



FlipShop – A Marketplace for buying and selling digital assets.

Final Year Project Report

By

Areej Razzaq

Huda Fatima

In Partial Fulfillment

Of the Requirements for the degree

Bachelor of Engineering in Software Engineering (BESE)

School of Electrical Engineering and Computer Science

National University of Sciences and Technology

Islamabad, Pakistan

(2023)

## DECLARATION

We hereby declare that this project report entitled “FLIPSHOP – MARKETPLACE FOR BUYING AND SELLING OF DIGITAL ASSETS” submitted to the “SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE”, is a record of an original work done by us under the guidance of Supervisor “DR ADNAN RASHID” and that no part has been plagiarized without citations. Also, this project work is submitted in the partial fulfillment of the requirements for the degree of Bachelor of Engineering in Software Engineering.

### Team Members

Huda Fatima

Areej Razzaq

**Signature**

---

---

### Supervisor

Dr Adnan Rashid

**Signature**

---

**Date:**

---

**Place:**

---

## **ACKNOWLEDGEMENTS**

We would like to extend our heartfelt gratitude to our families for their unwavering support throughout this project. Their encouragement, understanding, and belief in our abilities were invaluable. We would also like to express our appreciation to our dear friend Moaaz Tameer Islam for his invaluable assistance and support. His technical expertise, insightful discussions, and willingness to help were instrumental in the success of our project. We are truly grateful for his friendship and contributions.

# **ABSTRACT**

The Flip Shop project is an automated solution that aims to enhance the user experience and transparency in buying and selling digital assets. With the increasing number of internet users in Pakistan and the growing blogger and SME communities, Flip Shop seized the opportunity to create a tailored solution for the local market. Acting as a broker between buyers and sellers, Flip Shop establishes a reliable marketplace and automates the asset evaluation process using a machine learning model. The platform ensures transparent transactions and effectively handles disputes, providing a smooth and secure experience for all parties involved. Flip Shop's success lies in its ability to cater to the evolving digital asset market while prioritizing user-friendliness and trustworthiness.

# Table of Contents

Abstract .....	4
Chapter # 1.....	8
Introduction .....	8
1.1 Background .....	8
1.2 Challenges to the Evaluation of E-Business .....	9
1.3 Literature Review .....	9
1.4 Disadvantages of the available solutions .....	10
1.5 Proposed Solution.....	10
1.6 Stakeholders .....	10
1.7 Technology Stack.....	11
Chapter # 2.....	12
Software Requirement Specification .....	12
Functional Requirements.....	12
2.1 Register the user as a buyer or seller .....	12
2.2 Maintain user profile. ....	12
2.3 Login to the system .....	13
2.4 Accept the asset information, predict the price and enlist it .....	13
2.5 Verify User.....	14
2.6 Filter Search Results.....	15
2.7 Show interest in the asset.....	15
2.8 Make an offer on asset.....	15
2.9 Place bid on asset. ....	16
2.10 Accept/Reject Offer.....	16
2.11 Close Auction.....	17
2.12 Make and Send Contract.....	17
2.13 Accept/Reject Contract.....	18
2.14 Make Payment.....	18
2.15 Send Credentials.....	19
2.16 Send Credentials.....	19

2.17	Manage Disputes .....	20
2.18	Review Listings.....	21
2.19	Block User .....	21
2.20	Contact Us.....	22
	Other Nonfunctional Requirements .....	22
2.21	Performance Requirements .....	22
2.22	Safety Requirements.....	23
2.23	Security Requirements.....	23
2.24	Software Quality Attributes .....	23
	Chapter # 3.....	25
	Software Design Specifications .....	25
3.1	Architectural Style.....	25
3.2	Design Methodology .....	25
3.3	Data Representation Diagram .....	25
3.4	Process Flow .....	28
3.4.1	Use Case # 1 – Ideal Case .....	28
3.4.2	Use Case # 2 – Disputed Case .....	29
3.4.3	Use Case # 3 – Block User by Admin .....	30
3.4.4	Use Case # 4 – Price Prediction Blog .....	31
3.4.5	Use Case # 5 – Price Prediction Daraz.....	31
3.4.6	Use Case # 6 – Logical Implementation .....	32
3.4.7	Use Case # 7 – Cron Jobs .....	33
	Chapter # 4.....	34
	System Testing.....	34
4.1	Functional Testing .....	34
4.2	Non-Functional Testing: .....	35
	Chapter # 5.....	39
	Deployment and system integration .....	39
5.1	Hosting Plan, Domain and Subdomain.....	39
5.2	Backend .....	39
5.3	Databases .....	39

5.1	Frontend .....	39
5.2	FTP Server for Assets.....	39
	Chapter # 6.....	40
	User Interfaces .....	40
	Chapter # 7.....	45
	RESULTS AND DISCUSSION.....	45
	Chapter # 8.....	46
	Conclusion and Future Work.....	46
	REFERENCES.....	47

## List of Figures

Figure 1: Source: Statista .....	8
Figure 2: User Selling Daraz Account (1).....	9
Figure 3: User Selling Daraz Account (2).....	9
Figure 4- Architectural Style .....	25
Figure 5- ERD Diagram .....	27
Figure 6- Ideal Case .....	28
Figure 7- Disputed Case.....	29
Figure 8- Block User.....	30
Figure 9 - Price Prediction - Blog .....	31
Figure 10 - Price Prediction - Daraz .....	31
Figure 11- Data Response .....	32
Figure 12- Logical Implementation .....	32
Figure 13- Cron.....	33
Figure 14- Deployment Structure .....	39
Figure 15 - UI Figure 1 .....	40
Figure 16- UI Figure – Listings .....	41
Figure 17 - UI Figure - Services Page.....	41
Figure 18- UI Figure - Stats.....	42
Figure 19 - UI Figure - Testimonials .....	42
Figure 20 - UI Figure - Team .....	43
Figure 21 - UI Figure – FAQ.....	43
Figure 22- Seller Dashboard.....	44
Figure 23- User Profile.....	44

# INTRODUCTION

## 1.1 Background

As internet access and adoption are rapidly increasing worldwide, the number of digital buyers keeps climbing every year. According to a report by Forbes, Covid-19 Accelerated E-Commerce Growth 4 To 6 Years (Forbes, 2020). In 2020, over two billion people purchased goods or services online, and during the same year, e-retail sales surpassed 4.2 trillion U.S. dollars worldwide. With the Covid19 pandemic, more consumers are going online to buy everything from groceries to luxury items. As a result, Statista estimates that there will be 2.14 billion global buyers online by the end of 2021.



*Figure 1: Source: Statista*

It's much easier to renovate a single room than to build an entire house from scratch. Starting a business from the ground up is hard for similar reasons. Putting in long hours and not seeing immediate results can be discouraging. An existing online business that's making money is an attractive prospect for many investors. Similar to the physical world, where evaluations are made for business, the evaluation of online businesses are made which are then sold to potential buyers.



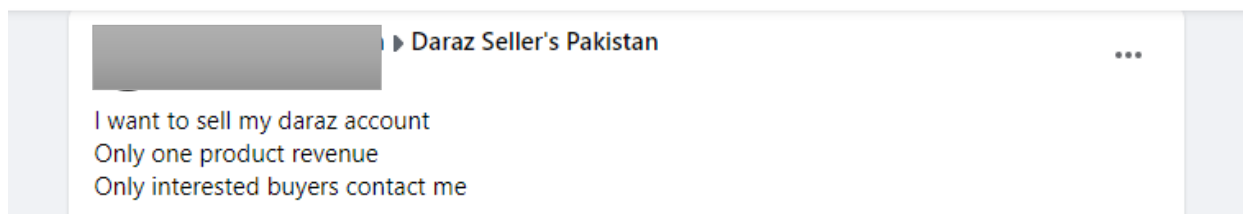
## 1.2 Challenges to the Evaluation of E-Business

Evaluating an online business is not same as assessing a physical one. Reflecting on the issue, the main challenges to deriving a fair business valuation are as follows:

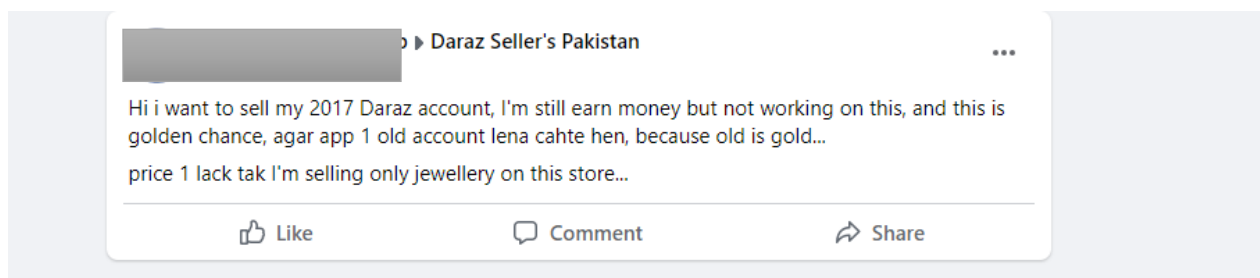
- Misunderstanding or bad use of valuation techniques
- Gathering or using the wrong information for inclusion in the analysis
- Lack of automation in the process

## 1.3 Literature Review

There exist a number of solutions for the said problem but none of them is defined for the Pakistani marketplace. [Flippa](#), [Exchange Marketplace](#), [Empire flippers](#) are some of the internationally available solutions. [Flipit.pk](#) is a solution available to Pakistani marketplace. In this literature review, Flippa (the most renowned international marketplace) and Flipit.pk are critically analyzed. In Pakistan, the buying and selling of online business is taking place on informal platforms such as Facebook Groups, Ads and word of mouth. Following pictures are taken from Facebook Groups that are being posted as an advertisement for the Online Business.



*Figure 2: User Selling Daraz Account (1)*



*Figure 3: User Selling Daraz Account (2)*

No such platform where all buyers and sellers' needs are met is in existence.

### Flippa

Flippa is a marketplace for buying and selling online businesses, based in Melbourne, Australia, and San Francisco, U.S. It was founded by Flippa was founded by Mark Harbottle and Matt Mickiewicz in 2009. It had traded more than \$400 million in websites, domains, and mobile apps (Flippa). It includes a wide variety of categories ranging from websites, domains, apps and more.

## **Flipit.pk**

Flipit.pk is the first of its kind startup in Pakistan located at NSTP aiming at selling online business. Flipit.pk is in its initial stages and there is no recorded success available for any transaction via them.

## **1.4 Disadvantages of the available solutions**

### **Flippa**

Flippa is an international marketplace, but it does not satisfy the needs of the local marketplace. Flippa is a great resource for online business transaction for the international community, but it lacks the ability to connect Pakistani buyers to Pakistani sellers. Since the demographic of the audience makes a huge impact in finding the potential buyers and making the right evaluations, Flippa does not sit right with the demands of the local community.

### **Flipit.pk**

Flipit.pk aims to solve the needs of the local market. While it targets both physical assets and online business, it leaves the focus divided between the two. No automation is provided by the Flipit.pk as each listing is handpicked by them and published by the startup every week. On analyzing Flipit.pk, unrealistic evaluations are being listed on their websites. The available listings lack analytics for the assessment of the listing. Along with this, one drawback of Flipit.pk is that the buyer has to directly contact the seller to assess its business and a member from their team will act as a broker between the two. These loopholes make the system prone to scams and hard for potential buyers to find what they need.

