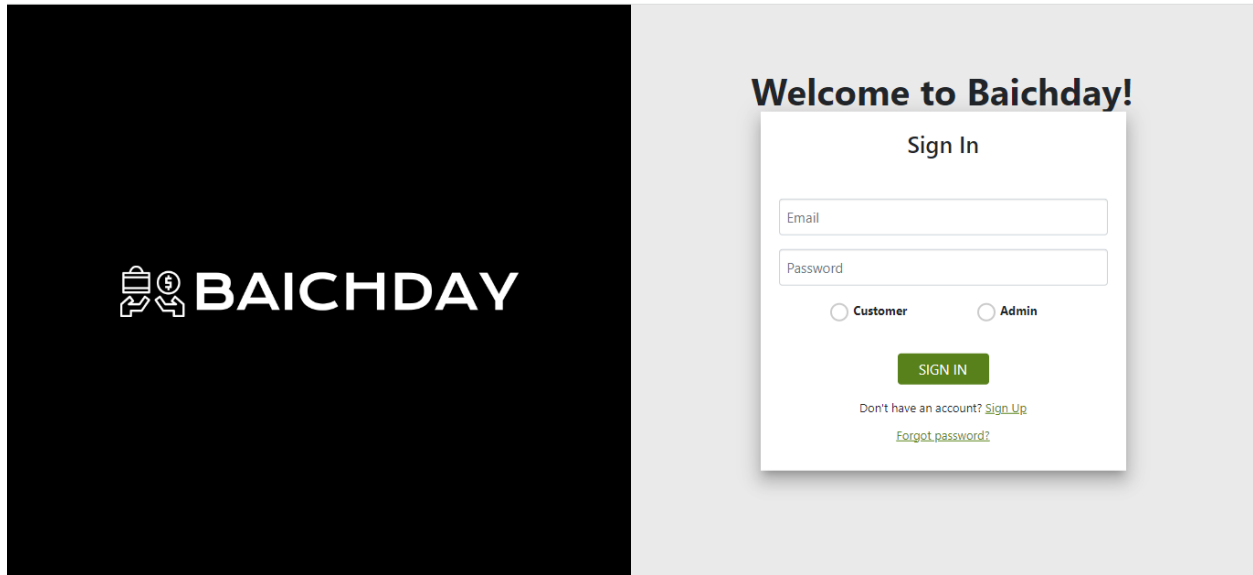
The image shows a web interface for signing in. On the left is a dark sidebar with the Baichday logo, which consists of a shopping cart icon with a dollar sign and the word "BAICHDAY" in white. On the right is a light gray main area. At the top of this area is the heading "Welcome to Baichday!". Below this is a white "Sign In" form. The form contains two input fields for "Email" and "Password". Below these fields are two radio buttons labeled "Customer" and "Admin". A green "SIGN IN" button is positioned below the radio buttons. At the bottom of the form, there are two links: "Don't have an account? [Sign Up](#)" and "[Forgot password?](#)".

Figure 18: Sign In

After signing in on the website, if the user wishes to bid on a product they need to have sufficient balance in their account. They can view their wallet by clicking on the profile icon on the top right side of the landing page.

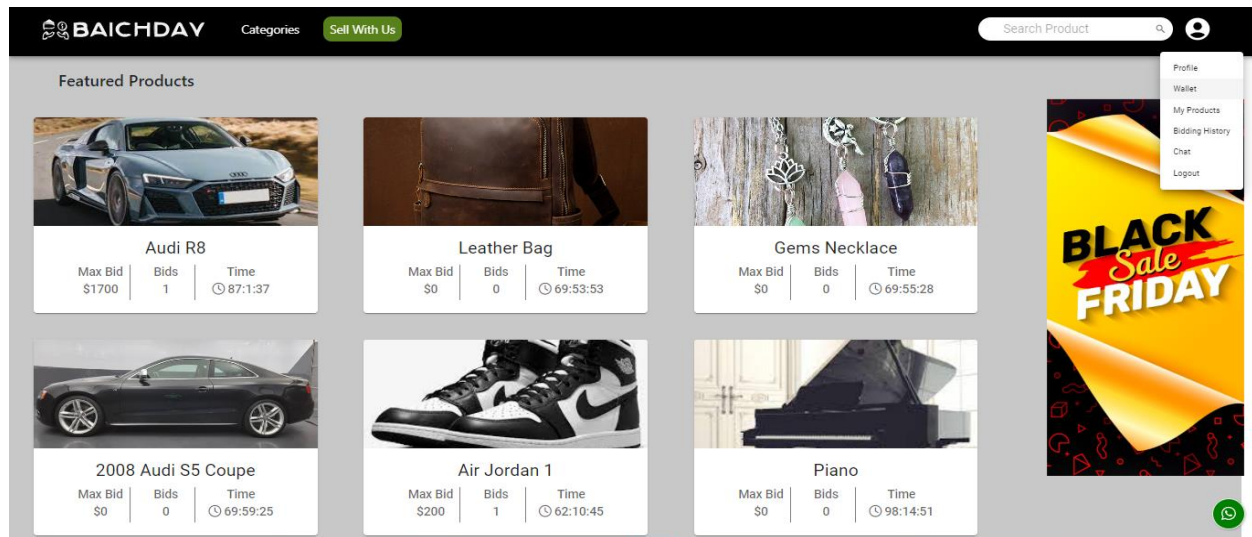


Figure 19: Homepage Signed In

By clicking on the wallet tab, the user will be redirected to the wallet page where they can view their account balance and a list of their current bids will also be displayed to them.

