BigBasket

A PROJECT REPORT

Submitted By

ARYA N.M

Reg.No:SJC17MCA-011

to

the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the degree

of

MASTER OF COMPUTER APPLICATIONS

Department of Computer Science and Applications

ST.JOSEPH'S COLLEGE OF ENGINERRING ANDTECHNOLOGY

BharananganamPravithanam Road, Kottayam, Palai,

Choondacherry, Kerala 686579

may, 2020

DECLARATION

I undersigned hereby declare that the project report "BigBasket", submitted for partial fulfilment of the requirements for the award of degree of Master of Computer Applications of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by me under supervision of **Mrs. Rinu Mathew**. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

Place: Palai **ARYA N.M**

Date: **Reg.No:SJC17MCA-011**

DEPARTMENT OF COMPUTER SCIENCE AND APPLICATIONS
ST.JOSEPH'S COLLEGE OF ENGINEERING ANDTECHNOLOGY
Palai, Choondacherry Kottayam, 686579
(*Approved by AICTE and affiliated to APJ Abdul Kalam Technological University)*

CERTIFICATE

This is to certify that the report entitled **"BigBasket"** submitted by **"ARYA N.M" , Reg.No:SJC17MCA-011"** to the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the Degree of Master of Computer Applications is a bonafide record of the project work carried out by her under our guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.

Mrs. Rinu Mathew,
Internal Guide

Mr. Alex Jose,
Project Co-ordinator

Dr.T.D.Jainendrakumar
Head of the Department

Viva-voce held on:.....

External Examiner 1:

External Examiner 2:

ACKNOWLEDGEMENT

If words are considered as symbols of approval and tokens of acknowledgment, then let words play the heralding role in expressing my gratitude. To bring something into existence is truly a work of God. I would like to thank God for not letting me down and showing me the silver lining in the dark clouds.

I would like to thank **Dr. J David**, Principal, St. Joseph's College of Engineering & Technology for his support and encouragement. I convey my heartfelt thanks **to Dr. T.D Jainendrakumar** (Head of the Department - Master of Computer Applications, St. Joseph's College of Engineering & Technology,) for providing an opportunity for the project presentation. It is my pleasure to express my gratitude to the project coordinator **Mr. Alex Jose,** Asst.professor, Department of Computer Applications, St. Joseph's College of Engineering & Technology whose support and constructive criticism has led to the successful completion of the task.

With the biggest contribution to this report, I would like to thank **Mrs. Rinu Mathew**, Department of Computer science and Applications who had given me full support in guiding me with stimulating suggestions and encouragement to go ahead in all the time of the this work.

I would also thank my institution and faculty, my family and friends without whom this project would have been a distant reality.

ARYA N.M

ABSTRACT

The products are available quickly and frequently at any store, every store has their set of glossary products which need to be managed properly in such a way that as one customer come to take the product the items can be easily removed and collected at a place, then it needs to be set according to the price.
Here what this system does is in this system we can see the items like wheat, rice, pulses, soap and any other product according to the attributes like type, brand, category, price, etc.

So that using system it can be located quickly and give instructions to staff to take out the products and the total of the goods are presented in the order table to total the price. The bill is made on that basis and given to the customers.

The admin maintains the detail of the clients and staff. Can be the counter manager or any staff. In this way, this gives a sense of management to the store and brisk the pace of work.

The module of the BigBasket is made of the combination of modules which work with collaboration with each other and make it beneficial to accomplish the main aim of the scheme.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

1.1 PROJECT OVERVIEW

The BigBasket enables vendors to set up online shops, customers to browse through the shops and a system administrator to approve the requests for new shops and maintain lists of shop categories.

Also on the agenda is designing an online shopping site to manage the items in the shop and also help customers purchase them online without having to visit the shop physically. Customer user has limited access right to access the system while the admin users have full control over the system. This project is a web based shopping system for an existing shop. The project objective is to deliver the online shopping application into android platform.

Our online shopping site will use the internet as the sole method for selling goods to its consumers. Shopping will be highly personalized and the mall will provide lower prices than most competitors. This, in brief, is a description of our product which will showcase a complete shopping experience in a small package.