## **1.5 Proposed Solution**

With the problem in hand, our group will solve the problem by developing “Flip Shop”, which is a dedicated platform for the buying and selling of online business in Pakistan. In our project, we propose Flip Shop that aims to provide a marketplace where people can buy and sell online business, such as Daraz Seller Accounts and Blogs. Flip Shop will act as a broker between both buyer and seller. It will ensure the smooth transaction of business between both parties. With Flip Shop, we aim to automate the evaluation process by using the analytics provided for the business by the sellers. Flip Shop will ensure transparency during the transaction.

## **1.6 Stakeholders**

The following are the stakeholders of FlipShop.

- Sellers – Users of the system who will list assets on platform and make sales through them.

- Buyers – Users of the system who will purchase the listed assets through the platform.
- Admin – Moderator of the system who will manage certain activities.

## **1.7 Technology Stack**

Flip Shop is divided into three tiers dividing the responsibilities with tiers. The presentation tier will be developed in React while backend is to be developed via Laravel. The database is managed and implemented in MySQL workbench. Communication between these tiers is done by following protocols. The presentation and logical tiers are connected via request response cycle following the REST API conventions. Additionally, the logic tier communicates with the data tier using the queries and commands made by the logic layer.

## **SOFTWARE REQUIREMENT SPECIFICATION**

This section of the document lists the features that Flip Shop will be occupied with.

### **Functional Requirements**

#### **2.1 Register the user as a buyer or seller**

##### **2.1.1 Description and Priority**

Flip Shop will enable the users to register themselves to the platform. This feature has the highest priority.

##### **2.1.2 Stimulus/Response Sequences**

Once the user registers at Flip Shop, the database should be updated. After successful registration, the user should be directed to the login screen.

##### **2.1.3 Functional Requirements**

- The user would select the role of seller or buyer for itself.
- The user would enter its name.
- The user would enter their valid email.
- The user would enter password.
- The user would confirm password.
- The user would be able to submit the registration form.
- The password should be hashed before storing into the database.
- The database should be updated with the entered credentials.
- The user would be redirected to the login screen.
- The field should display the error if any of the constraints failed.

#### **2.2 Maintain user profile.**

##### **2.2.1 Description and Priority**

Each user of the Flip Shop will have a profile associated with it. This feature has a medium priority.

##### **2.2.2 Stimulus/Response Sequences**

Once the user is registered, a profile should be created for the user with the credentials of the user. The user should be able to enter additional information into its profile. Additionally, the user should be able to edit the previously entered information.

### **2.2.3 Functional Requirements**

- Each user would have only one profile.
- The user would be able to upload a profile picture.
- The user should be able to add a description of themselves.
- The user should be able to see previous purchases.
- The user should be able to see his listings.
- The user should be able to edit the profile.
- The user should be able to save the changes.

## **2.3 Login to the system**

### **2.3.1 Description and Priority**

Flip Shop will enable the users to login to the system using either google oath or password-based authentication. This feature has the high priority.

### **2.3.2 Stimulus/Response Sequences**

When the user tries to login at Flip Shop, the credentials should be verified from the database. After successful login, the user should be directed to the home screen.

### **Functional Requirements**

- To login, user should be registered in the system.
- Users shall only be able to login with valid user name and password.
- The system shall authenticate login credentials (username and password).
- If the user's login credentials fail the system shall display a message to the user indicating that authentication has failed.
- If the user's login credentials fail, the system shall show an error message.
- After a successful login, the system shall manage a user's sessions.
- Session data that will be stored in the user's cookies will be: First Name, Last Name, User ID.
- Upon successful login, the system will redisplay the current page to the user with the new privileges.

## **2.4 Accept the asset information, predict the price and enlist it**

### **2.4.1 Description and Priority**

The system shall take the asset information from the seller, predict the price based of the asset data and enlist it for sale. This feature has a high priority.

### **2.4.2 Stimulus/Response Sequences**

When the seller enters the assets information the system should be able to calculate an estimated price for the asset. After the seller's approval, the database should be updated, and the asset should be available for sale on the website.

### **2.4.3 Functional Requirements**

For accepting the asset information:

- The seller should be able to provide preliminary details like name and description of the asset.
- The seller should be able to state the asset type i.e., Daraz Seller Account or Blog.
- The seller should be able to state the industry the seller operates in.
- The seller should be able to state the monetization methods of asset.
- The seller should be able to enter the expenses and revenue of the assets.
- The seller should be able to attach a profit and loss (P&L) statement of the asset.
- The seller should be able to enter a tagline of 120 characters about the asset.
- The seller should be able to choose between an auction and fixed-price listing.

For predicting the price and enlisting the asset:

- If the listed asset is a blog, the trained ML model should be called using API to predict the price of the asset.
- The added information about the asset will be used to calculate the price of the asset.
- If the listed asset is a daraz seller account, the asset will be evaluated by taking the product of the account's monthly profit and the months he/she wants the evaluation against.

## **2.5 Verify User**

### **2.5.1 Description and Priority**

Flip Shop should be able to validate the user and assign a batch to the user. This feature has the medium priority. Only verified users can create asset and make offers or place bids on assets.

### **2.5.2 Stimulus/Response Sequences**

Once the user has registered, it can verify the account, by verifying the code sent to user via email.

### **2.5.3 Functional Requirements**

- A code should be generated and stored against the user.
- The system should be able to send email containing the code upon user request.
- User should be notified once the email is sent.
- If the email sending fails, user should be notified accordingly.
- If the code entered by the user matches the sent code, the user should be verified.
- If the code entered by the user does not matches the sent code, the user should be notified of the invalid code.

## **2.6 Filter Search Results**

### **2.6.1 Description and Priority**

The user should be able to search the asset from the enlisted list based on some parameters. This feature has the medium priority.

### **2.6.2 Stimulus/Response Sequences**

The system should find the assets that satisfy the search parameters entered by the user and display the filtered assets on screen.

### **2.6.3 Functional Requirements**

- The user must be able to search for the assets based on the asset characteristics such as price and type etc.
- When the user searches for the asset, it is shown in search results. If the search is broad, then all the assets that satisfy the search parameters are loaded. On the other hand, if the no asset satisfies the perimeter, then the system should load 'no assets matched the search criteria.

## **2.7 Show interest in the asset.**

### **2.7.1 Description and Priority**

If the buyer wants to save any of the interested asset, he should be able to show interest in the asset. This feature has the low priority.

### **2.7.2 Stimulus/Response Sequences**

Once the buyer has shown interest in the asset, the seller must be notified with new customers. channels.

### **2.7.3 Functional Requirements**

- The user must be able to show interest by clicking on the “Show Interest” button.
- Once the buyer has shown interest in an asset, the system must be able to notify the seller.
- The seller should be able to view the profile of interested buyer.
- The buyer should be able to view the profile of the business owner.

## **2.8 Make an offer on asset.**

### **2.8.1 Description and Priority**

The buyer should be able to make an offer on fixed price asset.

### **2.8.2 Stimulus/Response Sequences**

Once the buyer clicks on the make offer button, a modal should be opened asking for the offer's amount. Once amount is submitted, the backend should store the offer information with the related asset and buyer.

### **2.8.3 Functional Requirements**

- The user must be able to make an offer by clicking on the “Make Offer” button.
- Once the buyer has made an offer on an asset, the offer should be visible to seller with its asset details.
- The buyer should be able to view all the offers he has made.
- The buyer should be able to edit the offer unless it has been accepted or rejected by the seller.
- The buyer should be able to delete the offer unless it has been accepted or rejected by seller.

## **2.9 Place bid on asset.**

### **2.9.1 Description and Priority**

The buyer should be able to make an place bid on auction based asset.

### **2.9.2 Stimulus/Response Sequences**

Once the buyer clicks on the place bid button, a modal should be opened asking for the offer's quotation. Once the quotation is submitted, the backend should store the bid information with the related asset and buyer.

### **2.9.3 Functional Requirements**

- The buyer must be able to make a bid by clicking on the “Place Bid” button.
- Once the buyer has made a bid on an asset, the offer should be visible to the seller with its asset details.
- The buyer should be able to view all the bids he has placed.

## **2.10 Accept/Reject Offer.**

### **2.10.1 Description and Priority**

The seller should be able to accept or reject any offer.

### **2.10.2 Stimulus/Response Sequences**

Once the seller accepts any of the offers that has been on an asset, the offer status should be changed to accepted. Additionally, all other offers that have been made on the asset should be rejected. Once offer statuses are managed, all the relevant buyers should receive notification via email.



### **2.10.3 Functional Requirements**

- The seller must be able accept or reject any of the offers by clicking the button against each offer.
- If an offer is accepted, the buyer should be notified by email. The rest of the buyers should be notified for the offer rejection.
- A notification status should be recorded for each email sent so no buyer is notified more than once. For instance, the seller rejected offer for buyer A, then accepted offer for buyer B. the email to buyer A should be sent only once, not twice.

## **2.11 Close Auction**

### **2.11.1 Description and Priority**

Once the auction period ends, the highest bid should be chosen as the closed bid.

### **2.11.2 Stimulus/Response Sequences**

A cron job should be run those measures if today is closing date for any of auction. If any of the auction has to be close today, the highest quotation is selected as the closed bid.

### **2.11.3 Functional Requirements**

- System should correctly filter the closing auction assets.
- The highest quotation should be selected from the placed bids.
- The quotation winner should receive notification via email.
- The seller should be notified with the closed bid quotation.

## **2.12 Make and Send Contract**

### **2.12.1 Description and Priority**

Once either of the offer acceptance of bid closing is done, the seller should be promoted to make contract. This has highest priority.

### **2.12.2 Stimulus/Response Sequences**

The seller should be able to make contract to transfer asset and send it to buyer.

### **2.12.3 Functional Requirements**

- The seller should be able make contract by providing open and close date for the contract period.
- Once the seller defines the contract period, he should be able to view the contract in pdf format with all the conditions defined for the asset migration.

## **2.13 Accept/Reject Contract**

### **2.13.1 Description and Priority**

Once the buyer, receives the contract, he can accept or reject the contract.

### **2.13.2 Stimulus/Response Sequences**

The buyer should be able to change the status of contract. If the contract gets accepted, migration period should start. Otherwise, the asset should move to new status with the contract being cancelled.

### **2.13.3 Functional Requirements**

- The buyer can accept the contract by clicking on the accept button.
- Once accepted, the migration period starts, and a transaction should be created against the contract.
- If the buyer rejects the contract, by clicking on the reject button, the contract should be aborted.
- Once the contract is aborted, the asset should move to “new” status and should be able to accept new offers or bids.

## **2.14 Make Payment**

### **2.14.1 Description and Priority**

Once the contract is accepted, a transaction is created for both seller and buyer. The buyer has to make payment within the contract period.

### **2.14.2 Stimulus/Response Sequences**

The buyer should be able to make payment within the contract period. If payment is not made, the contract should be aborted with the asset being moved to new status.