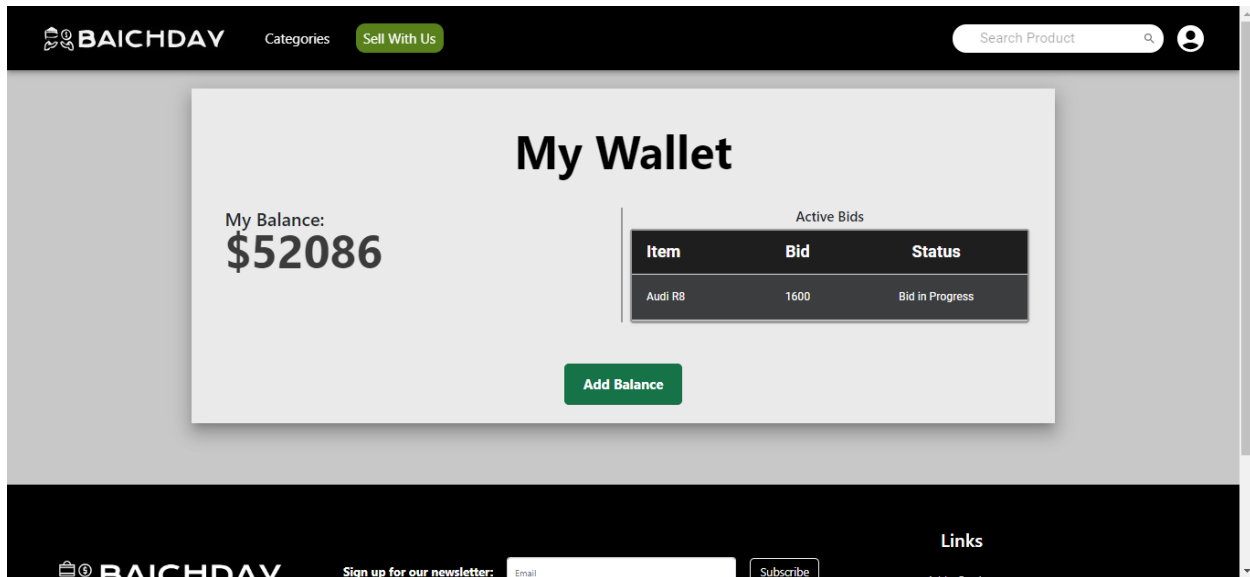
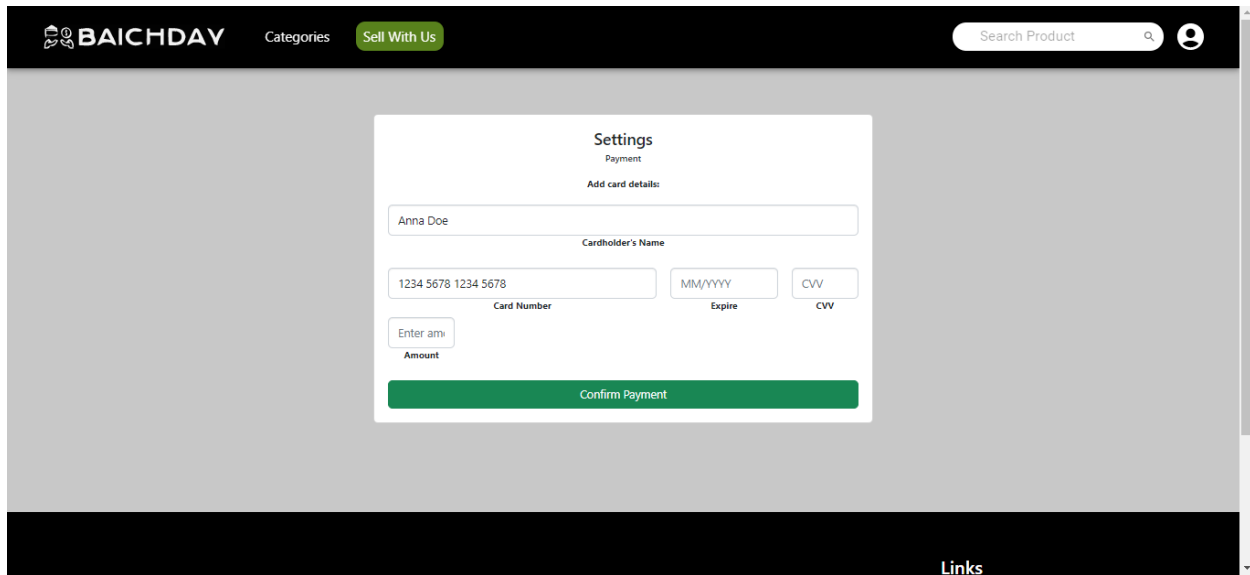


Figure 20: My Wallet

The user can then click on the add balance button if they want to add balance to their account. On clicking the add balance button the user will be redirected to the payment page where they can add their card details to transfer money into their Baichday account.



The screenshot displays the Baichday website's 'Settings - Payment' interface. The header includes the Baichday logo, navigation links for 'Categories' and 'Sell With Us', a search bar, and a user profile icon. The main content area features a white modal form titled 'Settings - Payment' with the subtitle 'Add card details:'. The form contains several input fields: a text field for the cardholder's name (pre-filled with 'Anna Doe'), a card number field (pre-filled with '1234 5678 1234 5678'), an expiration date field (pre-filled with 'MM/YYYY'), and a CVV field. Below these is an 'Enter amount' field. A green 'Confirm Payment' button is positioned at the bottom of the form. The footer of the page includes a 'Links' section.

Figure 21: Add Credit

After adding balance to their wallet they can come back to the home page by clicking on the website icon on the left side of the navbar. The user can then select any product that they want to bid on which will redirect them to the product page. The user can view the product description, minimum bid price, number of bids placed currently, and the time left for the auction to expire. The user can enter the amount they wish but it has to be greater than the minimum bid price and the maximum bid that has been currently placed on the product and then press the bid now button to place their bid.

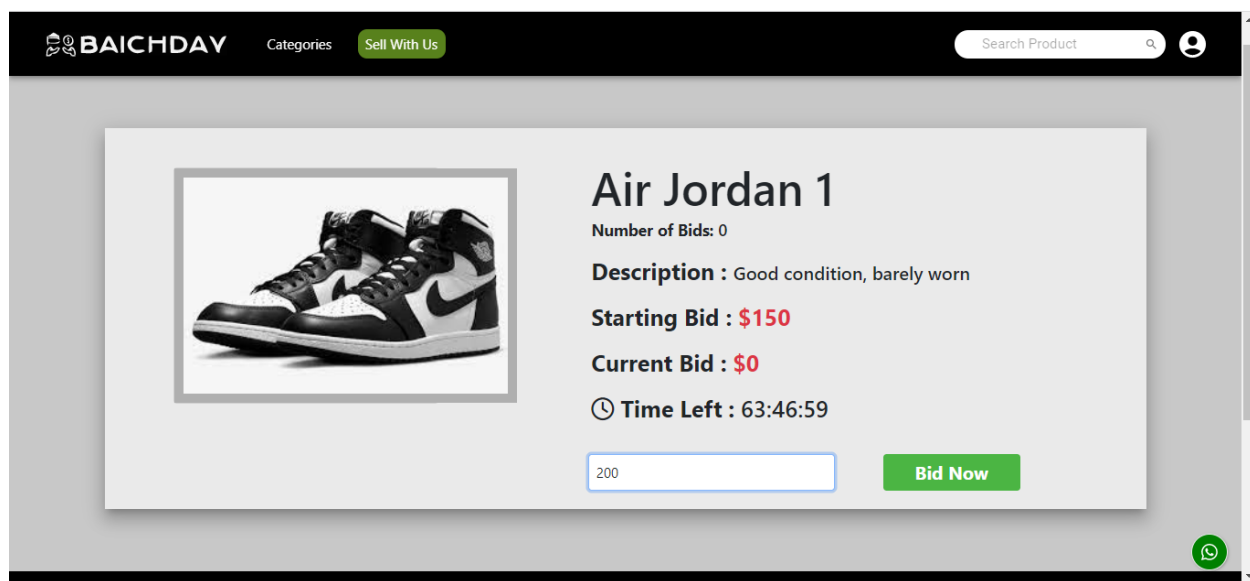


Figure 22: View Product

After placing the bid the user can view the products that they have placed a bid on by selecting the Biding Hisory tab that can be seen by clicking on the profile icon on the right side of the navbar.

