MunasibMall.PK

SPROJ Report

Daniyal Mumtaz 22100168
Abdur Rehman Masood 22100134

Muzammil Khan 21100161

Wali Ullah Aitemad 21100034
Waqar Ul Haq Khatana 22100199

Advisor: Waqar Ahmad
School of Science and Engineering
Lahore University of Management Sciences
Submission Date 18/5/22

Acknowledgement and Dedication

I am grateful to Allah Almighty with Whose grace and mercifulness this project was completed. I would also like to thank my supervisor, Waqar Ahmad, for his guidance throughout the project, and my group members for their valuable efforts and helping behavior. I would like to dedicate this project to my parents and siblings for their constant support and prayers throughout my academic career.

Waqar Ul Haq Khatana

Certificate

I certify that the senior project titled "**Add project title here**" 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

List of Figures

1. Introduction

a. Introduction

Munasib Mall is a mobile e-commerce application. It will allow different merchants to have multiple dynamic web stores and customers will be able to buy from them on this platform. Merchants will have complete control over their store(listing products, removing products) and will be able to customize their store according to their wants. They will also be given some customizability in designing their store page. Customers will be able to order products of all the merchants/stores. Customers will be able to look for specific products across various stores. Orders will be processed by the merchant who will also handle the delivery.

Payment options like easy paisa or COD will be shown to the customer who can choose their preferred method when purchasing. The system aims to provide a platform that gives more autonomy and the ability to handle orders to the merchants. Reviews and ratings will help the customer choose the right merchant.

b. Objective and Scope

The objectives of this project are, to provide a user-friendly mobile application to customers who can buy products that are used daily from the comfort of their homes, to provide the opportunity of selling products (by signing on the app as a merchant) to a wide range of users, and to make reviews of merchants and products more accessible to the customers to increase customer satisfaction.

c. Development Methodology

The group has chosen agile methodology to complete the project because the development of a mobile application like the group has chosen needs an iterative approach. The work was divided

into sprints so that a working model with a certain number of use-cases is prepared at the end of each sprint. The goals for each sprint were achieved by using this approach and the project was completed according to the requirements set at the beginning.

d. Contributions

Our mobile app offers a convenient option for buying fast-moving consumer goods. It is better than similar applications e.g. daraz. because the user has an easy to use interface where the categories of products can be seen more clearly, and more products can be searched if the desired category is not listed on the main screen. We offer an option of wishlist so that if the customer likes an item and does not want to purchase it at that time, the product can be added in the wish list to be bought at a later time.

2. System Requirements

The system requirements lists down the actors that are involved in the application, which are the merchant and the customer. It also includes the details of the functional requirements, which are the services that the software is offering. Finally, it includes the non-functional requirements which are the constraints or requirements imposed on the system, and deal with issues like scalability, maintainability, performance, etc.

a. System Actors

List down the actor names and give a 2-3 lines description of the role of each actor

Actor Name	Description
Merchant	Merchant has the ability to create and manage multiple stores and make product listings on them.
Customer	A registered or unregistered user that can make purchases from different merchants.

b. Functional Requirements

1) The user will be able to register to become a merchant

2) The user will be able to sign in using their credentials to be able to see their dashboard.

3) Merchants will be able to edit their profile information

4) User will be able to log out of the system

5) The user (Merchant) will be able to create a store

6) The user (Merchant) will be able to delete their store

7) The user (Merchant) will be able to edit their store information

8) The user (Merchant) will be able to search their stores by name

9) The user (Merchant) will be able to list multiple products on their store as well as upload photos of the products.

10) The user (Merchant) will be able to search their products based on different filters.

11) The user (Merchant) will be able to delete their products.

12) The user (Merchant) will be able to view a particular customer's order history.

13) The user (Merchant) will be able to search customers based on different filters.

14) The user (Merchant) will be able to update order status such as in-transit, delivered, etc.

15) The user (Merchant) will be able to search orders based on different filters.

16) The user can sign in as a customer to avail discounts/promotions on items.

17) The customer will have the option to logout and if that customer is inactive for a given amount of time then that it should be logged out by the system automatically.

18) The customer can browse different stores and their respective inventories.

19) The customer can add an item to its shopping cart.

20) The customer can alter the quantity of the product before checkout.

21) The customer can view its shopping cart at any time.

22) This takes the items in the customers shopping cart and processes them for a purchase i.e. checkout.

c. Non-functional Requirements

Sr#	Requirements
1	All transactions by any actor should not take more than 5s to be processed on a 2Mbps internet connection.
2	Maximum error rate in which an error might occur during a transaction will be low and on average not more than 1 error every 500 requests.