### **2.14.3 Functional Requirements**

- The buyer should be able to make payment by going to the transactions section and selecting the contract against which the payment has to be made.
- The buyer should be able to add his card details including card number, card holder name, cvc, expiry date to transfer the payment to the system.
- Once buyer makes the payment, the payment should remain with the system, until seller sends the credentials and buyer approves them.

- If the buyer fails to make payment within the contract period, the contract should be aborted. Once the contract is aborted, the asset should move to “new” status and should be able to accept new offers or bids.

## **2.15 Send Credentials**

### **2.15.1 Description and Priority**

Once the buyer makes payment, the seller should be able to send credentials to buyer.

### **2.15.2 Stimulus/Response Sequences**

The seller should be able to send credentials within the contract period once the buyer has made payment. If seller does not transfer credentials within the period, the contract should be aborted with the asset being moved to new status. Additionally, the payment will be reverted to the buyer.

### **2.15.3 Functional Requirements**

- The seller should be able to send credentials by going to the transactions section and selecting the contract against which the credentials has to be sent.
- The seller should be able to add credentials in the provided text box.
- Once the seller sends the credentials, the buyer should be prompted to accept or reject the credentials.
- If the seller fails to send credentials within the contract period, the contract should be aborted. Once the contract is aborted, the asset should move to “new” status and should be able to accept new offers or bids. Additionally, the payment should be reverted to the buyer.

## **2.16 Send Credentials**

### **2.16.1 Description and Priority**

Once the seller sends credentials, the buyer can approve or reject credentials. This has highest priority.

### **2.16.2 Stimulus/Response Sequences**

Once the seller sends credentials within the contract, the buyer can thus approve or reject the provided credentials. If accepted, the asset will be marked as sold and the transaction will be completed. If the buyer rejects the credentials, a dispute will be opened against the transaction.

### **2.16.3 Functional Requirements**

Once the buyer receives the credentials, if the buyer approves credentials, the contract and the transaction will be completed.

If approved, the asset will be marked as sold.

If approved, payment will be released to the seller.

Once the buyer receives the credentials, if the buyer rejects credentials, contract and the transaction will be in dispute.

If rejected, payment will remain with the system.

A dispute will be recorded against the transaction.

## **2.17 Manage Disputes**

### **2.17.1 Description and Priority**

Once credentials are rejected by the buyer, a dispute should be recorded. The buyer and seller will be presented with two decisions as “Pay Buyer” and “Pay Seller” to choose as one of the option to resolve dispute.

### **2.17.2 Stimulus/Response Sequences**

Both the users should be able to solve the dispute as per their liking. If both users reach a mutual decision, the relevant transaction will be processed. Otherwise, admin will be automatically authorized to intervene the dispute.

### **2.17.3 Functional Requirements**

- Once a dispute is created, both parties should be presented with the options of “Pay Buyer” and “Pay Seller”.
- If both parties select “Pay Seller”, the contract and transaction will be completed. The involved asset will be marked as sold. Additionally, payment will be released to the seller.
- If both parties select “Pay Buyer”, the contract and transaction will be aborted. The asset will be marked as new. Additionally, payment will be reverted to the buyer.
- If both parties select opposing options, admin will be authorized to access the dispute and make decision on the dispute.
- Admin can also select any of the options “Pay Buyer” or “Pay Seller”. The relevant process as defined above will be followed for either of the options.
- In case, any or both of the user does not select any of the options with in one week, admin will be authorized to make decision. Hence, followed by the same process.

## **2.18 Review Listings**

### **2.18.1 Description and Priority**

The assets should be removed to under review if no activity has been performed on the asset by seller within 60 days of its listing date. The seller should also have the option to refresh the asset in order to make the asset visible again. This has low priority.

### **2.18.2 Stimulus/Response Sequences**

A job should be handled at server which filters the asset from ongoing to the assets which need to be placed at under review.

### **2.18.3 Functional Requirements**

- The job should detect the assets that need to be made under assets.
- Under review assets should not be visible to the general listings.
- The seller should have the option to refresh the asset by going to under review assets and selecting the particular asset to refresh.
- Once the asset is refreshed, the created date at the databases should be updated with the current date.

## **2.19 Block User**

### **2.19.1 Description and Priority**

The admin should have the authority to block the user. The has low priority.

### **2.19.2 Stimulus/Response Sequences**

Once the admin blocks the user, the user should not be able to login the system.

### **2.19.3 Functional Requirements**

- The admin should be presented with the list of users. The admin should be able to block any of the users.
- If the user has no in process transaction, the user will be blocked.
- If the user has in process transaction, the user will not be blocked.
- If the user is blocked, a blocked status will be recorded.
- If the user is blocked, the user should not be able to login to the application and relevant message should be displayed to him.
- If any of the users are blocked, the admin should also be able to unblock the user. Consequently, changing the block status in database against the user.

## **2.20 Contact Us**

### **2.20.1 Description and Priority**

The visitor of the application should be able to send a query to admin. All the sent queries should be available to admin. This has low priority.

### **2.20.2 Stimulus/Response Sequences**

The authentication of user is not required for the visitor to send query. Once a query is received an email should be sent to admin. The admin should also be able to mark the queries as read once processed.

### **2.20.3 Functional Requirements**

- The visitor should be able to add name, email, subject and message for the query.
- Once query is sent, email should be sent to admin to the relevant query subject in title.
- Once the admin receives the query, he should be able to see the list of queries.
- After the admin processes the queries at his end, he can mark the queries as read on his portal.
- Additionally, the admin should be able to view the unread and read messages as separate messages.

## **Other Nonfunctional Requirements**

## **2.21 Performance Requirements**

### **2.21.1 Response Time:**

1. Any interface between the application and the user system shall take a maximum response time of 1 minute.
2. The customer service center should analyze problems reported within maximum 3 hours.

### **2.21.2 Workload and Scalability:**

3. The application must accommodate maximum 100 simultaneous users within a peak load time period.
4. The database should be able to perform 10 queries per second.
5. The database performance of the Flip shop should be fast and reliable.

### **2.21.3 Availability:**

6. The system shall be up 24/7 and only have a downtime of 15 minutes every two weeks for maintenance.
7. The customers should be able to make purchases multiple times in a day.

8. The online payment system should be available to customers 24 hours a day.
9. If the system is not operating, the application system shall provide users a notification that service is unavailable.

## **2.22 Safety Requirements**

NA

## **2.23 Security Requirements**

1. The access permissions for system data can only be changed by the systems data administrator of the application system.
2. The application should assure the data protection and privacy of the users.
3. Provider systems should resist unauthorized, accidental or unintended usage and provide access only to legitimate users.
4. The payment system should allow no money to be taken from a user without explicit authorization by that user.
5. Payment transactions should be atomic.
6. Privacy of information, the export of restricted technologies, intellectual property rights, etc. should be audited.

## **2.24 Software Quality Attributes**

### **2.24.1 Usability**

1. Users who have zero training and no understanding of English should be able to easily use the application system.
2. The website shall show different messages to the user showing progress to make it interactive.
3. The system shall be able to handle 200 recognition requests per minute.
4. One user should be able to send one request per second.

### **2.24.2 Reliability:**

5. The update process of the app shall be able to roll back all related updates when any update fails.

### **2.24.3 Scalability**

6. The website limit must be scalable enough to support 200,000 users at a time.
7. Provider systems should be designed to accommodate increased volumes, workloads and users.
8. Features can be enriched. Hence ensuring the evolution.

9. System should be scalable enough to add new asset types without any interruption.
10. Services should be easily modified under the evolution of the system.

#### **2.24.4 Correctness**

11. The system shall predict price with at least 90% accuracy for any kind of asset.
12. The system shall be able to verify asset data with 90% accuracy.

#### **2.24.5 Robustness**

13. The system shall be able to handle incomplete transactions or any other errors and show them to user interactively.

#### **2.24.6 Portability**

14. The system should be portable. So, moving from one operating environment to other does not create any problem.



# SOFTWARE DESIGN SPECIFICATIONS

## 3.1 Architectural Style

The selection of the Model-View-Controller (MVC) architectural style for Flip Shop was a deliberate choice due to its proven success in developing web applications. Flip Shop, a marketplace for digital assets built using Laravel and React, leverages the strengths of the MVC architecture to separate the application's data, user interface, and control logic. This separation of concerns allows for a clear division of responsibilities and promotes maintainability, scalability, and ease of development. The Laravel framework implements the MVC pattern, providing a clear structure for the application's backend logic and data storage. Meanwhile, react is used for the front end, providing a dynamic and responsive user interface.

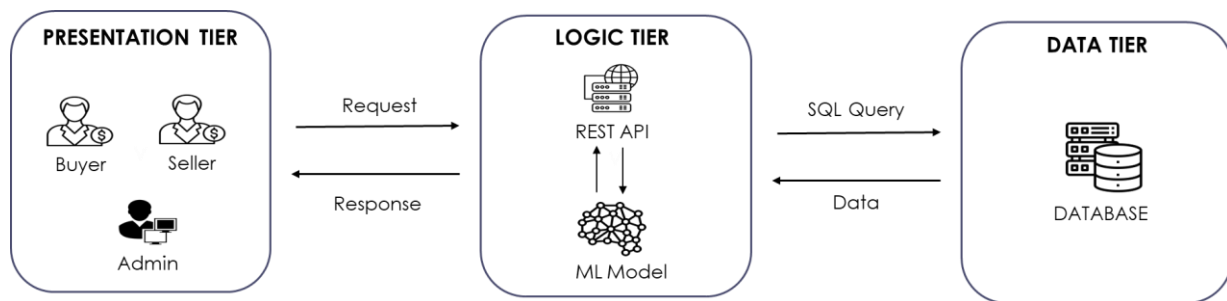


Figure 4- Architectural Style

## 3.2 Design Methodology

Object-Oriented Design (OOD) was chosen as the design methodology for Flip Shop because of its ability to effectively manage complexity in software development. As a marketplace for digital assets, Flip Shop requires a flexible and scalable design that can accommodate a wide range of requirements. OOD provides a well-defined structure for the application, making it easier to manage and maintain. The use of classes and objects in OOD allows for a clear separation of concerns, enabling developers to work on different parts of the application in parallel and reducing the risk of errors. Additionally, OOD encourages the use of reusability, abstraction, and encapsulation, making it easier to reuse code, manage dependencies, and ensure the stability of the application. The application of OOD in Flip Shop results in a well-structured, maintainable, and scalable platform for buying and selling digital assets.

## 3.3 Data Representation Diagram

The use of ERD (Entity Relationship Diagram) in the development of the web application Flip Shop helps to visualize and manage the relationships between entities in the database. By clearly

defining entities and their attributes, and the relationships between them, ERD ensures the accuracy and integrity of the data in the system, making it an essential tool in the development process. The ERD is represented in Figure 4 below.



Figure 5- ERD Diagram

## 3.4 Process Flow

The following listed use cases define the entire system and represent the structure of the web application.