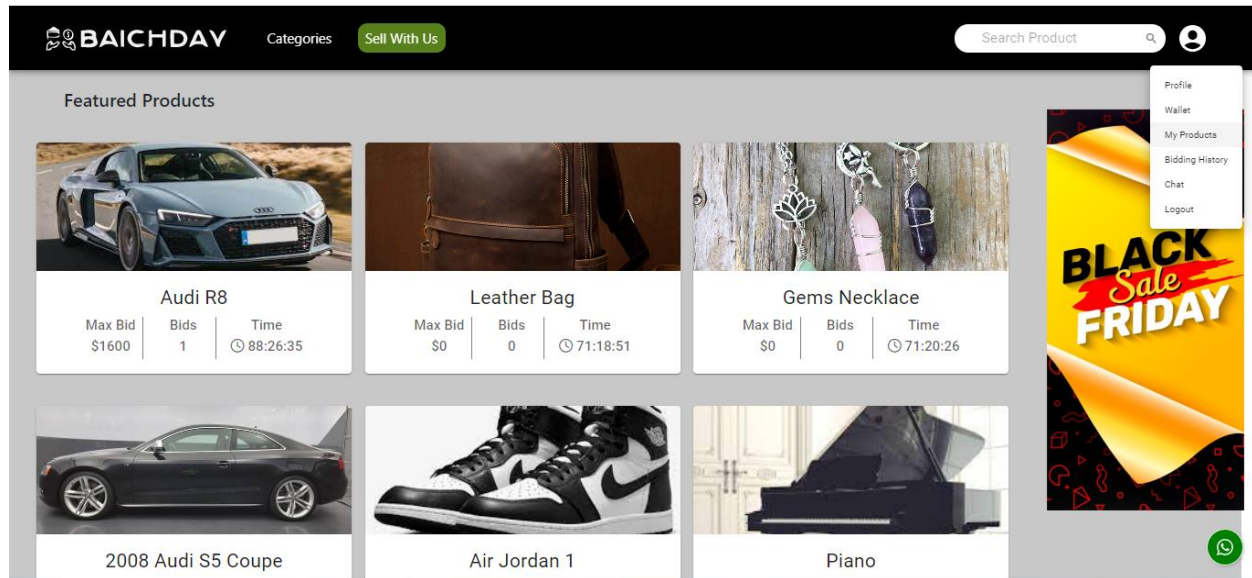
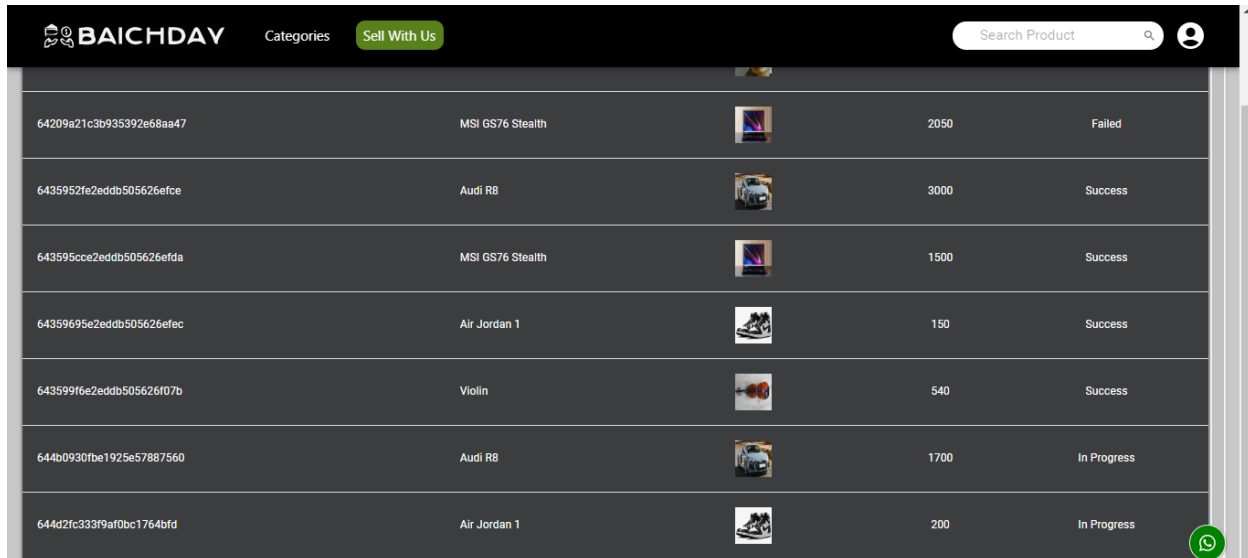


Figure 23: Homepage

After clicking on the Bidding History tab the user will be redirected to a list of bids that he has placed. The item status can be seen over here. After the timer expires the product status changes to success if the user is the highest bidder and the user becomes the new owner of the product.










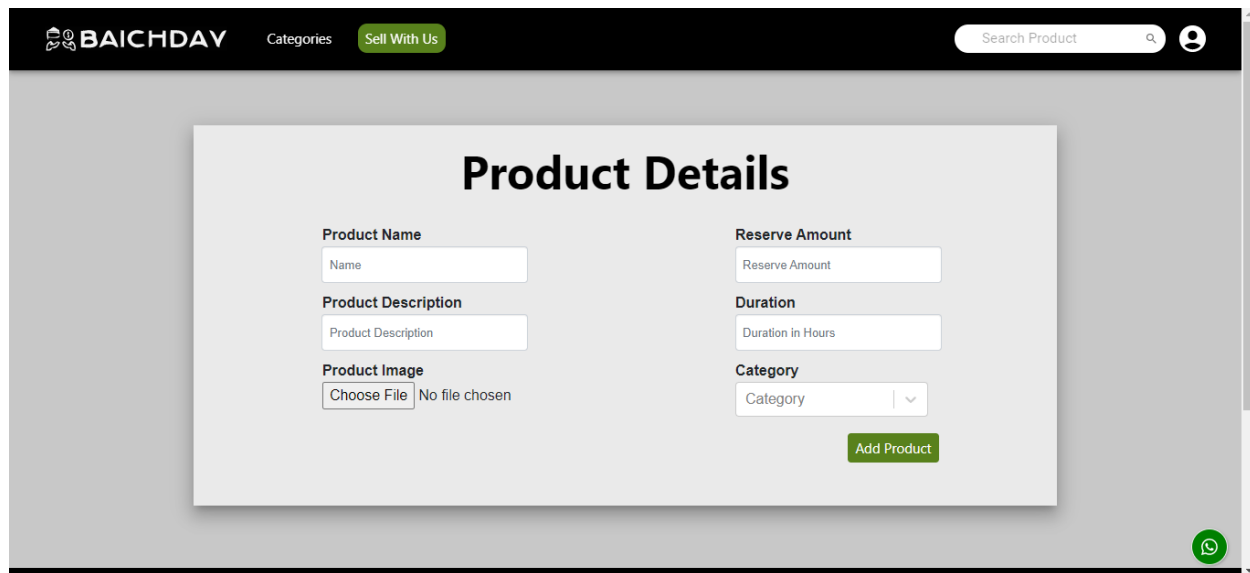
Bid ID	Product Name	Product Image	Bid Amount	Status
64209a21c3b935392e68aa47	MSI GS76 Stealth		2050	Failed
6435952fe2eddb505626efce	Audi R8		3000	Success
643595cce2eddb505626efda	MSI GS76 Stealth		1500	Success
64359695e2eddb505626efec	Air Jordan 1		150	Success
643599f6e2eddb505626f07b	Violin		540	Success
644b0930f8e1925e57887560	Audi R8		1700	In Progress
644d2fc333f9af0bc1764bfd	Air Jordan 1		200	In Progress

Figure 24: Item History

Similarly, the user can also set their own items up for auction using the same account. By clicking on the sell with us button in the navbar, the user can upload their product details and set it up for auction. The user has to fill all the required fields and upload an image of their product by clicking on the choose file button. After adding all the required details, the product will go live by clicking on the add product button on the bottom right side of the form.