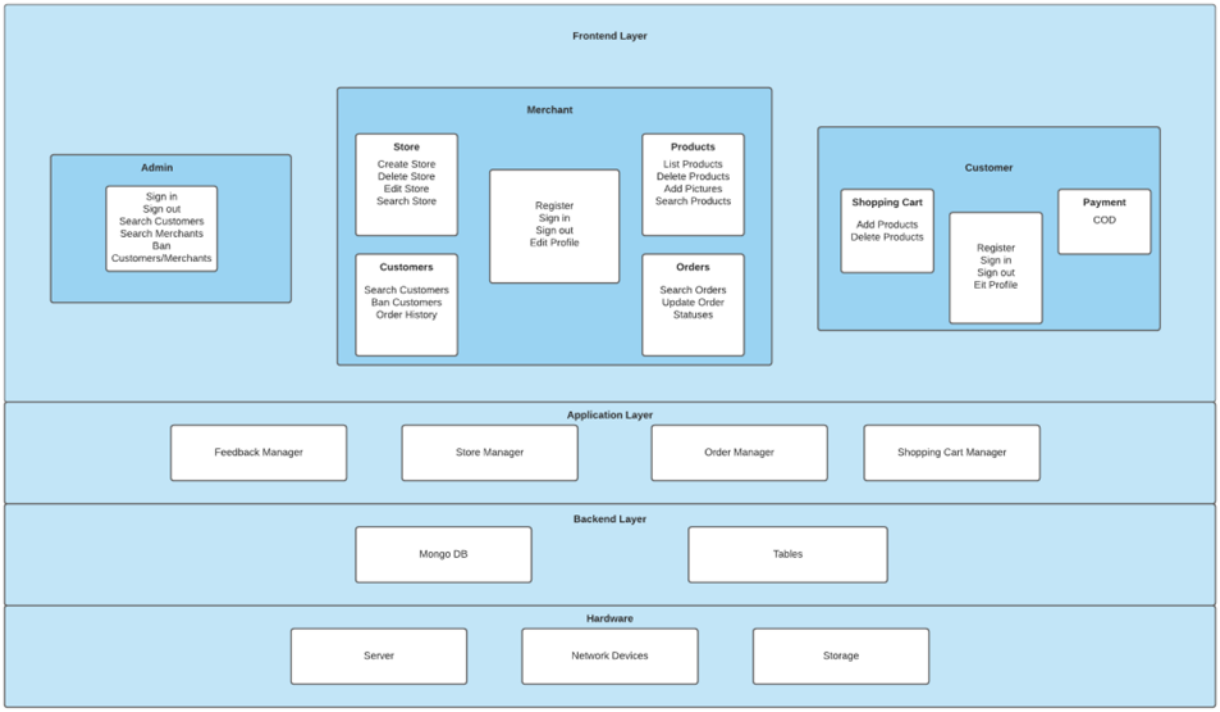
3	Ability to support 50,000 Customers and 200 merchants with the option to scale with time.
4	Throughput (transactions at a time) will be 200 users concurrently and the software will process 1500 requests on average in one day.
5	Scrolling on page will take at most 3s.
6	To secure user accounts, a Hash-based password login setup would be implemented.
7	Users can change password in case they forget it through an email sent to their registered email address.
8	Personal information of the users would be protected using encryption protocols. Passwords of users will be hashed before storing in the database.
9	Users will have the ability to stay logged in to the app.

10	In terms of usability, our interface will be user-friendly and easy to learn. There will be a step-by-step registration process. The icons used will be the ones most commonly used in some of the popular market places such as Daraz, Shopify, Alibaba. Products display will be similar to Daraz.
11	The app should be available on google play store.
12	The app should run on android OS.
13	The uptime of the system will be at least 90%.
14	The app should support file/image uploads of up to 100 MB in size for a single seller for a particular product.
15	All functionalities will be supported by touch input methods.
16	The system should be hosted on a web service that can support a twofold increase in user interaction each year for the next five years without degradation in performance. The initial system will be able to support 50,000 customers and 200 merchants.

17	The hosting service must also support a twofold increase in the database volume each year for the next five years. Initial system would be able to store 5tb of data.

3. System Architecture

a. Architecture Diagram



b. Architecture Description

FrontendLayer: The frontend layer is the layer that the user will directly react with. This will be built using React Native. This layer will provide an interface between the client and the server. The clients will send HTTP requests to the server via the frontend layer and the server will send the HTTP response to the client via the same layer.

Application Layer:

This layer contains the basic functionality of our software, which includes store manager and order manager for merchants and shopping cart and feedback manager for customers.

Backend Layer:

The backend layer consists of MongoDB and Tables. This layer will store and process data. All the CRUD operations will be performed on this layer.

Hardware Layer:

The hardware layer consists of a server which is going to be developed using Node JS and Express JS, storage devices, and the network devices.

c. Justification of the Architecture

Pros:

- Views and controllers can be easily be added, removed, or changed.
- Views can be added or changed during execution.

- User interface components can be changed, even at runtime.
- Scalable, easier to add more servers.
- React provides optimal performance for view.
- Easier to make reusable code.
- Front-end libraries and plugins support.
- Modular architecture.
- Easier to manage development.
- Easier to update and maintain the code.

Cons:

- Views and controllers are often hard to separate.
- Frequent updates may slow data display and degrade user interface performance.
- MVC style makes user interface components (views, controllers) highly dependent on model components.
- Clean separation between layers is often hard to achieve.
- Harder for developers to ensure consistency.

Justification:

With the MVC architecture model, we feel it would be easier to work parallely on the application. We would need to ensure all the layers are consistent in use and coordinate for that. Other than that, it is easier for us to autonomously work on separate layers. The MVC model would also make it easier for us scale up in the future and offer different updates easily when needed. With thousands of potential users, load balancing is important which we will be able to implement using our architecture.We will be following the scrum model, potentially adding new features in some modules previously developed.

How it helps non-functional requirements:

The modifiability of the system will be modeled with Model View Controller (MVC). This will allow maintainability and more structural means for a developer to easily make changes to the system when it is required.In the case of failure a new server can be deployed within 10 minutes hence making the system up and running within the time prescribed in the non functional requirements. The low coupling along with high cohesion will allow the developers to add new features and write new code, without compromising the existing ones. This makes the system highly extensible. Since the servers can be replicated across many servers, the system will be able to manage 5000 requests. Moreover, The performance of the whole system aims to ensure ease of comfort to have the site load within 5 seconds.

d. Tools and Technologies

List down development stack, tools and technologies etc. that you have used for development and deployment. Make sure that you mention name and version of the tools.

1. MongoDB 5.0.

MongoDB is a cross platform, no-SQL database program. MongoDB is easily scalable and supports all functions of modern database systems.

2. React Native 0.64

   React Native is a React.js powered , JavaScript based mobile app framework which allows cross platform development.

3. Express(.js) 4.17.1

   Express.js is a back-end web application framework for Node.js.

4. Node.js 16.7.0

   Node.js is a back-end JavaScript runtime environment that executes JavaScript code outside a web browser.

5. Heroku

   Application Backend will be deployed in Heroku which is a cloud Platform as a service(PAAS).