### 3.4.1 Use Case # 1 – Ideal Case

#### 3.4.1.1 Description

This scenario lists the successful selling/buying of the asset with the “fixed” pricing model. The seller lists the asset on the platform while the buyer will make offer on the listed asset. Once the offer is accepted by the seller, the seller must make and send a contract to the buyer by specifying an opening and closing date. Once the contract is sent, the buyer is liable to make payment within the contract period. In this scenario, the buyer makes the payment to the system. Once the seller receives the notification, that system has received the payment, only then seller can send credentials to the buyer. Once the seller sends the credentials and the buyer approves them, the payment will be released from the system to the seller.

#### 3.4.1.2 Sequence Diagram

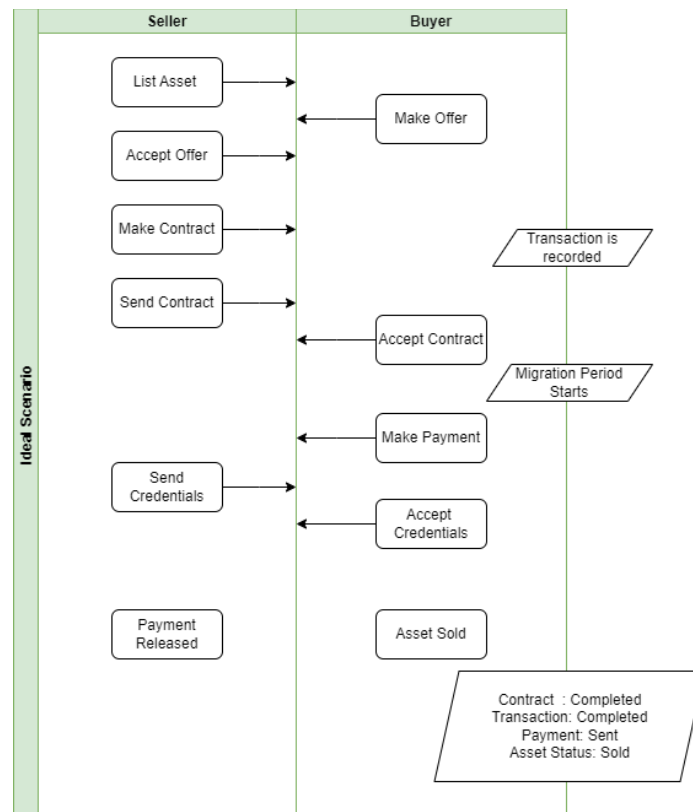


Figure 6- Ideal Case

## 3.4.2 Use Case # 2 – Disputed Case

### 3.4.2.1 Description

This scenario lists the disputed version for the selling process. All the steps from Use Case # 1 will be repeated until the buyer, instead of approving the credentials, will reject them. This will automatically open a dispute among both parties. Both parties will be allowed to select any of the two options as presented in the following diagram: “Pay Seller” or “Pay Buyer”. If both parties select the same option, the dispute will be resolved. If both parties select “Pay Buyer”, it means the contract will be aborted and transaction will be cancelled with the payment reverted to buyer. If both parties select “Pay Seller”, it means the contract and transaction will be completed with payment released from system to seller. However, in the case where neither party does not reach any mutual consensus or within one week, any of the parties do not select any of the provided options, the admin will get authorized to intervene in the dispute. The admin can then select any of the same options to resolve the dispute.

### 3.4.2.2 Sequence Diagram

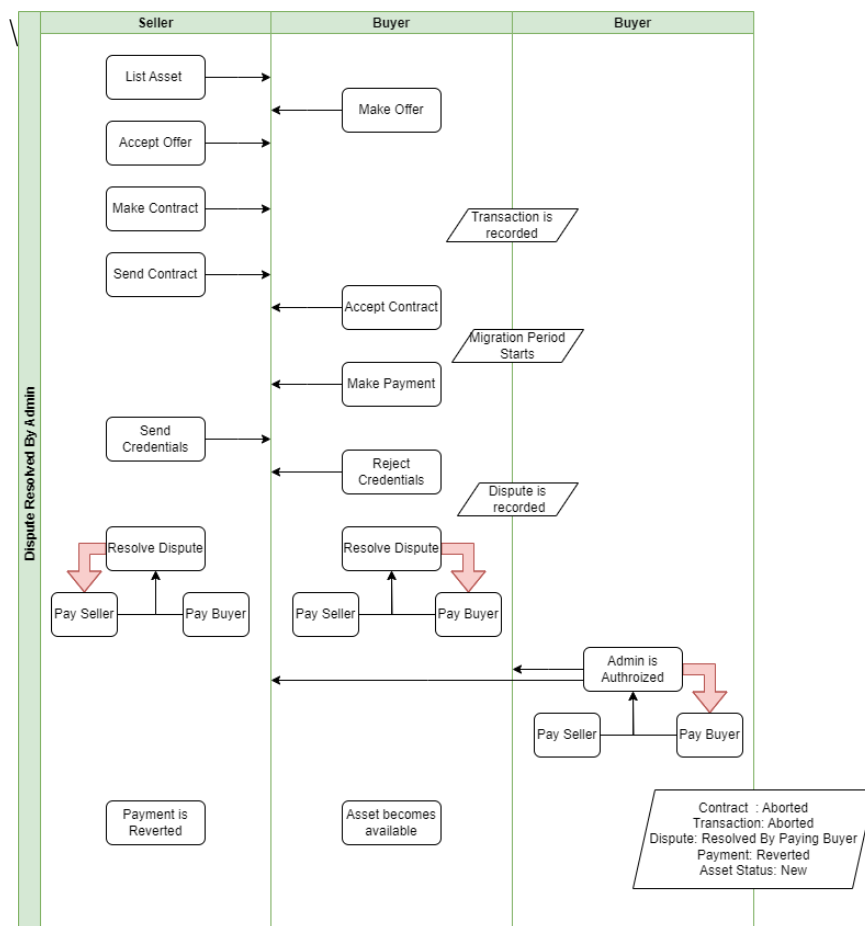


Figure 7- Disputed Case

### 3.4.3 Use Case # 3 – Block User by Admin

#### 3.4.3.1 Description

This scenario displays the authority of admin to block any of the user. However, if the user has any ongoing transaction, the user cannot be blocked.

#### 3.4.3.2 Sequence Diagram

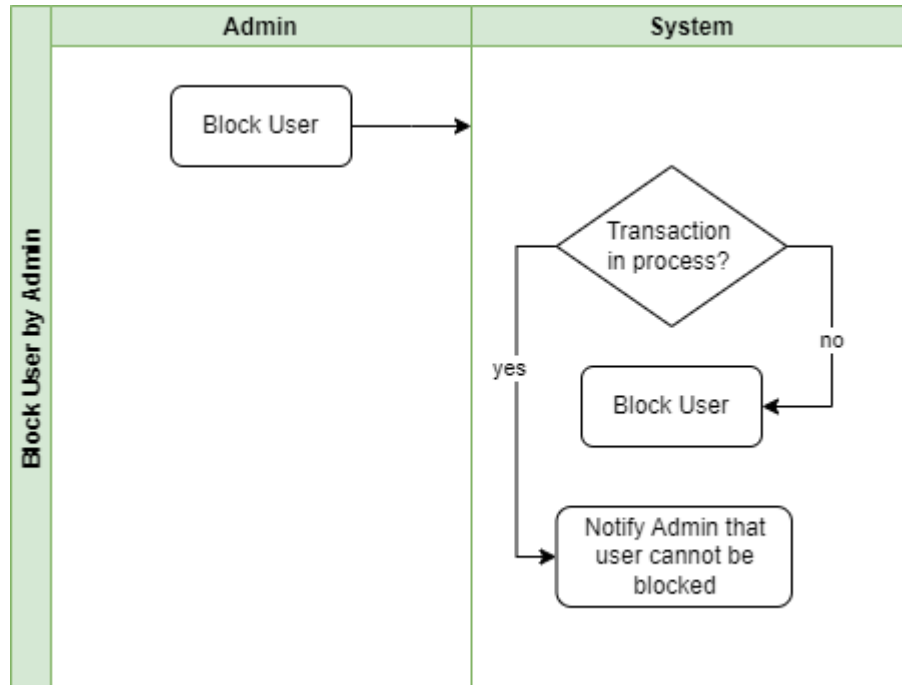


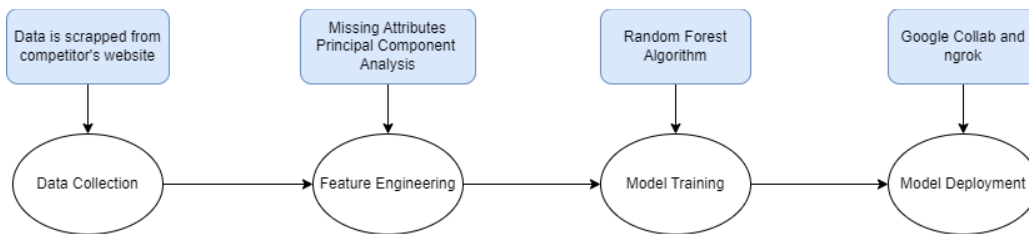
Figure 8- Block User

### 3.4.4 Use Case # 4 – Price Prediction Blog

#### 3.4.4.1 Description

The following diagram illustrates the working of price prediction model for blog.

#### 3.4.4.2 Flow Chart Diagram



*Figure 9 - Price Prediction - Blog*

### 3.4.5 Use Case # 5 – Price Prediction Daraz

#### 3.4.5.1 Description

The following diagram illustrates the working of price prediction model for Daraz account.

#### 3.4.5.2 Flow Chart Diagram



*Figure 10 - Price Prediction - Daraz*

### 3.4.6 Use Case # 6 – Logical Implementation

#### 3.4.6.1 Description

The following diagram illustrates the working of logical implementation at the backend. RESTFUL APIs have been used at the backend to communicate with the frontend. A request (GET, POST, PUT, DELETE) with the user token is sent to server and appropriate response is returned to complete the request/response cycle. Additionally, same structure of response is followed for responses.

```
"success": true,  
"data": [],  
"message": "Offer retrived successfully"
```

Figure 11- Data Response

Moreover, the logical flow is represented in the below flow chart with an example.