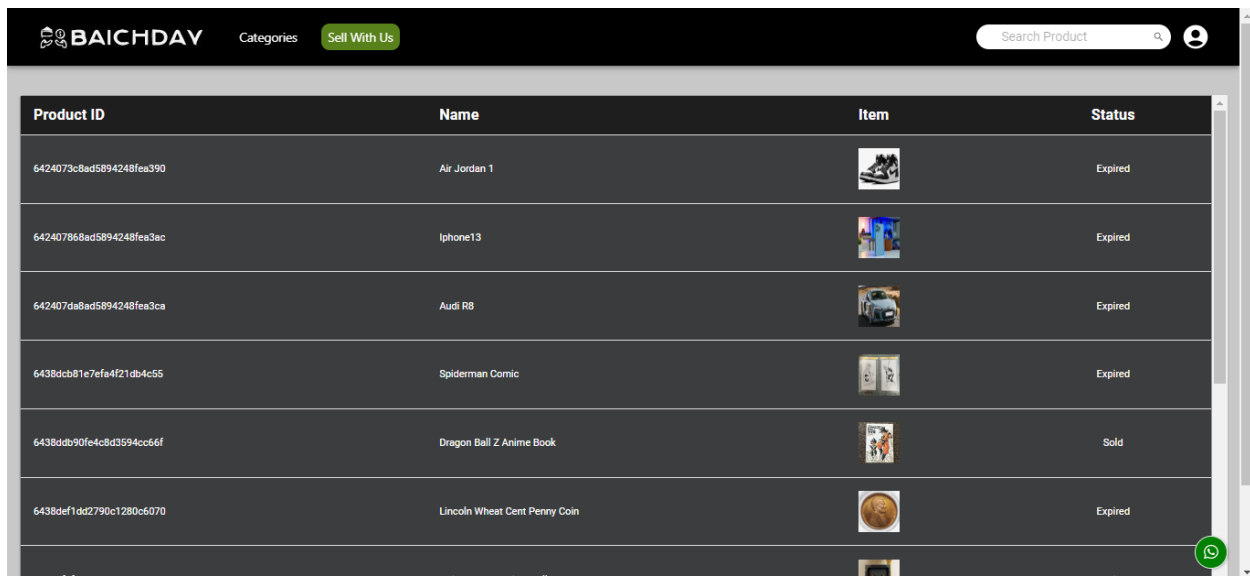


The screenshot shows the 'Product Details' form on the BAICHDAY website. The form is centered on a light gray background. It contains several input fields and a dropdown menu, organized into two columns. The left column includes fields for 'Product Name' (with a placeholder 'Name'), 'Product Description' (with a placeholder 'Product Description'), and 'Product Image' (with a 'Choose File' button and the text 'No file chosen'). The right column includes fields for 'Reserve Amount' (with a placeholder 'Reserve Amount'), 'Duration' (with a placeholder 'Duration in Hours'), and a 'Category' dropdown menu. A green 'Add Product' button is located at the bottom right of the form. The website's header is black with the BAICHDAY logo, a 'Categories' link, a 'Sell With Us' button, a search bar, and a user profile icon. A WhatsApp icon is visible in the bottom right corner of the page.

Product Details	
Product Name <input type="text" value="Name"/>	Reserve Amount <input type="text" value="Reserve Amount"/>
Product Description <input type="text" value="Product Description"/>	Duration <input type="text" value="Duration in Hours"/>
Product Image <input type="button" value="Choose File"/> No file chosen	Category <input type="text" value="Category"/> ▾
<input type="button" value="Add Product"/>	

Figure 25: Upload Product

After uploading their products the user can view the status of their products by clicking on the My Products tab in the profile icon dropdown. The user will be displayed with a list of their auctioned products. The product status is changed to sold if no customer places a bid on the product before the timer expires.



The screenshot shows the BAICHDAY app interface. At the top, there is a navigation bar with the BAICHDAY logo, a 'Categories' link, a 'Sell With Us' button, a search bar labeled 'Search Product', and a profile icon. Below the navigation bar is a table displaying a list of auctioned products. The table has four columns: 'Product ID', 'Name', 'Item', and 'Status'. The items listed are 'Air Jordan 1', 'Iphone13', 'Audi R8', 'Spiderman Comic', 'Dragon Ball Z Anime Book', and 'Lincoln Wheat Cent Penny Coin'. The status of most items is 'Expired', while 'Dragon Ball Z Anime Book' is 'Sold'. A green WhatsApp icon is visible in the bottom right corner of the table area.







Product ID	Name	Item	Status
6424073c8ad5894248fea390	Air Jordan 1		Expired
642407868ad5894248fea3ec	Iphone13		Expired
642407da8ad5894248fea3ca	Audi R8		Expired
6438dcb81e7efa4f21db4c55	Spiderman Comic		Expired
6438ddb90fe4c8d3594cc66f	Dragon Ball Z Anime Book		Sold
6438def1d42790c1280c6070	Lincoln Wheat Cent Penny Coin		Expired

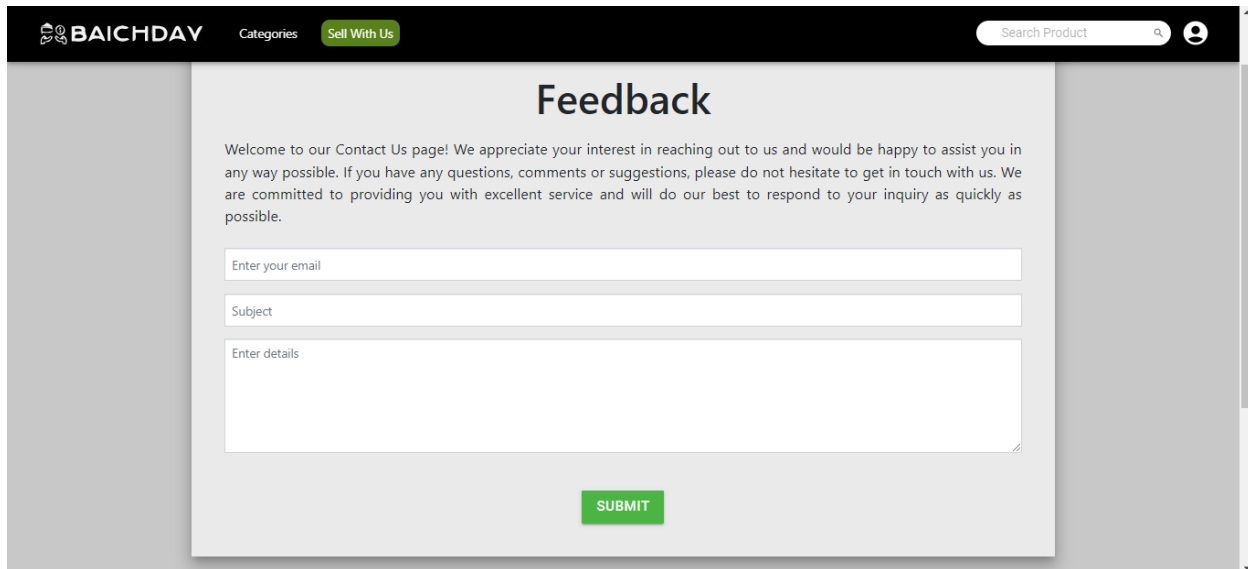
Figure 26: Item History

Now that we have concluded how to bid or sell products, the user can also contact us for any kind of complaints and queries that they may have. They can do this by clicking on the contact us link in the footer.