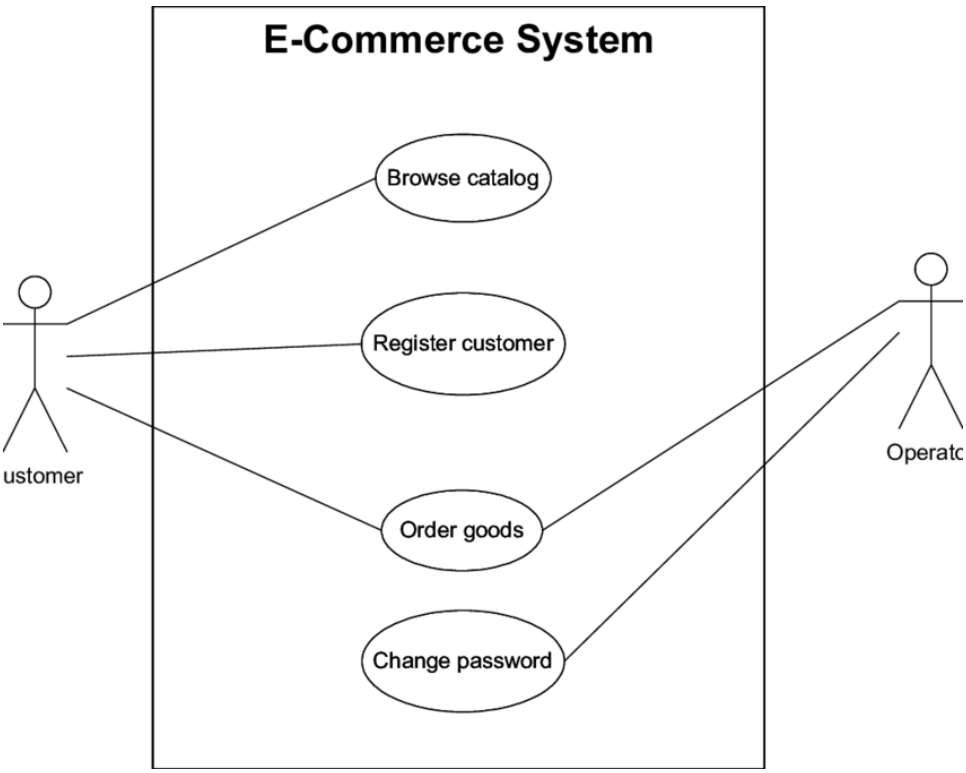
6. Android Studio

To create APK files that can be run on android to test/check the app.

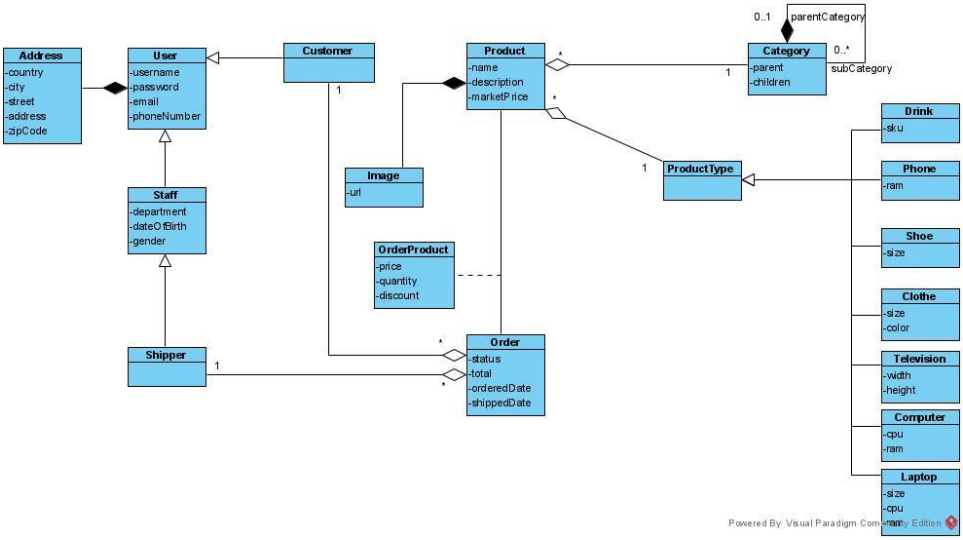
4. Requirements Specifications

Brief introduction of this chapter in a paragraph highlighting the content

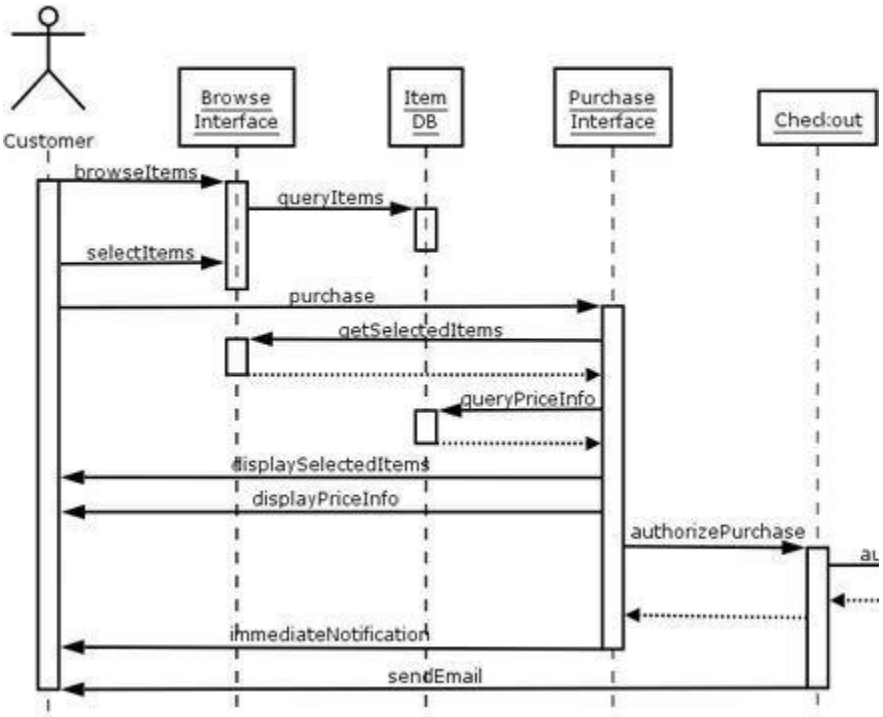
a. Use Cases



b. Class Diagram



c. Sequence Diagrams



5. Software Development Methodology and Plan

There are two main software development methodologies, agile and waterfall. The team choose an agile approach because of the nature of the project. The reasons for choosing this are listed in the following sections.

a. Software Process Selection

Agile and waterfall are two distinctive methodologies of processes to complete projects or work items. Agile is an iterative methodology that incorporates a cyclic and collaborative process. Waterfall is a sequential methodology that can also be collaborative, but tasks are generally handled in a more linear process.