#### 3.4.6.2 Flow Chart Diagram

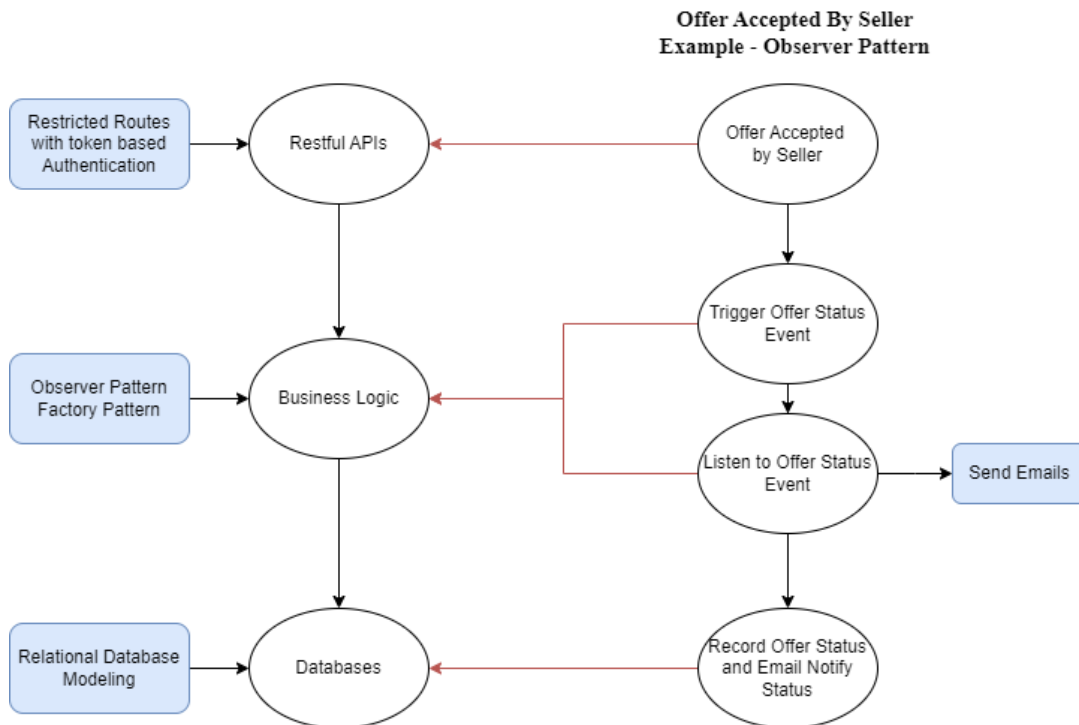


Figure 12- Logical Implementation



### 3.4.7 Use Case # 7 – Cron Jobs

#### 3.4.7.1 Description

This scenario lists the Cron jobs that have been scheduled at the backend. A total of six jobs have been defined where a certain action is performed if a certain condition has been met in a given period of time. For instance, consider the job represented at point # 4, if the buyer does not transfer the payment to the system within the contract period, the contract will be cancelled, and the asset will be moved to a new status to accept any new offers.

#### 3.4.7.2 Flow Chart Diagram

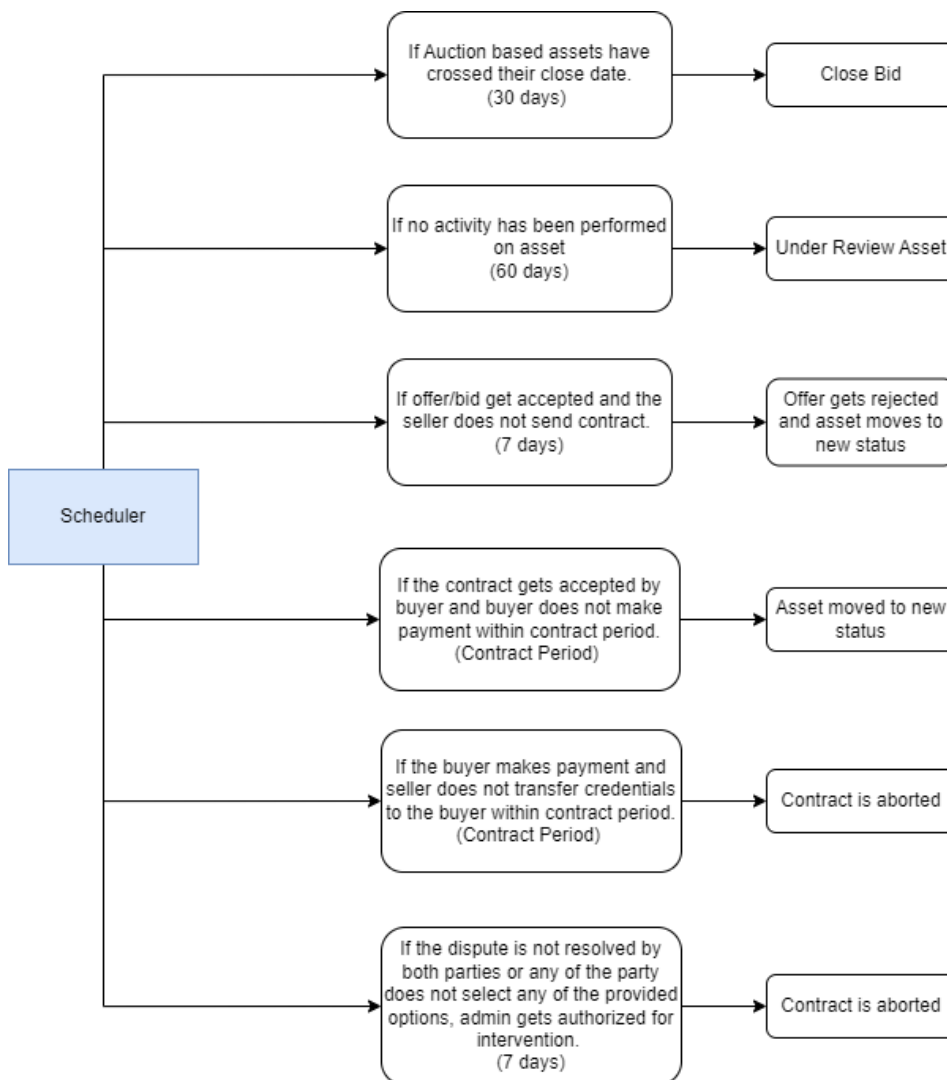


Figure 13- Cron

## **SYSTEM TESTING**

### **4.1 Functional Testing**

During the system development, functional testing was conducted to ensure that the website met the specified requirements and performed as expected. The testing phase involved executing various use cases provided by the client to validate the functionality of different features and components of the website.

The use cases in section 3.4 served as a guideline for the testing team to simulate real-life scenarios and interactions with the website. By following the steps outlined in each use case, the we were able to verify if the website's functionalities were working correctly, identified any issues, and recorded the encountered bugs.

After executing the functional tests and identifying the bugs, we worked to resolve these issues. The bugs were thoroughly investigated to understand their root causes, and appropriate fixes were implemented. We followed an iterative approach, focusing on identifying and resolving one bug at a time to ensure the stability and reliability of the website.

In addition to functional testing, the website was also tested with real-time users. This involved inviting a group of users to interact with the website and provide feedback on their experience. These real-time users were encouraged to perform typical actions and tasks on the website, such as signing up, browsing content, making purchases, and interacting with various features.

The objective of testing with real-time users was to observe and gather insights into the usability of the website from an end-user perspective. Any issues or bottlenecks encountered by the users were documented and addressed promptly. This approach allowed the development team to make necessary improvements and optimizations to enhance the user experience.

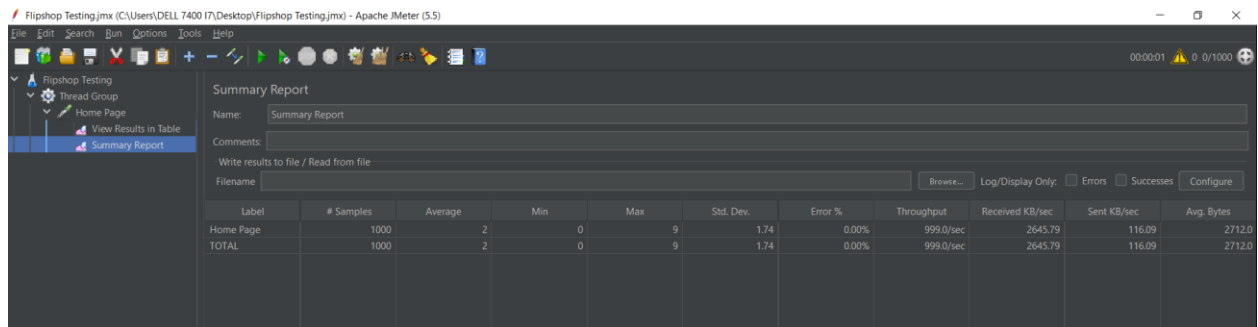
By conducting functional testing with the provided use cases and involving real-time users in the testing process, the team was able to identify and address the bugs that occurred during the testing phase. The website was thoroughly tested, and the resolved issues ensured a smoother user experience, improved functionality, and enhanced overall performance.

## 4.2 Non-Functional Testing:

In addition to functional testing, non-functional testing was conducted to evaluate the performance, scalability, and accuracy of the website. To accomplish this, we utilized JMeter, a popular performance testing tool, to perform various types of non-functional tests, including load, stress, spike, and regression testing.

### 4.1.1 Load Testing

Load testing was performed to assess the website's performance under normal and expected user loads. The testing was conducted according to the given requirements, which specified a target of 200 users per second. By simulating concurrent user interactions, the team analyzed how the website handled the expected load and ensured that it responded efficiently without any significant performance degradation.

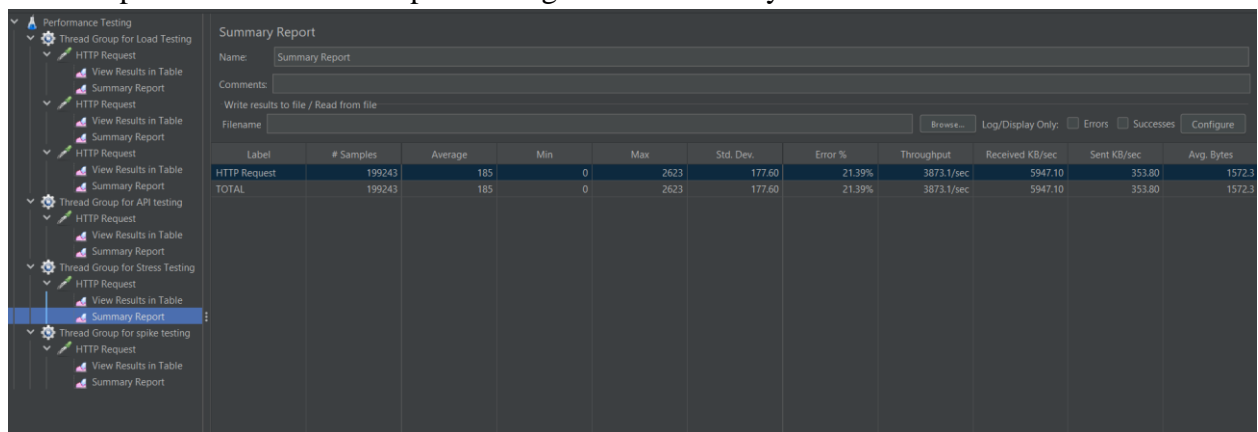


The screenshot shows the Apache JMeter 5.5 Summary Report for a test named 'Flipshop Testing.jmx'. The report displays performance metrics for the 'Home Page' and 'TOTAL' across 1000 samples. The metrics include Average, Min, Max, Std. Dev., Error %, Throughput, Received KB/sec, Sent KB/sec, and Avg. Bytes. The error rate is 0.00%.

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
Home Page	1000	2	0	9	1.74	0.00%	999.0/sec	2645.79	116.09	2712.0
TOTAL	1000	2	0	9	1.74	0.00%	999.0/sec	2645.79	116.09	2712.0

### 4.1.2 Stress Testing

Stress testing aimed to determine the website's stability and performance under high-stress conditions. The website was subjected to a higher-than-expected user load to assess its behavior and identify any potential bottlenecks or issues. The system gave 21% error under 1000 users per second stress. This type of testing helps ensure that the website can handle sudden spikes in traffic or unexpected surges in user activity.