Figure 27: Footer

After clicking on the contact us page the user will be redirected the following page.



The screenshot shows a web browser window with a dark header. On the left, the BAICHDAY logo is visible. In the center of the header, there are links for 'Categories' and 'Sell With Us'. On the right, there is a search bar labeled 'Search Product' and a user profile icon. The main content area has a light gray background with the title 'Feedback' in a large, bold font. Below the title is a welcome message: 'Welcome to our Contact Us page! We appreciate your interest in reaching out to us and would be happy to assist you in any way possible. If you have any questions, comments or suggestions, please do not hesitate to get in touch with us. We are committed to providing you with excellent service and will do our best to respond to your inquiry as quickly as possible.' Below this message are three input fields: 'Enter your email', 'Subject', and 'Enter details'. At the bottom center of the form is a green 'SUBMIT' button.

Figure 28: Feedback

The user complaints will reach us through email and we will cater to their complaints. However, if the user has issues with navigating the website that they need help with spot on, they can click on the whatsapp icon thru which they can start a chat with one of our customer service representatives that will help them solve their issue.

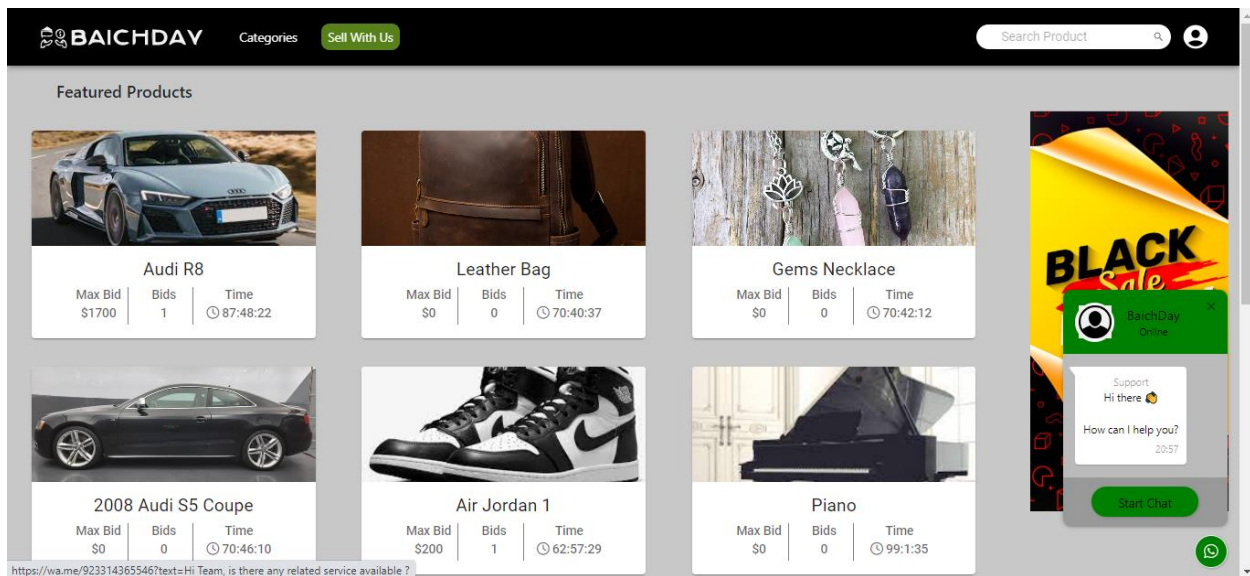


Figure 29: Chatbot

The user can also edit their account details by clicking on the my profile tab that is displayed by clicking on the profile icon. They can make changes to their details and save them by clicking the apply button. The user is also able to delete their account by clicking on the delete account button.

The screenshot shows a web browser window with the address bar displaying 'localhost:3000/CustomerProfile'. The website header includes the 'BAICHDAY' logo, 'Categories', 'Sell With Us', and a search bar. The main content area is titled 'My Profile' and features a circular profile picture with the initials 'WZ'. Below the profile picture, there are two columns of form fields. The left column contains 'First Name' (with 'Wbagar' entered), 'Last Name' (with 'Zaka' entered), 'Email' (with 'waqarzaka@gmail.com' entered), and 'Password' (with masked characters). The right column contains 'Contact', 'House Address', 'City', and 'Country' (a dropdown menu). At the bottom of the form, there is a red 'DELETE ACCOUNT' button on the left and a green 'APPLY' button on the right. The browser's taskbar at the bottom shows various application icons and the system clock indicating 9:00 pm on 29/04/2023.

Figure 30: My Profile

8. Project Security

The security of a project is essential to protect against potential threats and losses. This section aims to identify and address potential security threats to the system. To do this, the information and functionality that are most vulnerable from a security perspective in the context of the project has been carefully considered. By identifying potential security risks and losses, appropriate security controls can be implemented. These controls can be categorized into detective, protective, responsive, and recovery controls, and may include input validation, audit logs, multi-factor authentication, user roles, and more. By implementing these controls, the project can protect against potential security threats and minimize potential losses.

a. Project Threats

Our auction web application has the potential to be affected by the following threats which need to be addressed or else risk consequences.

- **Fraudulent Activity**

1. Seller does not dispatch the same item as displayed.
2. Buyer demands a bank reversal after bidding on the product.
3. Buyer claims to have not received product
4. Seller has changed their mind about selling after the deadline to reconsider and refuses to send the product.

- **Malicious Attacks**

1. Denial of Service Attack
2. Injection Attack
3. Cross-site request forgery
4. Cross-site scripting

- **Legal Threats**

1. Seller has sold items that do not belong to them which makes our platform an accomplice to theft and fraud.
2. Seller has listed items that are not legal in context to Pakistan
3. Buyer has performed fraudulent and unauthorized transactions to purchase an item.

b. Potential Losses

- **Fraudulent Activity**

1. **Seller does not dispatch the same item as displayed:** This has the potential to cause reputation loss to the platform as the buyer will lose trust in purchasing items on the platform even if they come to no financial harm.
2. **Buyer demands a bank reversal after bidding on the product:** This can cause a financial loss to the platform if not handled correctly as the platform will have to bear the loss if the bank reverses the transaction.
3. **Buyer claims to have not received product:** This can cause financial, reputation losses as well as litigation against the platform. This can raise a dispute that will end up in legal proceedings.
4. **Seller has changed their mind about selling after the deadline to reconsider and refuses to send the product:** This has the potential to cause reputation loss to the platform as the bidder does not receive the item. The platform can not possibly force the bidder to dispatch the item against their will.

- **Malicious Attacks**

1. **Denial of Service Attack:** This can result in financial losses as users will not be able to access the platform.
2. **Injection Attack:** This can result in financial, reputation and also result in legal consequences for not ensuring proper security.
3. **Cross-site request forgery:** This can result in financial, reputation and also result in legal consequences for not ensuring proper security.
4. **Cross-site scripting:** This can result in financial, reputation and also result in legal consequences for not ensuring proper security.

- **Legal Threats**

1. **Seller has sold items that do not belong to them which makes our platform an accomplice to theft and fraud:** This can be a cause for legal proceedings against the platform and lead to severe reputation and financial losses.
2. **Seller has listed items that are not legal in context to Pakistan:** This can be a cause for legal proceedings against the platform and lead to severe reputation and financial losses.