Following the agile methodology, your project will move through a series of cycles throughout the lifetime of the project. The development phase, review, feedback, and then approval of the work item – either yes or no. If yes, implement and complete the task. If no, record and make any necessary changes, track and adjust the backlog or prioritization to reflect the newly acquired knowledge, and then move on to the next task or sprint.

Following the waterfall methodology is a simpler process of moving tasks through the phases of defining requirements, designing the implementation, implementing the work item, verification of implementation and quality assurance, and then maintenance of the feature in the end.

Selecting the right methodology for our project depended on preference and the nature of the project. The development of a mobile application required a more iterative process rather than a sequential approach, so agile was chosen to successfully complete the software requirements within time.

b. Gantt Chart

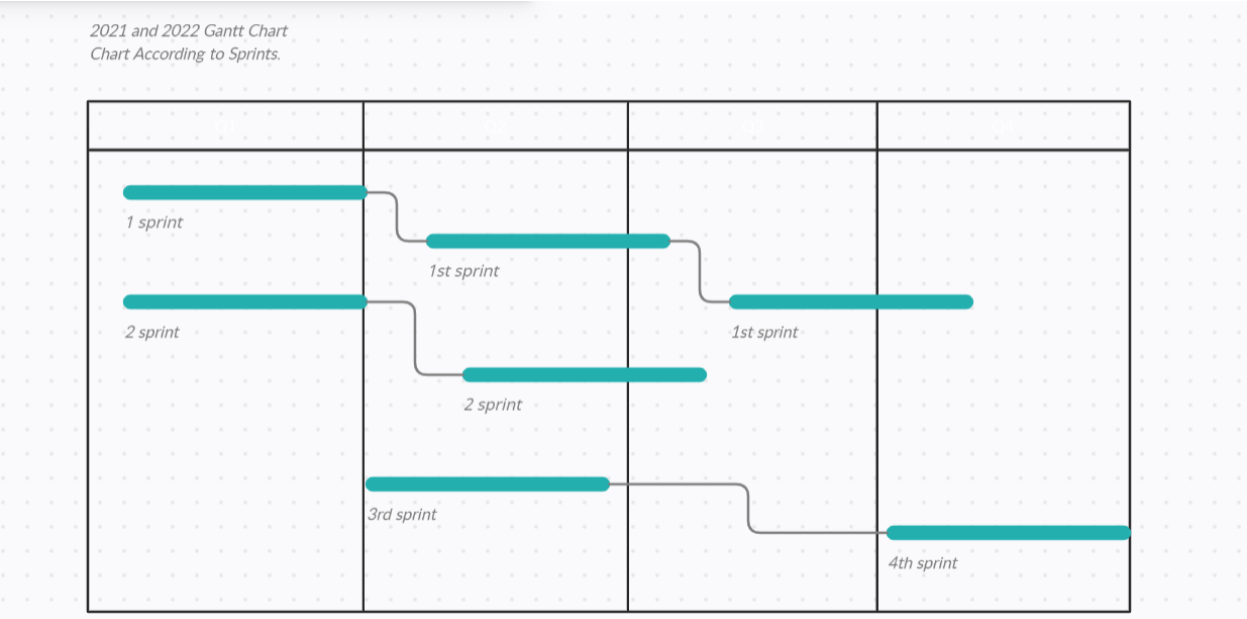
Each sprint = 8 weeks.

Sprint 1: Abdur Rehman Masood and Daniyal Mumtaz

Sprint 2: Waqar Ul Haq Khatana and Wali Ullah Aitemad

Sprint 3: Muzammil and Wali Ullah Aitemad

Sprint 4: Waqar Ul Haq Khatana, Abdur Rehman Masood, and Daniyal Mumtaz



6. Database Design and Web Services

Brief introduction of this chapter in a paragraph

a. Database Design

Create data model of your system. For example, E/R diagram. Give brief description of entities and data fields.

URL for the Data Model:
https://viewer.diagrams.net/?tags=%7B%7D&highlight=0000ff&edit=_blank&layers=1&nav=1&title=er%20diagram.drawio#R7V1bd5u4Fv41fnQWd8OjYzedrtXMpE3mnM55maUY2aYF5ANyE%2FfXjwBhg7awsY18yaSrqzWyLGDvb983omeOotePCVrM74mPw56h%2Ba89c9wzDGMwsNh%2F2ciKj2iuUYzMksAvxvTNwGPwC%2FNBjY8uAx%2BntYmUkJAGi%2FrghMQxntDaGEoS8lKfNiVh%2FawLNMNg4HGCQjj638Cn82LUtbXN%2BG84mM3LM%2Bsa%2FyZC5WQ%2BkM6RT15qQ%2FiV3pGY8kt8wEmEYhxT9s09Sn7gpGd%2FmFOa3emwZ9yxv9Ns9s2MkFmI0SJIbyYkYsOTlE25m6IoCDM6Vxa65Qux05kfeuYoIYQWn6LXEQ4zZpVsKK7pruHbNR2SbN0WP3j6%2BXGVfvhi%2F%2Bf%2B69fb6Td9MX6O%2Bzpf5icKl5zAf6bF5TEC0VVJ9fQliEJ2Cz3zFp63XAMnFL9Whvh1fMQkwjRZsSn8W9Oyi59wEPbNkgEvG5bqunkz4PPmVY5afC7ibJqt19%2FcPfvACSAnxpcvxvxWn%2FU%2FD79b3v9%2FJ86vUb9vDgAxYhThnNmMGEkQz3qGE7LruH1mJHJm2SccoSDcPmWB0vSFJP6OWXMS49%2BX0TMjvzksZwl8YATO5s5pxM451tlHFAazmH2eMHZkv7zN2BAweRnyL6LA97Of3yY4DX6h53wpLTshCTjU7duePc7WWlKSFhKfLc2ugfzAIxIStu44Jjn7p0EYCkMQEdux1honfefGqCPFkuCkxE4VJE4HGJHfhAEwMkJJpiE%2BURwplptBqYRXpU7WJfSwJPToRGi%2B%2F896ffb%2Fnt79FnnTvzXnc6A9yQhCAxrukJpFEkz4lGlIEIUzmJGZJMGCBiTevtRzguJMtG7z%2F8HXE0TxjLAbNYej8iOYFES51SmlrpAHMKtJGl%2FmAcWPC5Td0fiF2d26hDZwuY3EGFvhIKDBtCU6VIYFTZlwmAALj3OyWOR80ya5nKg1LGYbAZESpQsBkRPFAkR5QKso9wbUEsMe3NS150BCDde4kRhZT5m6gNSImJNaiPhjg4gz25kbRTblj%2FyjxMQG%2FpAWU8ZMzOGMlCK6zBy3x6uzrAepDmu7ZTUEaJgSZJzUrtoAF5zVij1RUUZMV6YxTiwkkBi7DeYiIf5yQovYRHsojkp