The screenshot shows the Apache JMeter 5.5 Summary Report for a test named 'Performance Testing'. The report displays performance metrics for 'HTTP Request' and 'TOTAL' across 199243 samples. The metrics include Average, Min, Max, Std. Dev., Error %, Throughput, Received KB/sec, Sent KB/sec, and Avg. Bytes. The error rate is 21.39%.

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
HTTP Request	199243	185	0	2623	177.60	21.39%	3873.1/sec	5947.10	353.80	1572.3
TOTAL	199243	185	0	2623	177.60	21.39%	3873.1/sec	5947.10	353.80	1572.3

Performance Testing

Thread Group for Load Testing

View Results in Table

Summary Report

HTTP Request

View Results in Table

Summary Report

HTTP Request

View Results in Table

Summary Report

Thread Group for API testing

HTTP Request

View Results in Table

Summary Report

Thread Group for Stress Testing

HTTP Request

View Results in Table

Summary Report

Thread Group for spike testing

HTTP Request

View Results in Table

Summary Report

View Results in Table

Name: View Results in Table

Comments:

Write results to file / Read from file

Filename: Browse... Log/Display Only: ☐ Errors ☐ Successes ☐ Configure

Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)
230350	02:06:47.316	Thread Group for Str...	HTTP Request	1220		1232	119	1220	0
230351	02:06:48.519	Thread Group for Str...	HTTP Request	0		2825	0	0	0
230352	02:06:47.349	Thread Group for Str...	HTTP Request	1188		1232	119	1188	0
230353	02:06:47.348	Thread Group for Str...	HTTP Request	1190		1232	119	1190	0
230354	02:06:48.519	Thread Group for Str...	HTTP Request	0		2825	0	0	0
230355	02:06:48.519	Thread Group for Str...	HTTP Request	0		2825	0	0	0
230356	02:06:48.519	Thread Group for Str...	HTTP Request	0		2825	0	0	0
230357	02:06:48.519	Thread Group for Str...	HTTP Request	0		2825	0	0	0
230358	02:06:47.349	Thread Group for Str...	HTTP Request	1190		1232	119	1190	0
230359	02:06:48.519	Thread Group for Str...	HTTP Request	0		2825	0	0	0
230360	02:06:47.350	Thread Group for Str...	HTTP Request	1189		1232	119	1189	0
230361	02:06:48.518	Thread Group for Str...	HTTP Request	1		2825	0	0	1
230362	02:06:48.518	Thread Group for Str...	HTTP Request	1		2825	0	0	1
230363	02:06:47.350	Thread Group for Str...	HTTP Request	1190		1232	119	1190	0
230364	02:06:48.518	Thread Group for Str...	HTTP Request	1		2825	0	0	1
230365	02:06:48.518	Thread Group for Str...	HTTP Request	0		2825	0	0	0
230366	02:06:48.518	Thread Group for Str...	HTTP Request	0		2825	0	0	0
230367	02:06:48.518	Thread Group for Str...	HTTP Request	0		2825	0	0	0
230368	02:06:47.350	Thread Group for Str...	HTTP Request	1190		1232	119	1190	0
230369	02:06:48.518	Thread Group for Str...	HTTP Request	0		2825	0	0	0
230370	02:06:48.517	Thread Group for Str...	HTTP Request	1		2825	0	0	1
230371	02:06:48.517	Thread Group for Str...	HTTP Request	1		2825	0	0	1
230372	02:06:48.517	Thread Group for Str...	HTTP Request	1		2825	0	0	1
230373	02:06:48.517	Thread Group for Str...	HTTP Request	1		2825	0	0	1
230374	02:06:48.517	Thread Group for Str...	HTTP Request	1		2825	0	0	1

☐ Scroll automatically? ☐ Child samples?

No of Samples: 309243 Latest Sample: 32 Average: 191 Deviation: 173

### 4.1.3 Spike Testing

Spike testing was performed to evaluate the website's response and stability when there are sudden and significant increases in user activity. By simulating sudden spikes in user load, we assessed the website's ability to scale and handle the increased demand without adversely impacting its performance or causing any system failures. System gave 59.69% percent error with 1000 users in no time.

Summary Report										
Name: Summary Report										
Comments:										
Write results to file / Read from file										
Filename:							Browse...	Log/Display Only: <input type="checkbox"/> Errors <input type="checkbox"/> Successes <input type="checkbox"/> Configure		
Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
HTTP Request	31000	104	0	1359	219.72	59.69%	85.2/sec	181.71	3.99	2182.9
TOTAL	31000	104	0	1359	219.72	59.69%	85.2/sec	181.71	3.99	2182.9

### 4.1.4 Regression Testing

Regression testing was carried out to ensure that the changes made during bug fixes and feature enhancements did not introduce new issues or negatively impact existing functionalities. By retesting previously validated features and scenarios, the team verified that the website's overall functionality remained intact after bug fixes and updates were implemented.

### 4.1.5 Accuracy Testing

Furthermore, the team set a requirement for the accuracy of the machine learning (ML) model used within the website. The specified target was a minimum of 90% accuracy. However, the team achieved an impressive 95% accuracy rate, surpassing the defined requirement. This high accuracy level demonstrates the effectiveness and reliability of the ML model incorporated into the website.

```
> ~
from sklearn.ensemble import RandomForestRegressor

# Instantiate the regressor
rfreg = RandomForestRegressor()

# Train the regressor on the training data
rfreg.fit(X, Y)

# Make predictions on the test data
pred = rfreg.predict(X_test)

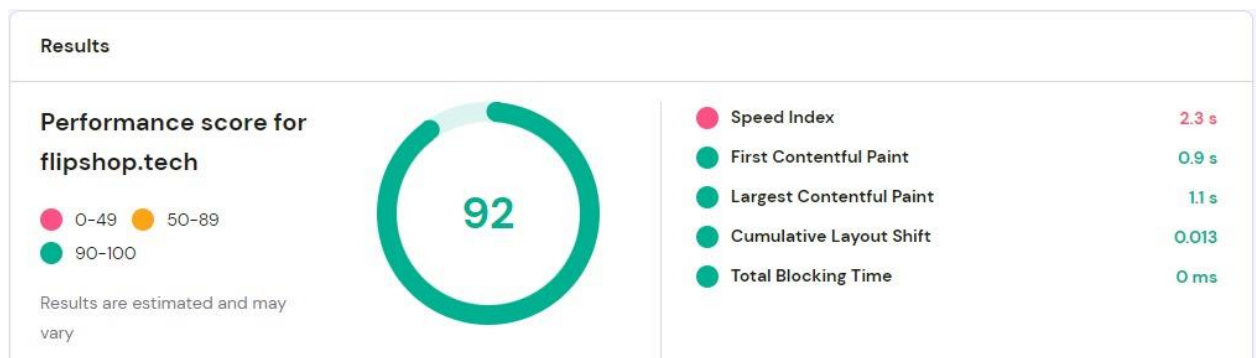
# Evaluate the performance of the regressor on the test data
score = rfreg.score(X_test, y_test)

# Print the score
print("RandomForestRegressor score: {:.2f}".format(score))

398]
Python
... RandomForestRegressor score: 0.95
```

### 4.1.6 Performance Testing

Throughout the non-functional testing phase, the performance of the system was closely monitored using Hostinger, a monitoring tool. This allowed the team to track and analyze various performance metrics, such as response times, resource utilization, and server stability. By continuously monitoring the system's performance, the team could identify any potential performance bottlenecks and make optimizations to ensure optimal website performance.



#### **4.1.7 Scalability Testing**

During the scalability testing phase, we followed a systematic approach to validate the system's ability to handle the addition of new asset types. Initially, we integrated one asset type into the system and thoroughly tested its performance and stability. The system proved to be working seamlessly with the new asset type, meeting the expected performance benchmarks.

Building upon the successful integration of the first asset type, we proceeded to add a second asset type to further evaluate the system's scalability. This step aimed to assess the system's capacity to handle increased complexity and workload. The testing process involved simulating a realistic scenario by introducing a significant number of assets of both types into the system simultaneously.

The results of the scalability testing were highly satisfactory, as the system effortlessly handled the increased load and successfully processed and managed the diverse asset types. The multi-tier architecture of the system played a crucial role in ensuring that the addition of the second asset type did not impact the stability or performance of the existing assets and functionalities.

Overall, the successful integration and testing of multiple asset types demonstrated the system's scalability and its ability to accommodate new additions with ease. This validated the effectiveness of the design approach, providing confidence that the system can scale efficiently to support the incorporation of additional asset types in the future while maintaining stability and optimal performance across all tiers.

In summary, through the use of JMeter, the non-functional testing phase evaluated the website's performance, scalability, and accuracy. The website successfully passed the load, stress, spike, and regression tests, meeting the specified requirements. Additionally, the ML model surpassed the required accuracy rate, and the performance of the system was continuously monitored using Hostinger, ensuring a high-performing and reliable website.

# DEPLOYMENT AND SYSTEM INTEGERATION

## 5.1 Hosting Plan, Domain and Subdomain

A shared hosting plan for six months was purchased. A domain “*flipshop.tech*” was also purchased for the project. A subdomain “*api.flipshop.tech*” was created to host backend separately. FileZilla Client was used to upload files on server.

## 5.2 Backend

The back end is hosted on the created subdomain “*api.flipshop.tech*”.

## 5.3 Databases

MYSQL is used for the databases. Remote database with server and hostname are created to host live databases.

## 5.1 Frontend

As mentioned above the front-end is developed using React. The react components are developed in such a way the data is rendered from the backend using the REST APIs developed and deployed on Laravel. The react project is also deployed on Hostinger. First the build command “*npm run build*” and then the built is uploaded to Hostinger.

## 5.2 FTP Server for Assets

The assets uploaded via web-application are uploaded on assets folder using the ftp storage disk available in Laravel. Thus, a separate ftp server is used to host the asset directory. The following figure displays the system integration and deployment of the project.