3. **Buyer has performed fraudulent and unauthorized transactions to purchase an item:** This can lead to financial losses as the seller will have to be compensated or else risk reputation losses.

c. Security Controls

Detective Controls:

- Implement monitoring and auditing tools to detect suspicious activity
- Implement user rating and review systems to detect and prevent fraudulent activity
- Regularly review user activity and transaction logs for unusual behavior
- Use algorithms to detect usage of words that imply illegal and unwanted activity.

Protective Controls:

- Implement secure coding practices to prevent injection attacks and cross-site scripting
- Implement input validation and verification measures to ensure product listings are legal and accurate
- Implement buffer size control as well make input non executable to ensure no buffer overflow attacks. Also ensure input files are of supported types only (.jpg, .png, etc) to prevent uploading of files containing malicious code.
- Implement two-factor authentication and password policies to prevent unauthorized access to accounts

Responsive Controls:

- Implement customer support and dispute resolution procedures to handle fraudulent activity and legal threats
- Implement fraud detection and prevention measures to identify and block unauthorized transactions
- Establish contingency plans in case of website downtime or legal action

Recovery Controls:

- Implement data backup and recovery procedures to recover from data breaches or website downtime

- Implement disaster recovery plans to recover from website attacks or server failures
- Implement incident response plans to quickly respond to security incidents and minimize damage.

d. Static and Dynamic Security Scanning Tools

SonarQube:

SonarQube is an open-source platform for static code analysis that can be used to identify and resolve code quality and security issues. It supports over 25 programming languages and can analyze source code for a variety of issues, including security vulnerabilities, bugs, and code smells. It provides real-time feedback on code quality, and can integrate with development tools such as IDEs and continuous integration (CI) servers. SonarQube provides detailed reports and metrics on code quality, making it a valuable tool for improving the security and maintainability of software projects.

OWASP ZAP:

OWASP ZAP (Zed Attack Proxy) is a free, dynamic, open-source web application security scanner that can help identify and mitigate potential security vulnerabilities in web applications. It can be used to scan for a wide range of vulnerabilities, including SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF). It provides a GUI and a REST API for performing scans and generating reports, making it easy to integrate into existing workflows. OWASP ZAP is actively maintained and updated by a community of developers, and is a valuable tool for improving the security of web applications.

9. Risk Management

Potential Risks and Mitigation Strategies

Sr.	Risk Description	Mitigation Strategy
1.	User privacy risks.	Salted Hashing used. Passwords are not saved as plain text. Other encryption is also implemented and can be further improved upon in the future.
2.	Route Protection. Users should not be able to reach web pages by simply writing the url. The web application should have a flow of access.	Users need to be signed in to access certain pages. In addition, our frontend deployment platform has built in route protection. Only the landing page is accessible by directly entering the url.
3.	Secure card payments. Card Details are extremely sensitive information.	We do not store or save card details to prevent malicious activity. This does add a layer of extra effort by the user but increases security. Also use third party secure APIs such as Stripe for processing payments to prevent financial fraud.
4.	Users may use the site for trading of illegal substances.	To ensure that contraband and other illicit materials are not purchased or sold, an additional layer of security can be implemented that reports usage of trigger words in item name or description.
5.	Fraudulent activities. The website may attract fraudulent users or activities, leading to loss of revenue and reputation damage.	To mitigate this risk, stronger user verification measures can be implemented as well as monitoring user activity for suspicious behavior,

		and providing clear guidelines on prohibited activities.
6.	Legal compliance. Websites in Pakistan often face the risk of censorship by PTA.	Ensure that the website complies with the legal rules and regulations of websites in accordance with PTA to prevent banning of website.
7.	Scalability. The website may not be able to handle increasing traffic or user volume, leading to slow performance or crashes.	To mitigate this risk, regularly monitor website traffic and performance, optimize code for efficient processing, and implement scalability measures such as load balancing.
8.	User Experience. The user interface might not be attractive and easy enough for users which can affect user retention and engagement.	Conduct user testing and feedback sessions, also optimize website design and functionality
9.	Product is the same as displayed by the seller.	To mitigate this risk, we can either use a third party verification service and add the verified flag or wait till the buyer confirms that the product is the correct one before releasing payment to seller.
10.	Logistics challenges. We need to ensure that the seller has dispatched the product and the buyer has transferred the payment.	To mitigate this risk, research and partner with reputable logistics providers. The buyer should deliver the product to the logistic partner and the seller should provide the payment in Escrow style.

10. Testing and Evaluation

We were continuously in the testing stage throughout our development cycle. However, that testing was done manually in order to deal with the corner cases. In the testing phase, we applied automated testing.

To test our application, we used the e2e (end-to-end) testing strategy. In this strategy, the website's front-end is accessed and a certain use case can then be tested for instance login, signup, etc. In end-to-end testing, the Front-End and Back-End are both tested so we did not need unit testing here. Although, end-to-end testing is relatively slower, but our code base is smaller than large-scale systems and only a few test cases were supposed to be made. That is why end-to-end testing was preferable in our case. We used **Cypress** in order to test our MERN stack application. Test scripts were written in javascript.

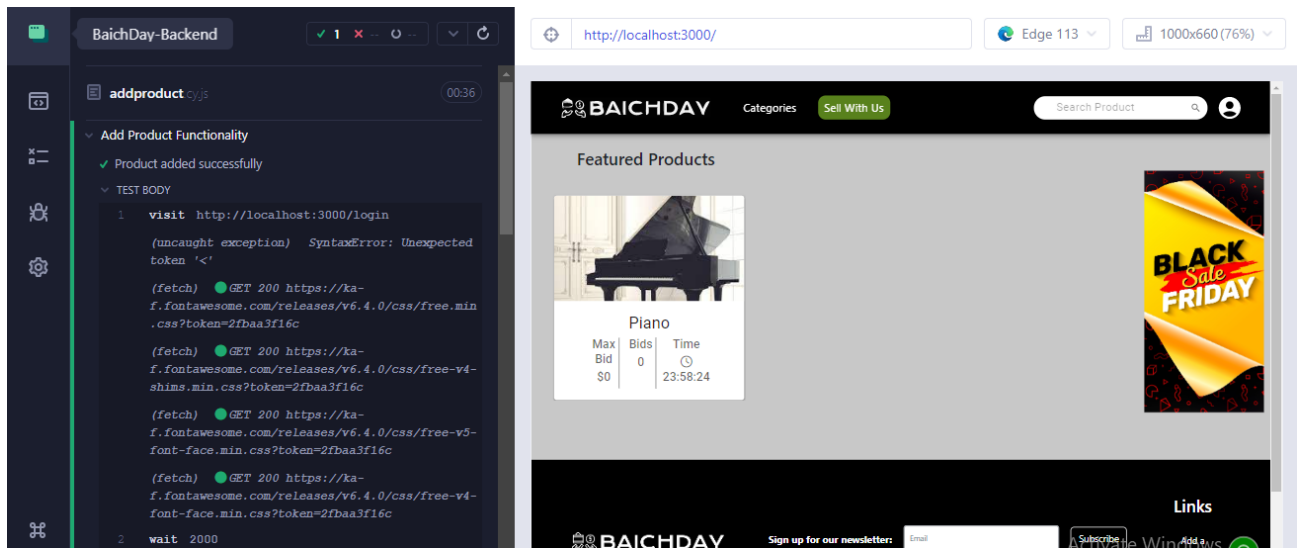
In Cypress, the ".cy.js" file extension is used for files that contain test code. Cypress tests are typically written in JavaScript, and the ".cy.js" extension is used to indicate that the file contains Cypress-specific test code. This helps to distinguish Cypress tests from other types of JavaScript code that may be used in a web application.