T9sbBbm8Hu8hjQ2IpZV6kqQztDmDwiESLEAXbrQJTYgnlMblhdiMCfVMDMiBzs5vCMUO
ZFEAiLdPSDhRRqigDEU4mc5SRYnhffoQe4ZrS1xhmHSQfznYEgDCrrYDYygQEhuJPCZr8k
DGrW4vQtwaCW%2B3IhEEmCboySYDU2O0SrR2eLY5VkI5xGDDi4DyCIiTEKJbFYnxS3c
V640Iz2I4TQWW2lBh1DpQLMLJRgUolxrPFoAuQYnDaMNSD1nWZUrb6VneyZly1rmKwnb
RZ0%2BE0xHEgTgBRsqw2J4Ru1EUax%2F4wy%2BKzww9fIxT7f2xoJ%2FsiIcvYz7RLLtaM
kMnqW3bAPA5%2B%2BNd6ZXYwfq3OHK%2F4UVuR5YxnFzPDbVKR2K8VGyAbW%2F
MpwSGiTEnWcCJjHj%2FHA8ldvRInAkz6olOVkmUywfxHRiXJL6xjeYIr1xcVTkEbsBLjHVp
VpnEFvOuCN0gsFtjgck3EI%2FLxUI6R7zNTkW7PCP65w0vMbWdaNZ5XFyW1FoU9VNXa
vW92Ar1T%2BoClDFTYP%2FSjIFZszlg8BJW0I%2FP3uggP5cCHOppjWlqcu2acrpncnj91lNrt
vI0uolT5DcCc96bOoNbvMuuU0DVp6koVcuWRCqwn7y6hti71XBPyDwk5CixBhjdDXVrLUJ
a2MiSlDJ5nVA11UMqQKGlpwKV3YZ%2BkWIfpqSssfMY%2FGVeKWZ%2FWnw%2Bsjyb
4Z4Cz%2Fp7hvgKsrIJqNJRB1t1PLUGkTqQginojozfU4L%2BPjDcw69KxoGlCvOpJKCRrMH
CVWZQDDEq1FGLfrkshEsg2hgKNSWM%2BeZOuv7rY4SA5asggb5EjGUqUZb8MWDGrqL
MT2yZbUkz1ZG5YF0keqdBAO10LnJvSwSklkx%2FFlE8MpLPrijUOwvWelUNZmleWF1JmL
1wHsCRLYj3yQ06QetaNJHROZiRG4WdCFpy23zGlK57ky8gupPleA%2Fqt8jlL0vGcHTvaZO
mygzJJV8sB8gvJx%2B6C7B55Ws8XZrCRyveN8lgkwLYQxjR67VJ6R2bqPK9uIfsDgdUNCb
YDkmJb77Mi20EKMJHMSfScVX%2FauVbH5bgdgSK6rM1Ikc8gpxGMyefovETyhJZnT2IyT
0sjaDEnJKbMpbksQsnKr6clFPTQizDnnFQauHWZk4XDp6USLFFHKGZR43np1LeEyoqsAnd
aQsHc7pRAP%2BucCtwdnJtGsPBzEbrJBtVcDfqDJ6VUGWRdGqVESNnn1uIWzCOw%2BH1
yZvUkkqltG646MkHvMsETzNzlM%2BtxU3z0RZKYOi2loI%2FJE7FKsw26LsQfUkfytN1pU
AfxFF27gs%2B%2FowfTaqjtlADXW3Tkn7baM4AuHWBSGPAwPyNf%2BTyunmcgKKIVDv
gona8TEiF6xuEDo3%2BBgHFS3M2awSGe0m3sTZmqYZB5ypMZ2mbgc%2F67sSPl%2BIIkd
ERidqlF7%2B8Yo5S%2B4JR2pKEsQUNJi7HSFiJF%2FHOhXL7zr5l%2FhtD32%2Fegb%2Be
YEgYqa4BxYYgOGAgTahWF2dR816u03lU68Vo335XP%2Fe9qsyudrn3a7OwtFD4yddc3xDZ
3MXfXts1ObOsEC3XUZaeb8vN01XYn1%2FvQBwSwa9chWk0C724OleSKq0c8I73JQv9V%2
BUaekW5E7s5s8npDiEtpEO0bjldXT6LdaIvdbCWxSfRo%2BHad5XY761LuEoONeLoYmHji
M4ltUXFoWWNfjSaeRz%2BBQnNhOukYO9qmztUJUpSaQsHb0Q8FDlxJDEjOrky8Fo7wMc
qEDT6RexSvzmnSLkYFMUDoNUCY4tYK%2B0BLfHL34iyVp9hbegfXVhf%2BCHAZVufga
j5V%2FaKNE1g9DzYBqbZ6lce7HN2sR5maZe2IM%2FOjB5wE7M6z9N5VGFJLE1UUSLS2
Dio95pezf9Z%2FnDo6TfPG1itfu0rAml%2BF1DVQi9YWuY53NdqVGoVMNsVCSesnDoWO
YVOE%2F%2FkttKTF%2BjBoNT7xKm27UxITnBgWttixu5cyqy1lXZ7r1iIuVAGMuoYRYfJ9