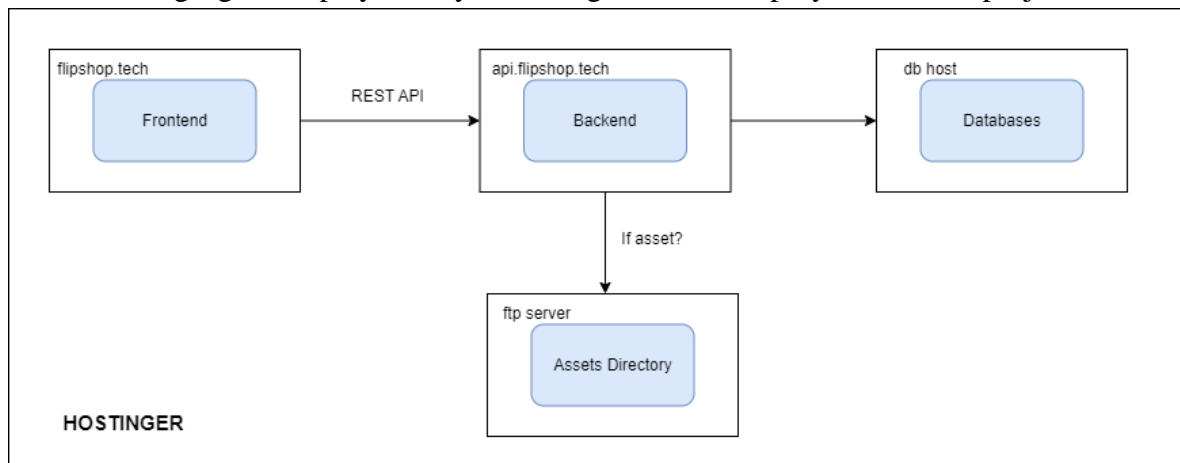


Figure 14- Deployment Structure

# USER INTERFACES

The user interface design for Flip Shop, a web application for buying and selling digital assets, is inspired by the design principles of Google. This includes a focus on simplicity, consistency, and accessibility, as well as the use of color, typography, and imagery to create a visually appealing and intuitive user experience. The use of Google's design principles helps to create a cohesive look and feel for the platform and ensure that users can easily navigate and interact with the platform. This means that the interface is uncluttered and easy to navigate, with a focus on functionality and usability. The use of clean and simple design elements, such as clear typography and ample white space, helps to ensure that users can find what they're looking for quickly and easily. The design is also optimized for all devices, so users can access the platform from their desktop or mobile device with the same level of ease.