BigBasket is the process whereby consumers directly buy goods or services from a seller in real-time, without an intermediary service, over the Internet. It is a form of electronic commerce. The products are available quickly and frequently at any store, every store has their set of grocery products which need to be managed properly in such a way that as one customer come to take the product the items can be easily removed and collected at a place, then it needs to be set according to the price.

So that using system it can be located quickly and give instructions to staff to take out the products and the total of the goods are presented in the order table to total the price. The bill is made on that basis and given to the customers.

This project is an attempt to provide the advantage of online shopping to customers of a real shop. It helps buying the products in the shop anywhere through internet by using an android device. Online shopping is the process whereby consumers directly buy goods or services from a seller in real-time, without an intermediary service, over the Internet.

1.2 ABOUT THE ORGANIZATION

Srishti Innovative is a leading, Technology Services Company for Web, Mobile and Solutions. We are a **Deloitte Technology Fast 50 India Company for 2014** and rank among **Deloitte Technology Fast 500 APAC 2014 and 2014 Red Herring Top 100 for Asia.**We are experts in mobile and web technology and create exceptional, brand led digital experiences. Srishti provides solutions for B2E, B2B and B2C through process driven approach, rapid application development and proven agile work flow methodology.

Founded in 2007, our in-house team of domain experts and tech engineers has delivered 1700+ applications across Web, Cloud and iOS, Android, Windows, Blackberry platforms. We are an ISO 9001:2008 certified company. We have worked with over 850 global clients across key industries - Healthcare, Financial Services, Retail, Travel, Media and Entertainment. We have offices in India and US.

Here at Srishti, we take our watchwords - **"**Quality beyond Contract" very seriously and you will find it exemplified in all realms of our work. We understand that Quality demands Discipline, rigorous Effort, Commitment, Creativity and a Unique Work Methodology. We take a consultative approach to creating tangible value and competitive advantage for our clients.We help you navigate through the labyrinth of technology building blocks like Smart Devices, NFC, Augmented Reality, Proximity-alert and Responsive Design guiding your web and mobile strategy to success while delivering stable, secure and scalable solutions for your business.

Srishti Innovative aims to cut through the noise with pinpoint delivery of solutions and be always relevant to our clients. Our clients are able to increase efficiencies across the board by being more responsive and productive with on the go business readiness. We have years of expertise in designing world class web and mobile solutions tailor-made to your unique business requirements and go-to market strategy. With design and development expertise in diverse platforms, best-of-breed tools and techniques, combined with industry best practices, we offer scalable end-to-end application development solutions.

Our professional and thorough approach to project planning gets your app idea right on track from the word go and maximizes returns on your investment. We bring unparalleled expertise, technological knowhow and industry proven practices to the table driving engagement and higher monetization for our clients. Our strong process framework has evolved out of our rich experience in executing various projects of different sizes, complexity and domains. To be competitive in today's fast moving marketplace, organizations need to drive innovation in every part of their business. As a result, more and more companies are embracing agile work flow as a viable development methodology that produces results and offers customer value faster.

Agile focuses on evolutionary development that is iterative, dynamic and encourages collaboration with the customer.With Agile software development, adaptability and flexibility are more highly valued than heavy documentation or rigid plans and end-users are expected to work closely with developers every step of the way. Whether it's undertaking new application development or transitioning an existing legacy system, Srishti Innovative has the Agile Development experience and expertise you can trust to deliver fast results.

Services:

1.Mobile Application Development

World class mobile applications on iOS, Android, Windows and Blackberry platforms that inspire users, transform businesses and promise engaging experiences.

2.Web Development Solutions & Consulting

Full service web solutions that include mobile optimized websites, eCommerce Solutions and web portals and complex content management system.

3.Enteprise Applications & Mobility Solutions

Unleash the potential of your business with on the go enterprise-class applications and mobility solutions.

4.Cloud Enabling Services

Today's dynamic market conditions require businesses to be more agile and responsive. Cloud technology gives the ability to swiftly develop and deploy IT services and applications to improve the way your business gets done.

CHAPTER 2

SYSTEM ANALYSIS

II.1 INITIAL INVESTIGATION

Preliminary investigation is a problem solving activity that requires interactive communication between the system user and system developer. It does various feasibility studies. In these studies, a rough figure system activities can be obtained from the decision about the strategies to be followed for effective system study and analysis can be taken.