GS3Kp9CvAjWiBnDEhPjhmHFE764jIyde8qDjNIT8qc5Lt3CO5lRt3I1W%2BHCNdu7Qp%2FmzRVr7ebtTtVy9737o370wwQG5sCOSd6LPJj7v2lX6zhXOU98HYPd85wSC9m7v2wLQFUNg81B7D5c6a6pGiosW%2FQzvNYU2rLXFFO%2FhKLGOtvBdo0Rx5ek9EN0OiIH4UFx7bAm%2B4znrEFJkSRrlALT2yMh1wl2FqbGMt1a9FLDZouIQzQEWE6HS3aNXcv7BHuJ31aBQNdTl2T00rIQ6xjvaO%2B4cWt0m6y9dNcAuZttbi%2Fe%2B%2FNU1Y%2FdiqlXDhcQc1%2BZbiokYr7vc0Wbv%2F8uRcoWNKDluVlfYiiLZ28dc83JfGLjwnQdwMcW6QLJv3VvW5a4uSt7BLrwrKHLRWVTNuBb%2B3Vti3A5qt64A2uflmvGuVAFP4JuiLOtQneqAl%2BrAtVTz%2BEzu1Ruv1LlHtaoIQn%2BaSp3b8RMQcri1yAG%2BIUNg22Dv32P0hWXJljutwjBb%2BGDvqZrOujKtMjV2bBkpe4pSrCOJVdGzh3GS7YU68zjeOyubH6h1biz3CFh5jrb5U2%2FINTTvxqv%2BGSgxZkBOdGd9T0oNmmSbkJ7hoCjbqCF%2BThc5HLTOhtJ8p1K%2B1WC%2BaSncCLajU4lvhBLE8II3EThulyxdKKO2fPenuo2%2FoU6khCK2xrBhn%2BPsdaD5O3OL%2FVS3vET3gpm4YyeIHbu%2FGKKX5JydjdDz9YN0wuwVzfkE5LGBuUH6gAL%2Bwuv1a6X%2BLWwV8iGWwk1D2eHmDfKFtUjQYn5PfJzN%2BAc%3D

ER diagram description: We have 11 entities:

1) User: Before signup, any user of the app can view the home page as a general “user”.
2) Customer: Sign up as a customer and one can buy products by following the chronological order mentioned in section 7.

3)Merchant: Sign up as a merchant and one can create multiple stores and add multiple products.

4) Order: Placed by customer.

5) Store: Created by merchant.

6) Inventory: Merchants can add/create products and assign them categories out of the six (cosmetics, shoes, etc) and change inventory.

7) Payment: Cash on delivery payment.

8) Product: Merchant can add/create products and assign them categories out of the six (cosmetics, shoes, etc)

9) Shopping Cart: Customers add products to the shopping cart before proceeding to checkout.

10) Cart Item: Products added in the cart.

11) Category: Type of product i.e. clothing, shoes, electronics, cosmetics, etc.

b. API Specification

This sub-section will contain the list of external APIs' that you have used in your project.

1) sign up
2) login
3) account verification
4) password reset
5) create store
6) update or delete store
7) add product
8) delete product
9) edit product
10) create order