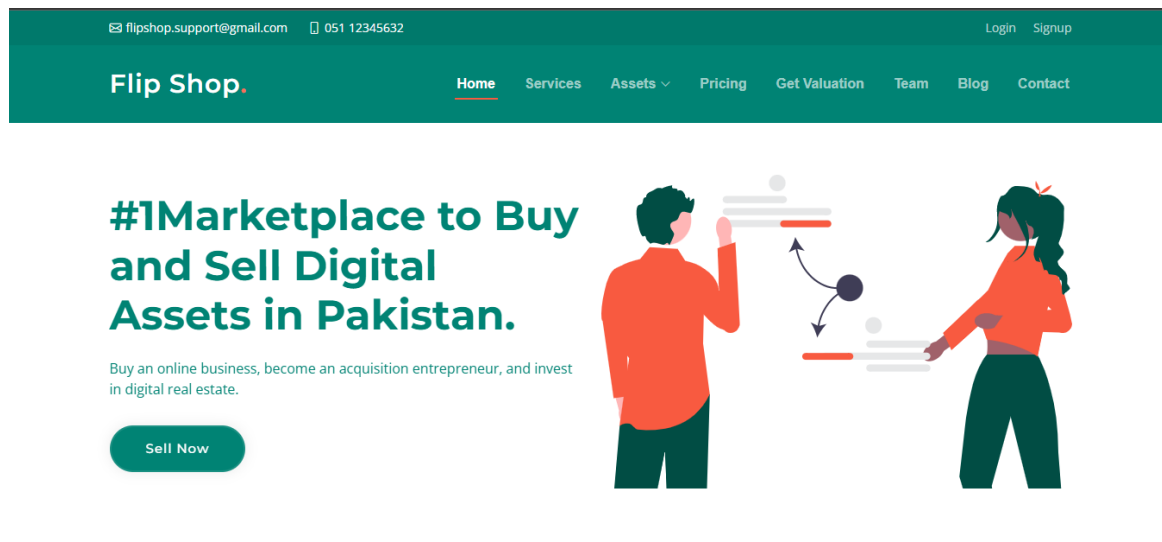


Figure 15 - UI Figure 1



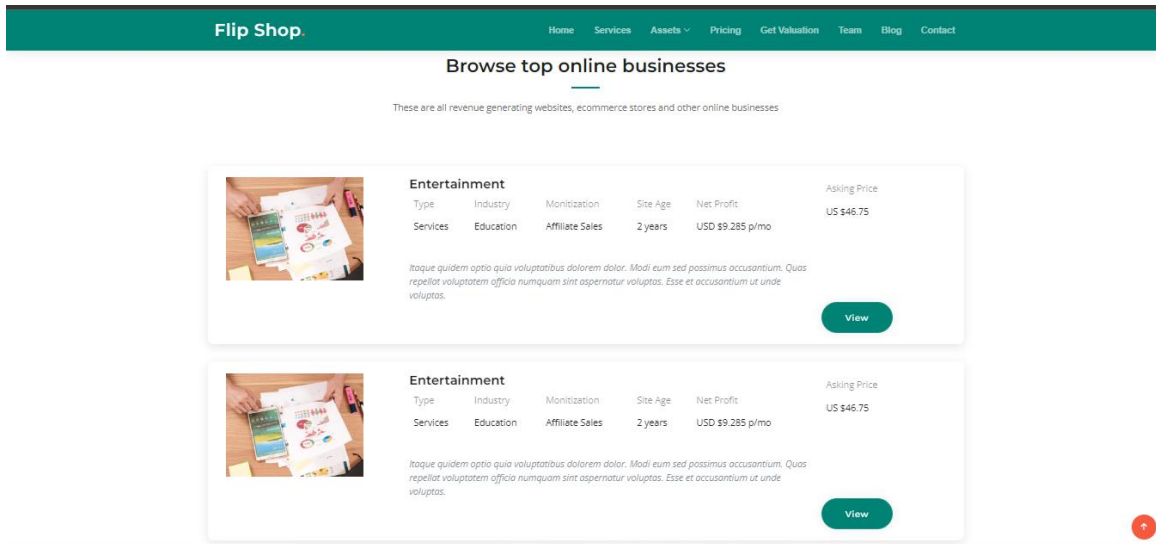


Figure 16- UI Figure – Listings

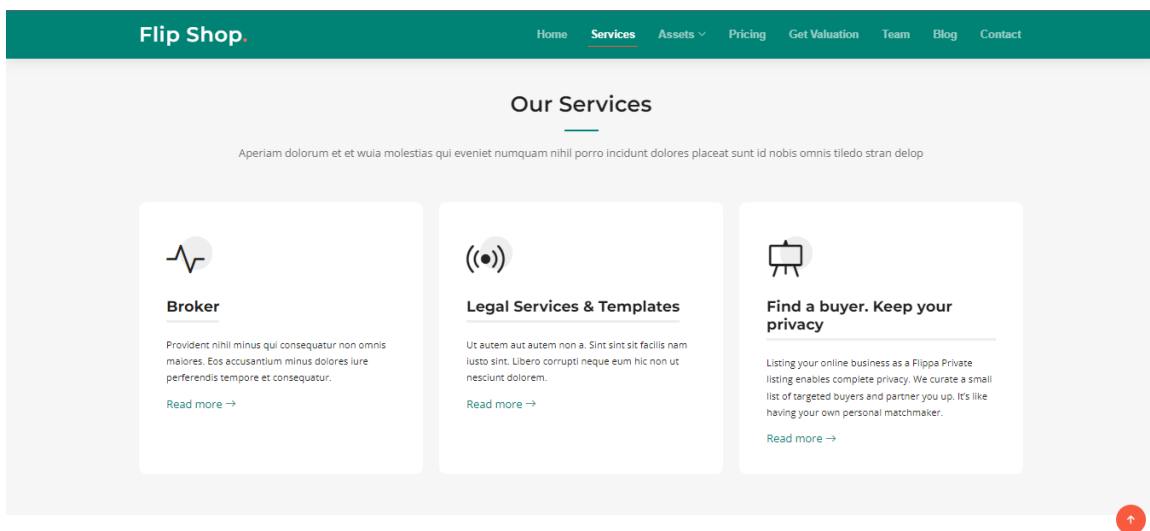


Figure 17 - UI Figure - Services Page

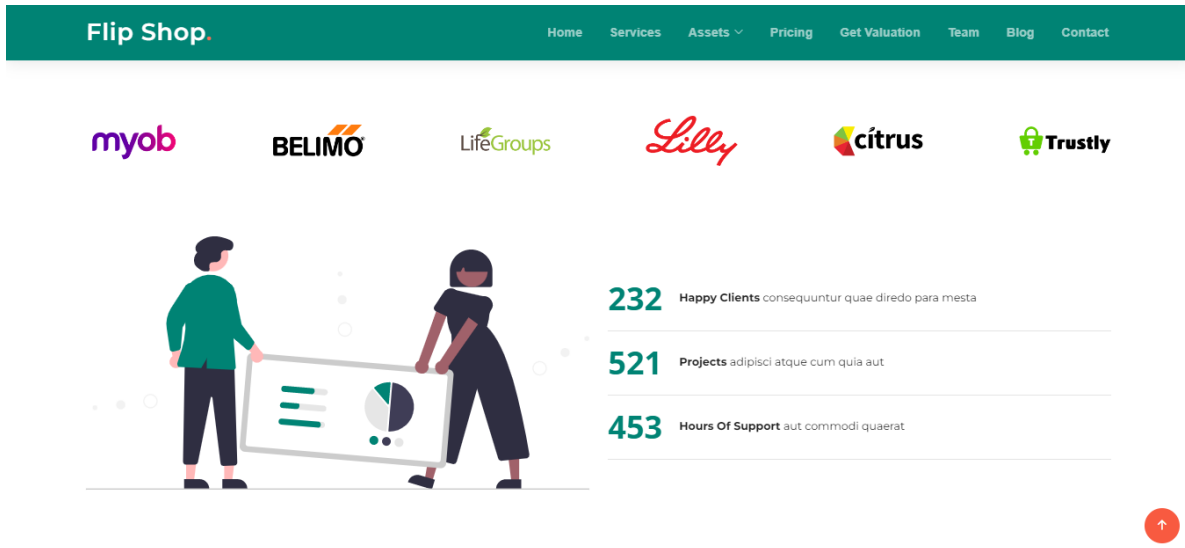


Figure 18- UI Figure - Stats

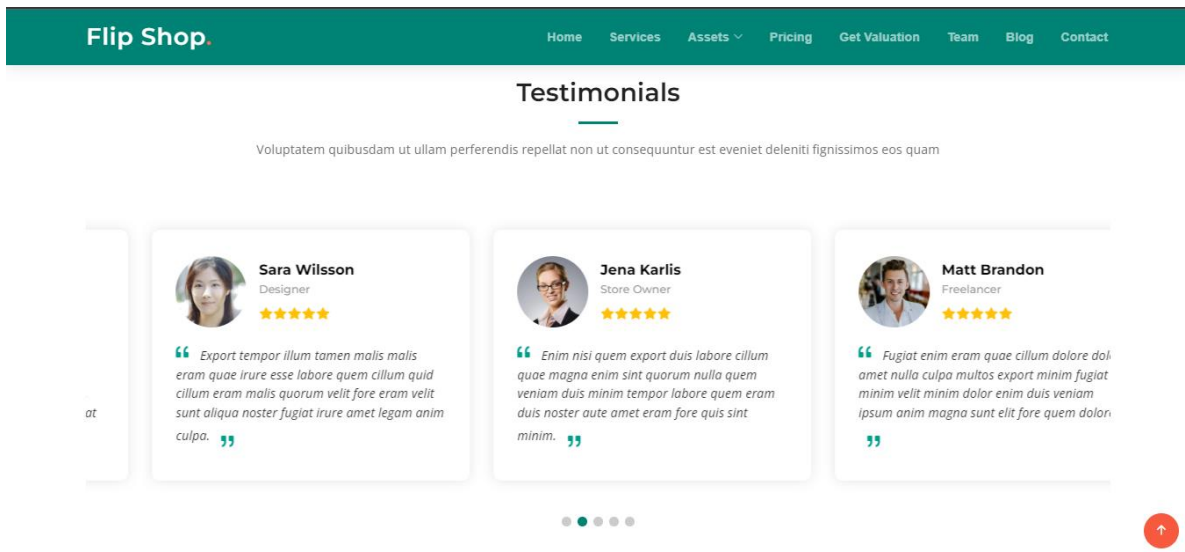


Figure 19 - UI Figure - Testimonials

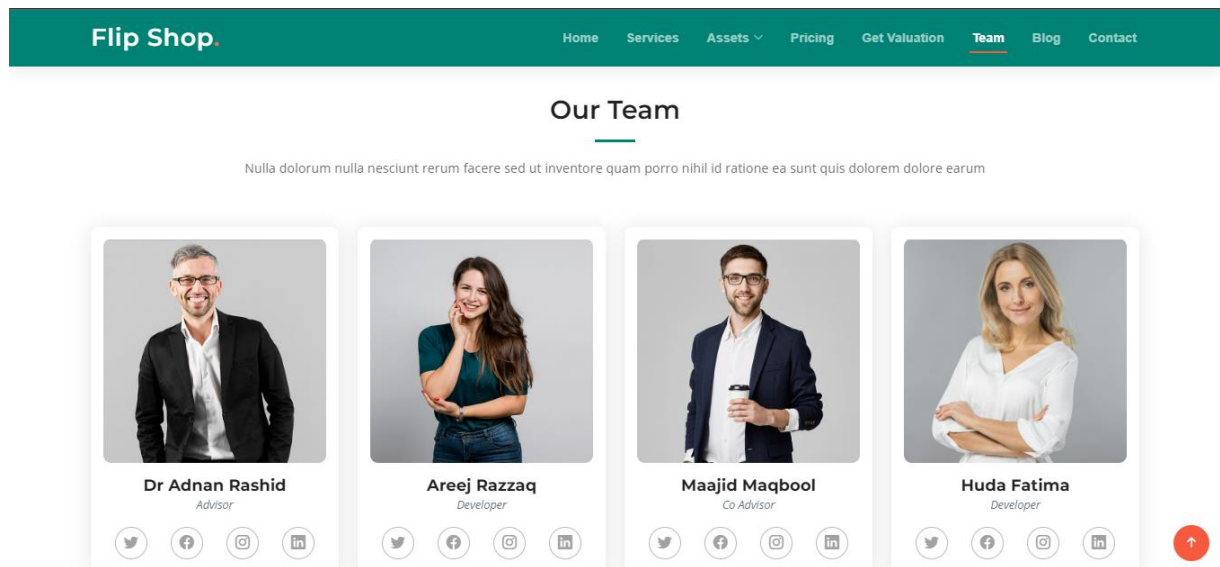


Figure 20 - UI Figure - Team

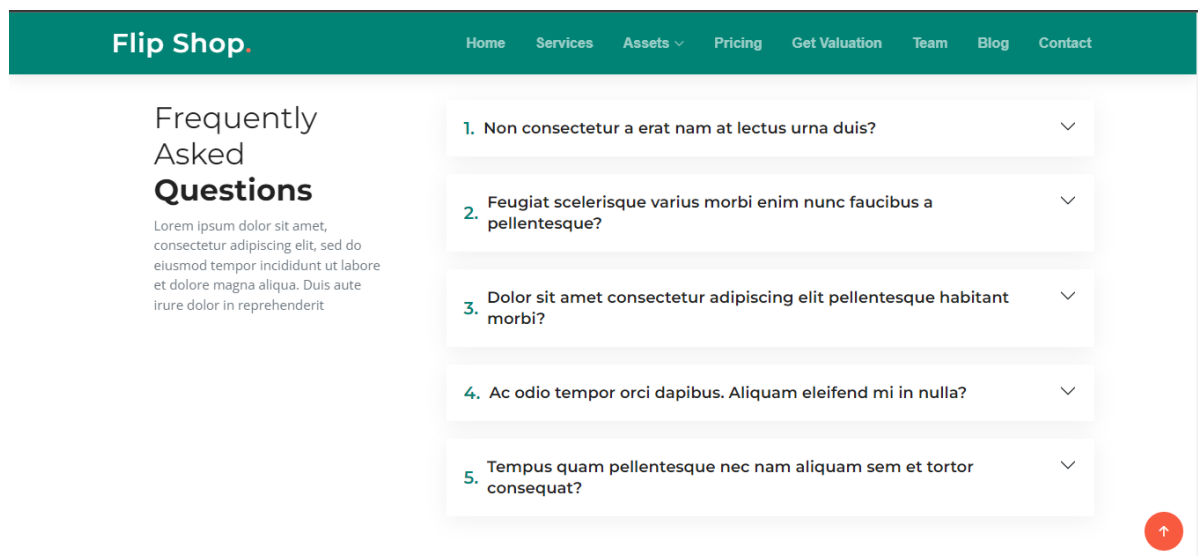


Figure 21 - UI Figure – FAQ

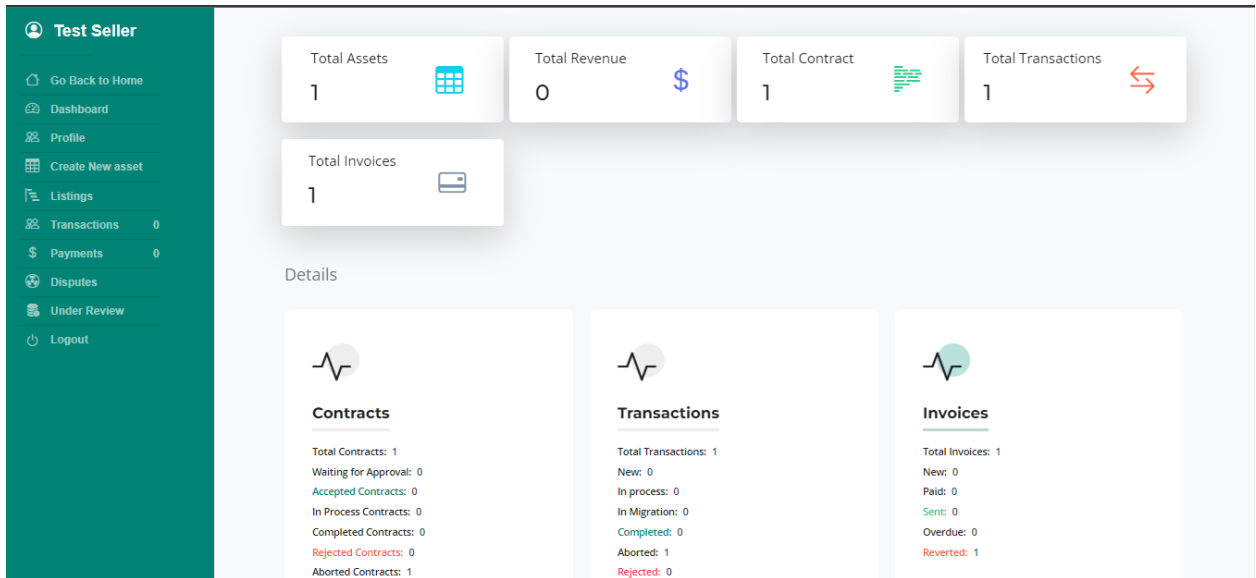


Figure 22- Seller Dashboard

**Test Seller**  
seller@flipshop.com

**VERIFIED**

**Edit Profile** 83% Complete **Update**

**USER INFORMATION**

Name: Test Seller Email address: seller@flipshop.com

Picture: Choose File No file chosen

**CONTACT INFORMATION**

Mobile Number: 67896553 Location:

Instagram: Facebook: Twitter:

**ABOUT ME**

About me: hey i am a seller

**Summary Card:**

- xx Listings 1 Reviews xx Sales
- Test Seller**
- Created At: 4/5/2023
- Seller at - Flip Shop
- Cras fermentum odio eu feugiat lide par.

Figure 23- User Profile

## **RESULTS AND DISCUSSION**

The Flip Shop project has successfully developed an automated solution that caters to the needs of its clients, providing a seamless and efficient experience. The project focuses on enhancing the user experience through its user-friendly interface and transparent procedures, ensuring a smooth journey for both buyers and sellers.

Flip Shop serves as a broker, connecting sellers with potential buyers and providing them with a reliable marketplace to advertise their digital assets. This platform acts as a trusted intermediary, offering transparency and ensuring smooth and secure transactions for both parties involved. By automating the evaluation process through a machine learning model, Flip Shop streamlines the asset assessment and pricing, simplifying the selling process for asset owners.

Transparency plays a crucial role in the Flip Shop platform. The project has implemented robust procedures to ensure transparent transactions, fostering trust and confidence among buyers and sellers. In case any disputes arise, Flip Shop is equipped to handle them effectively, providing a fair resolution process and maintaining the platform's integrity.

The success of Flip Shop lies in its ability to tap into the growing market of digital asset sales and provide a tailored solution for the local community. By offering a reliable marketplace, automating evaluations, and ensuring transparent transactions, Flip Shop has established itself as a valuable platform for buyers and sellers alike.

Looking ahead, Flip Shop can further enhance its offerings by continuously improving its machine learning model to provide more accurate evaluations and pricing suggestions. Additionally, expanding its reach and user base would contribute to the growth and sustainability of the platform, ultimately solidifying Flip Shop's position as a leading marketplace for digital asset transactions in Pakistan.

## **CONCLUSION AND FUTURE WORK**

In conclusion, while the system has successfully implemented the given requirements, there are exciting opportunities for future work and enhancements. One potential avenue for development is the extension of the digital asset concept to incorporate additional asset types such as Ecommerce Websites and Instagram Stores. This expansion would allow the system to cater to a broader range of digital assets, providing a more comprehensive and versatile solution.

Furthermore, there is room for improvement in terms of establishing standards and procedures for administrators. By implementing transparent guidelines, administrators can efficiently manage and oversee the system, ensuring smooth operations and enhanced user experiences.

Another area of future work lies in leveraging machine learning techniques, such as Recommendation Engines and Spam Detection, to augment the web application. By integrating these technologies, the system can offer personalized recommendations to users, enhancing their browsing and purchasing experiences. Additionally, employing advanced spam detection algorithms can help mitigate unwanted and malicious content, ensuring a safer and more secure environment for users.

In summary, the future work for the system includes expanding the asset types, establishing standards and procedures for administrators, and leveraging machine learning capabilities to further enhance the web application. These enhancements will contribute to a more comprehensive and advanced system, delivering improved functionality, user experience, and security.

## REFERENCES

<https://www.forbes.com/sites/johnkoetsier/2020/06/12/covid-19-accelerated-e-commerce-growth-4-to-6-years/?sh=445d8cff600f>

<https://github.com/features/issues>

<https://flipit.pk/>

<https://flippa.com/>

<https://empireflippers.com/>

<https://exchangemarketplace.com/>