Below is the list of test cases covered by automated testing using cypress:

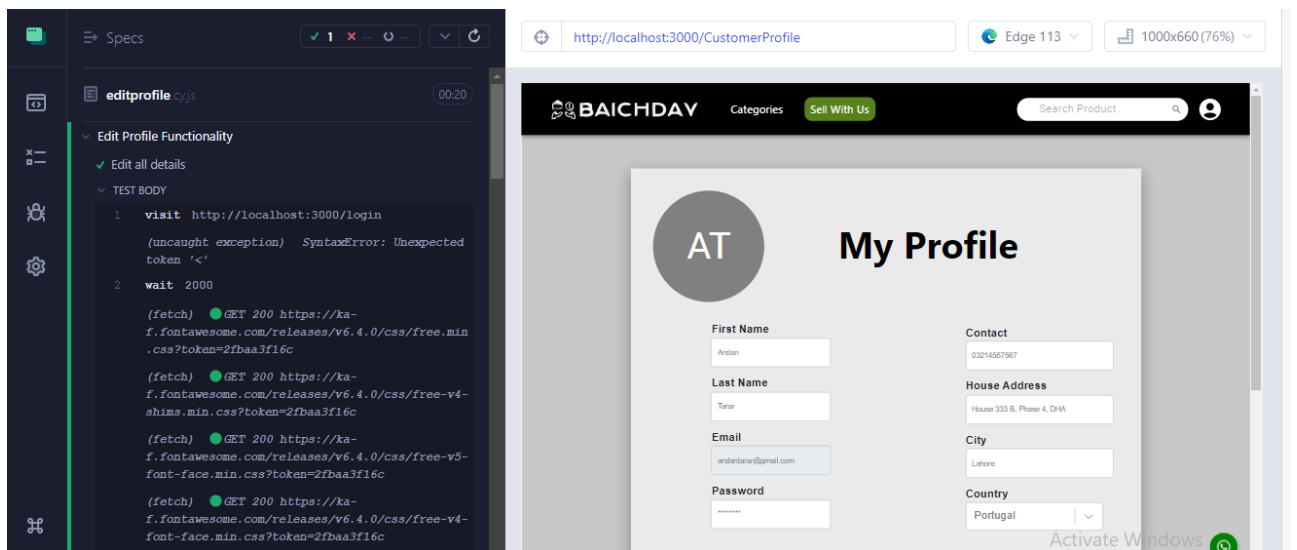
- Add a product to the web application.
- Bid on a product.
- Deactivate the account.
- Edit profile of the user.
- User login.

Automated Testing:

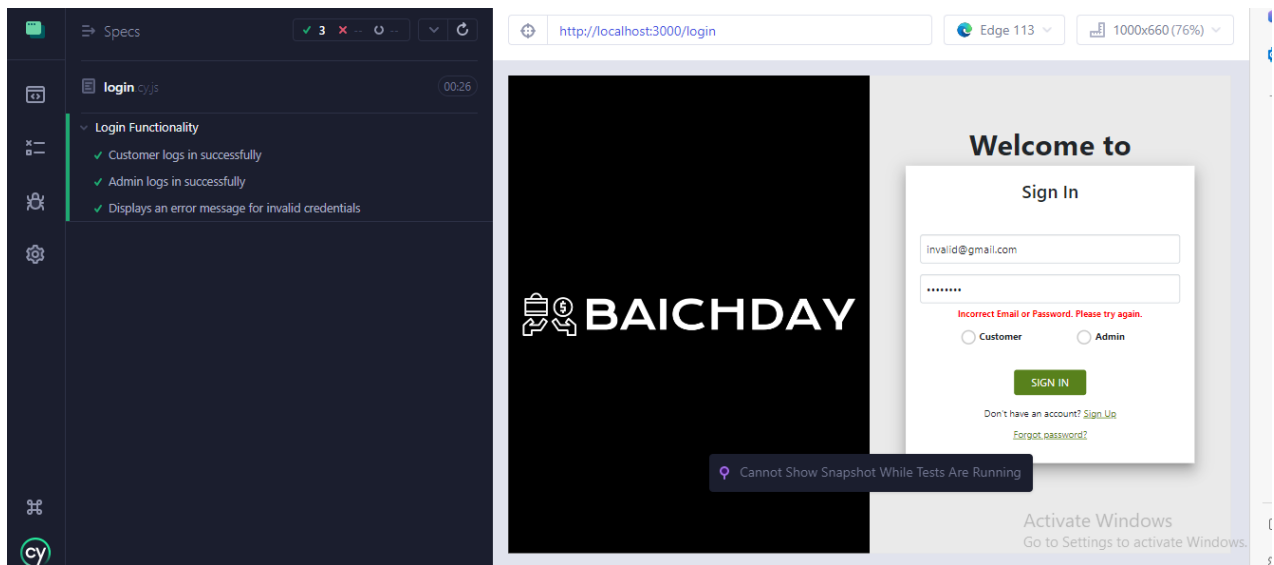
1. Add product



2. Edit Profile



3. User login



Manual Testing:

1. Edit Profile

Test Case ID	T02	Use Case Name	Edit Profile
Test Created by	Silal Anwar Chattha	Test executed by	Silal Anwar Chattha
Test Case Priority	High	Test Case Objectives	Check the functionality of edit profile use case
Test browser	Browser: Google Chrome etc.		

r/platform	
Pre-conditions	User is logged-in and on the profile page.
Post-conditions	The user details are updated according to the users wishes.

Step No	User actions	Inputs	System response	Expected Outputs	Actual Output	Test Result (Pass/Fail)	Comments
1	The user chooses to edit their profile and fills the fields they want to edit.	Arslan, Tarar, password123, 03214404024, 310-C Street 18 Phase 10 DHA, Lahore, Pakistan	The filled fields are displayed as the user fills them.	Arslan, Tarar, ***** *, 03214404024, 310-C Street 18 Phase 10 DHA, Lahore, Pakistan	Arslan, Tarar, ***** **, 03214404024, 310-C Street 18 Phase 10 DHA, Lahore, Pakistan	Pass	
2	The user confirms the changes	Click Apply	The profile is edited and	The user is directed to	The user is directed to	Pass	

			new details are stored in the database. The user is directed to the homepage	the homepage	the homepage		
3							
4							
Test Case Execution Result	Passed	Failed	Not Executed				

2. Add product

Test Case ID	T03	Use Case Name	Add Product
Test Created by	Moiz Nafey	Test executed by	Moiz Nafey
Test Case Priority	High	Test Case Objectives	Check the functionality of the add product use case
Test browser/platform	Borwser: Google Chrome etc.		
Pre-conditions	The user is logged-in and wants to auction an item.		
Post-conditions	Item is uploaded for auction.		