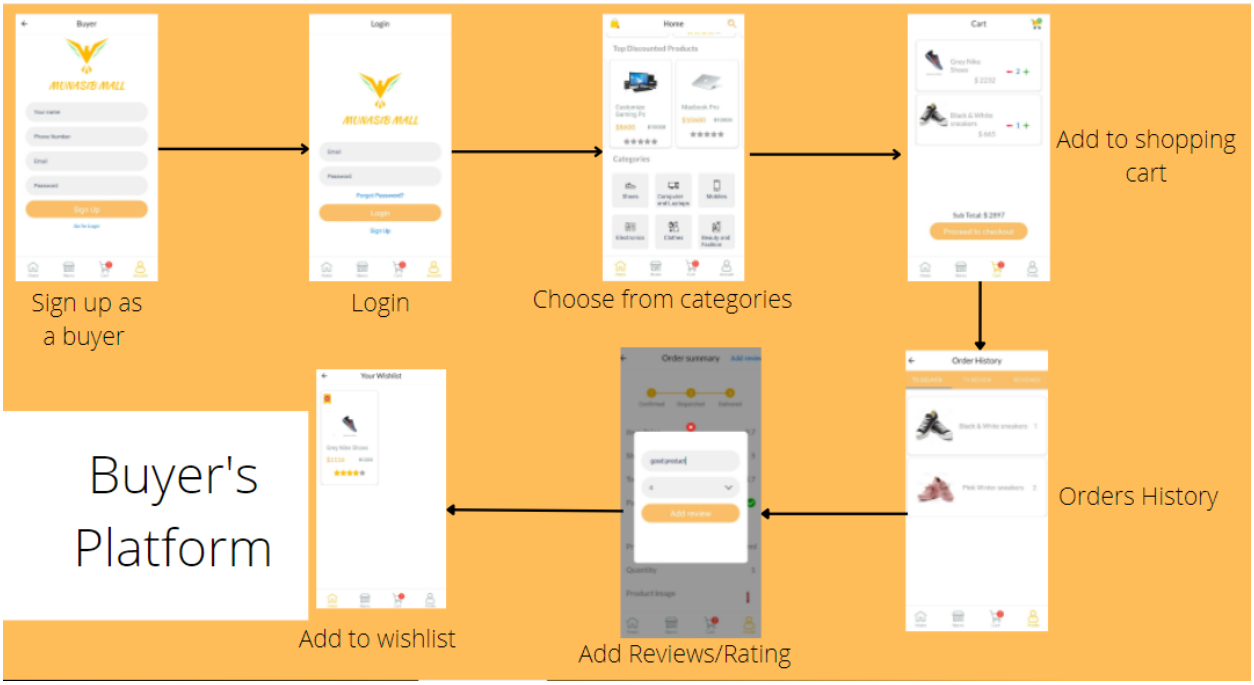
7. System User Interface

Brief introduction of this chapter in a paragraph

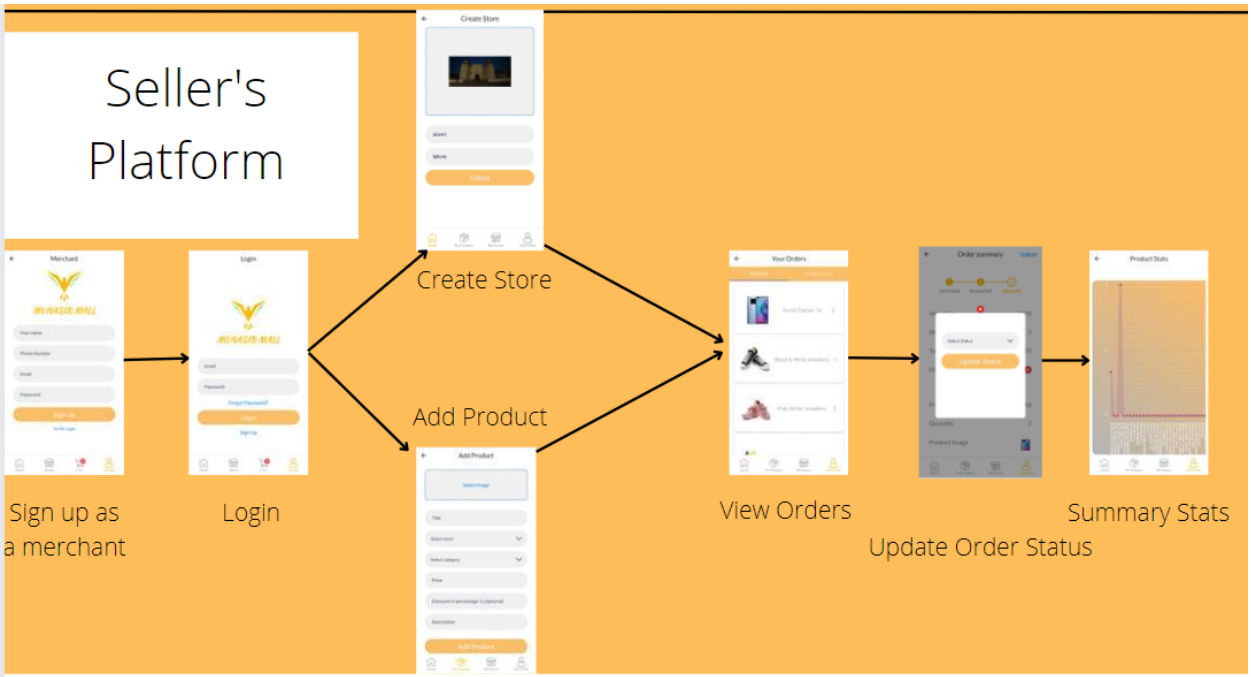
This sub-section should explain the functionality of your application to the end user with supporting screenshots.

There are 2 platforms:

1) Buyer's Platform: This platform has the following chronological order:



After sign up and then sign in, the user can choose from six different categories and apply filters as well. We have 2 filters: one is the price range filter and the other is rating(stars) filter (both lowest to highest and highest to lowest). After this, products are added to shopping cart. '+' and '-' buttons are there to increase or decrease the quantity of the items. 'Empty Cart' button is at the top right corner to wipe out all the products right away. The buyer goes to his/her order history and views products in three categories: 'to deliver', 'to review' and 'reviewed'. For items that are to delivered and to be reviewed customer can add a comment and star rating. The customer can bookmark any item by pressing the wishlist button at top left corner of the last screen above.



After sign up and sign in as a merchant, he/she can create a store. A single merchant can create multiple stores as well. He/she can add products after providing their details and assigning a category out of the six categories we have. The merchant can view orders that are categorized into further 2 types: pending and completed. For pending orders, the merchant can update status to dispatch and then delivered once he gets the payment confirmation from the delivery man since we have cash on delivery only as the payment method. Moreover, the merchant can view the trend of sales of his products by viewing the graph of ‘summary stats’.

8. Project Security

Keeping security aspects of a software in mind while developing it is very important. An increase in hacking attacks demands a more secure software design, and measures that would avoid such attacks and/or would allow quick recovery from such attacks.

In this section, we have discussed three different software security threats, potential losses that they could cause, and controls that could be implemented to mitigate these threats.

Security Threats

1. Broken Access Control
2. Insecure Design
3. Cryptographic Failures

Potential Losses

1. **Broken Access Control**
    - Unauthorized access to a customer's account can result in unwanted orders being placed by the name of the customer and financial losses to the company
    - Unauthorized access to a merchant's account can result in financial losses to the merchant if their store’s data is tampered
2. **Insecure Design**

    A bad practice in software design is to use security questions for password recovery. Since more than one person can know the answer to such questions, they cannot be trusted. In case of our app, such a design flaw could potentially lead to;

    - Sensitive data loss of merchants as well as buyers
    - Fake orders
    - Fraudulent transactions, etc
3. **Cryptographic Failures**
    - Forensic investigation costs
    - Remediation costs
    - Loss of sensitive information (e.g. industry secrets)
    - Loss of competitive advantage
    - Direct financial losses (e.g. illegitimate financial transactions)
    - Litigation

- Compensation to customers
- Fines
- Loss of reputation
- Loss of business
- Reduction in share price
- Dismissed executives
- Business closing down (as has been the result of some other data breaches)

Security Controls

1. **Broken Access Control**
    - Fingerprint
    - Face ID
    - Complex passwords functionality will help to reduce broken access control (Protective)
2. **Insecure Design**
    - Using OTP code sent via email/sms for password recovery instead of security questions (protective, recovery)
3. **Cryptographic Failures**
    - Avoid using outdated algorithms (protective)
    - Avoid the use of ECB cipher mode (protective)
    - Do not roll your own crypto (protective)
    - Avoid hardcoded crypto keys (protective)

Static and Dynamic Security Scanning Tools

- The **OWASP ZAP** is one of the world's most popular mobile app security testing tools that is free to use and is actively maintained by hundreds of volunteers worldwide. OWASP ZAP helps in finding security vulnerabilities automatically in applications during the development and testing phase. It's also a great tool for pentesters who are experienced enough to use it for manual security testing.
- **Mobile Security Framework** is an automated mobile app security testing tool for Android and iOS apps that is capable of performing static, dynamic analysis and web API testing. MobSF can effectively be used for a quick security analysis of Android & iOS apps. It supports binaries (APK & IPA) and zipped source code
  Read more at: https://www.appknox.com/blog/mobile-app-security-testing-tools

9. Risk Management

Potential Risks and Mitigation Strategies

Sr.	Risk Description	Mitigation Strategy
1.	Staff Illness	The work of the staff can overlap so that upon the unavailability of a member, the group can reorganize and divide the work of the missing member in order to complete it on time.
2.	Product Competition	In order to increase the profits from the project and minimise the effect of competitors, better marketing strategies can be implemented, for example, more attractive discounts and concessions in the case of an e-commerce website.
3.	Time Constraints of Deliverables	The team will manage time and track the progress of each deliverable by obtaining feedback from the members on a daily basis.
4.	Requirement Change Risk	The customer will be informed that if the requirements change in the future, more time will be required to complete the project, as modifying the project requires time.

5.	Economic Risks (Policy Changes of external organisations)	If there is an external policy change, for example, changes in the policies of payment applications (Easypaisa app) and methods that are recommended on the website. The customers will be informed of the updated policies and new policies for the payment option will be designed, for example if easypiasa app introduces a policy of delayed payments, the customers will be required to pay in advance of the product ordered.
6.	Server Risk	If the server used for the project is low-powered and upon increase of website traffic, there are delays, the management will be suggested to purchase a high-powered server.
7.	Skills Risk (Some members not familiar with the tools and technologies being used)	The tools and technologies will be discussed in detail with the staff and time and other group members with experience will be allocated to the members who are not familiar with the tools decided.
8.	End-User Risk (end-users have problems while using the software)	The end-users will be surveyed on a regular basis so that if they are facing difficulties while using the application, the required changes will be made.
9.	Operational Risk (If there is a problem in the working of the software e.g. unexpected crashing of software)	The problem will be identified by the team, and stakeholders will be informed of the unexpected problem. The team will work on solving the problem after giving a possible timeframe to the stakeholders.

10.	Faults in Reusable Components of Software	The reusable components of the software will be kept in check by the team, and upon a fault occurring, the team will rectify it so that it can be reused when required.

10. Testing and Evaluation

Discuss your testing strategy. List down some sample test cases that your created. Moreover, list down the automation tools you used.

We did manual testing. Please see the following link for this part:

Testing and Evaluation.xlsx

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.).

Clone the code from github. Give the yarn command. if it's ios, then install the relevant port. Run in the react environment. create a build for android or ios. To check on laptop, open it on blue stacks and open the apk link you build earlier. The backend is deployed on heroku.

The apk we generated is given below. You can download it its 90 MB approximately and run on android or for laptop run it on blue stacks:

https://drive.google.com/drive/folders/1tvHW9r3gNLOm2eAVdu_9pFrhUaA5H8Iu?usp=sharing

12. Conclusion

a. Summary

The idea of developing an ecommerce platform for our Senior Project was inspired by the fact that the trend of online shopping is increasing in Pakistan and coding such an application will give us hands-on experience of developing commercially used applications. MunasibMall.pk is a mobile application that connects sellers with buyers. Its functionality to cater to the needs of both merchants and buyers has been inspired by Daraz, while it also allows a single merchant to create more than one store for different kinds of products, which is not the case with Daraz.

In terms of development, we divided the use cases among all the group members so that all of us could have an experience of both front and backend development. A lot of documentation was done before developing the actual product so that we could streamline the process. However, in the development phase, the methodology employed was agile and frequent feedback was taken and incorporated from our supervisor, enhancing our product.

The journey of almost one year made us learn a lot. It was the first time for most of our group members to develop a full stack application using MERN(React Native) stack. We also got exposure to industry standards through frequent guest lectures from industry specialists. Team work, time management, and commitment are some other skills that we learnt during this journey.

b. Challenges

There were a lot of technical challenges that we had to face while developing MunasibMall.pk. Firstly, none of the group members had experience of mobile app development before. It was daunting at first to think that our SPROJ, which is very crucial for a fresh grad, would be our first app developed in React Native with no prior experience in the said technology.

Another challenge was to identify potential security threats while creating this app and implement security measures to mitigate some of those threats.

At times there were issues regarding platform because different group members were using different operating systems and were not familiar with containers.

We also had to face some non-technical challenges during the development process. For example, since it was our first time coding a full stack application, we did not know **how things get done**. Frequent feedback from our supervisor, and guest speaker sessions from industry specialists helped a lot in this regard.

We are extremely thankful to Allah Almighty that He helped us overcome these seemingly daunting challenges. These challenges equipped us with skills that would, God willing, help us in our professional lives.

c. Future

MunasibMall.pk is in its most basic form at the moment. Looking at the increased trends in online shopping in Pakistan, we are positive that our product certainly has a market. However, to make it market-ready, more functionality needs to be added to it.

A possible extension of the work could be to create a web application in addition to the mobile app that we created. We recommend that this should be done after the mobile app is ready and can serve as the minimum viable product.

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)
1,2,3	Daniyal Mumtaz
4,5,6	Wali Ullah Aitemad and Muzammil
7,8,9	Abdur Rehman Masood
10,11	Waqar Ul Haq Khatana

14. References

https://www.heroku.com/