Step No	User actions	Inputs	System response	Expected Outputs	Actual Output	Test Result (Pass/Fail)	Comments
---------	--------------	--------	-----------------	------------------	---------------	-------------------------	----------

1	The user chooses to upload an item for auction.	Click Sell with us	The website takes user to the add product page.	The add product page is displayed to the user where all the required fields are visible.	The add product page is displayed to the user where all the required fields are visible.	Pass	
2	The user fills the fields with the product details.	<u>Product Name,</u> <u>Product image,</u> <u>Product cost,</u> <u>Auction Time,</u> <u>Product description,</u> <u>Product Category</u>	The filled fields are displayed as the user fills them.	Product Name, Product image, Product cost, Auction Time, Product description, Product Category	Product Name, Product image, Product cost, Auction Time, Product description, Product Category	Pass	
3	The user clicks on Add Product button to upload the item.	Click Add Product	The item is uploaded for auction.	The item is uploaded for auction.	The item is uploaded for auction	Pass	
4							

Test Case Execution Result		<i>Failed</i>	<i>Not Executed</i>				

3. Place Bid

Test Case ID	T04	Use Case Name	Place Bid
Test Create d by	Mahad Mubashir Beg	Test executed by	Mahad Mubashir Beg
Test Case Priority	High	Test Case Objectives	Bid on Product
Test browser/platform	Browser: Google Chrome etc.		
Pre-conditions	User is logged-in and wants to bid on a product.		
Post-conditions	Users bid is placed on the product.		

Step No	User actions	Inputs	System response	Expected Outputs	Actual Output	Test Result	Comments
---------	--------------	--------	-----------------	------------------	---------------	-------------	----------

						(Pass /Fail)	
1	The user clicks on the product he wants to place a bid on.	Click product card	The user is taken to the product page.	The product page is displayed.	Pass		
2	The user places a bid on the product.	Enter Bid amount and click on Place Bid	The bid is placed on the product and user is redirected to the home page.	The bid is successfully placed and user is redirected to the homepage.	Pass		
3							
4							
Test Case Execution Result		Passed	Failed	Not Executed			

11. Deployment Guidelines

List down the steps for deployment of your system. Start from where the code (link of the github repository) should be picked and then mention all the steps for deployment in a production environment. Also mention the online link where your application is hosted along with access information (user/password etc.).

Link for Front-End Github Repository: <https://github.com/nafey7/BaichDay-Frontend.git>

Link for Back-End Github Repository: <https://github.com/nafey7/BaichDay-Backend.git>

Clone these repositories to your local machine. Now for each deployments, steps are provided below:

Front-End deployment:

- Run this command in the terminal of the folder in which the github repo is cloned in: “npm i (install all the packages on the local machine).
- Run this command in the terminal of the folder in which the github repo is cloned in: “npm run build” (a build of the code base is made which is ready to be deployed).
- Create an account on Netlify.
- Choose the Git provider where your React app's source code is stored. Netlify supports Git providers such as GitHub, GitLab, and Bitbucket.
- Select the repository where your React app's source code is located, and grant Netlify access to it if necessary.
- Configure the build settings for your React app. Under the "Build settings" section, set the following:
 - a) Build command: npm run build
 - b) Publish directory: build
- Click the "Deploy site" button to start the deployment process. Netlify will automatically build and deploy your React app to a unique URL.
- Once the deployment is complete, you can preview your React app by clicking on the URL provided by Netlify.

Back-End deployment:

- Make sure you have a Heroku account and the Heroku CLI (Command Line Interface) installed on your machine.
- Include this line “start: "node server.js"” in package.json
- Run this command in the terminal of the folder in which the github repo is cloned in: “heroku login” (provide credentials to login to heroku account from terminal).
- Run this command in the terminal of the folder in which the github repo is cloned in: “heroku create” (creating a heroku app through terminal).
- Push the code to the github repository.
- Run this command in the terminal of the folder in which the github repo is cloned in: “git push heroku master” (the updated code is then deployed on heroku).
- Run this command in the terminal of the folder in which the github repo is cloned in: “heroku open” (open the link on which the Back-End is deployed).
- Variables present inside file “config.env” are to be set manually for Heroku inside their application except for the PORT variable.

Link of the application: <https://baichdaypakistan.netlify.app/>

Credentials:

- email: waqarzaka@gmail.com
- password: abcd1234

12. Conclusion

a. Summary

We used MERN Stack to develop an auction site application. We utilized ReactJS to create a dynamic and user-friendly frontend, NodeJS and ExpressJS for the backend, and MongoDB for the database. Through this project, we have learned about the benefits and limitations of using the MERN stack, including its flexibility and scalability, as well as its challenges, such as managing dependencies and debugging errors. We have gained a deeper understanding of the importance of collaboration and communication in software development, as well as the significance of testing and troubleshooting in ensuring the quality of the final product. We look forward to continuing our exploration of the MERN stack and further improving our expertise in web development.

b. Challenges

During the development of this software project using MERN Stack, we encountered several technical and non-technical challenges that we had to address to ensure the success of the project.

One of the technical challenges was managing the dependencies between the different components of the MERN stack. We had to make sure that all the versions of the libraries and frameworks we used were compatible with each other, and any updates were carefully tested before implementation to avoid compatibility issues. Additionally, we had to optimize the performance of the application to ensure that it was responsive and fast, especially when dealing with large datasets.

Another technical challenge we faced was debugging and testing the application. With the complex structure of the MERN stack, it was difficult to identify and resolve errors, especially when there were multiple developers working on the project. We had to implement a rigorous testing process, including unit testing, integration testing, and end-to-end testing, to detect and fix issues as early as possible.

Non-technical challenges included communication and collaboration among team members, managing project timelines and milestones, and dealing with unforeseen circumstances. Communication among team members was crucial to ensure that everyone was on the same page regarding the project's progress and the tasks assigned to them. We had to establish a clear timeline and set realistic deadlines to ensure that the project was completed on time. Additionally, we had to be flexible and adapt to changes in the project scope or requirements, especially when dealing with unexpected issues. To address these challenges, we implemented several strategies, such as regular team meetings, version control using Git, and continuous integration and delivery.

Overall, these challenges taught us the importance of teamwork, communication, and adaptability in software development. By implementing these strategies and addressing these challenges, we were able to successfully complete the project and develop a high-quality web application using the MERN stack.

c. Future

The auction site application we have developed using the MERN stack can be extended in several ways in the future to improve its functionality and usability. Here are some ideas for future extensions:

1. Adding additional features: One way to extend the application would be to add new features to enhance its functionality. For example, we could add a notification system to inform users when they have been outbid.
2. Improving the user interface: We could also improve the user interface by redesigning it to make it more user-friendly and visually appealing. We could also add new themes and customization options to allow users to personalize their experience on the platform.
3. Introducing new payment options: We could also integrate new payment options to make it easier for users to pay for their bids. For example, we could add support for cryptocurrency amount to be added to the users wallet or make a new wallet for cryptocurrencies.
4. Enhancing security: To improve security, we could add additional authentication options, such as two-factor authentication, to prevent unauthorized access to user accounts. We could also implement SSL encryption to protect user data during transmission.
5. Scaling the application: As the user base grows, we may need to scale the application to handle increased traffic and maintain its performance. We could achieve this by adding more servers or utilizing cloud-based hosting services like Amazon Web Services or Google Cloud.

In summary, there are several ways in which the auction site application can be extended in the future to improve its functionality, usability, and performance. By implementing these extensions, we can ensure that the platform remains competitive and meets the evolving needs of its users.

13. Review checklist

Before submission of this report, the team must perform an internal review. Each team member will review one or more sections of the deliverable.

Chapter/Section Name	Reviewer Name(s)
6,7	Mahad Mubashir Beg
1,10,11	Moiz Nafey
2,3,4	Arslan Tarar
8,9	Silal Anwar Chattha
5,12	Nashit Iftikhar

14. References

none