In the preliminary investigation an initial picture about the system working is got from this study and data collection methods are identified. To launch a system investigation we need a master plan detailing the steps to be taken, the people to be questioned and the outcome expected. The scope of preliminary investigation may vary from a brief one person effort to an extensive series of activities requiring the participation may vary from a brief one person effort to an extensive series of activitie or equiring the participation of many individuals.

II.2 EXISTING SYSTEM

The study of the existing system is a prerequisite for developing any software system. The study of the system reveals many features of the existing system. This gives analyst an insight into the working of the system and helps the developer to design an appropriate system, which will eliminate the many limitations present in the existing system. The existing system is manual where the users must have to perform their tasks manually. It will take more time and this whole procedure is very tedious and takes a lot of time.

II.3 PROPOSED SYSTEM

As the proposed system is a Android application where the users can perform all the arrangements easily and efficiently. This system provides a searching facility to the user based on their demands.

To overcome the drawbacks of the existing system, the proposed system has been evolved. The system provides with the best user interface. The efficient reports can be generated by using this proposed system. In this system there are mainly three modules, admin, customer and staff.

The proposed system computerization is developed using SQL server as back-end and Android as admin, user and staff module . This languages is managed, safe environment for application, development and execution. The software is developed as a simulated system and the complex procedures are avoided to make the system easy to use. The proposed system is user friendly and has simplicity and security. In the proposed system the data redundancy can be avoided to certain extend and the data consistency can be main-tained. The record keeping and searching process are easy.

II.3.1 Advantages of the Proposed System

- The proposed system provides accurate data.
- The proposed system is very much faster than existing issue, searching is taking small amount of computerized time.
- Less time consuming and more efficient.
- Simple user interface to reduce processing
- Eliminate chances for errors and reduce effort

II.3.2 Features of the Proposed System

The various features of proposed system are as follows:

- Access to the system and database as per user identification the maximum security ensured.
- Integrity reliability and integrity of data User friendly and flexible in all aspects
- Data entry updates is quite easy
- Effective table manipulation as facilitated by the rich SQL Good validation checking
- Easy maintenance
- Removes chances of leakage of information. Provides a better record keeping system
- All these forms the major aspects and advantages of the proposed system. Provision is made for effective improvements of maintenance are needed at any stage.
- All these form the major aspects and advantages of the proposed system. Provision is made for effective improvements of maintenance are needed at any stage.

II.4 FEASIBILITY STUDY

During system analysis, a feasibility study of the proposed system was carried out to see whether it was beneficial to the organization. The main aim of the feasibility study is to determine whether it would be financially and technically feasible to develop the product. While evaluating the existing system, many advantages and disadvantages raised. Analyzing the problem thoroughly forms the vital part of the system study. Problematic areas are identified and information is collected.

The benefits of this site are users can easily interact and get the services without much complexity. It helps to make it possible that more users can interact with the site at a time. Feasibility study is to determine whether the proposed system is technically, economically and behaviourally feasible in all respects.

The main aim of feasibility study is to evaluate alternative site and propose the most feasible and desirable site for development. If there is no loss for the organization then the proposed

system is considered financially feasible. A feasibility study is carried out to select the best system that meets performance requirements.

The feasibility study activity involves the analysis of the problem and collection of all relevant information relating to the product such as the different data items which would be input to the system, the processing required to be carried out on these data, the output data required to be produced by the system as well as various constraints on the behaviour of the system.

In this scenario, problems are identified. Essential data are being gathered for the existing problems. It is necessary that this analysis familiarizes the designer with objectives, activities, and the function of the organization in which the system is to be implemented. The feasibility study was divided into four:- Technical, Economical, Operational and behavioural. It is summarized below:-

II.4.1 TECHNICAL FEASIBILITY

According to feasibility analysis procedure the technical feasibility of the system is analyzed and the technical requirements such as software facilities, procedure, inputs, are identified. While considering the problems of existing system, it is sufficient to implement the new system. The proposed system can be implemented to solve issues in the existing system. It includes the evaluation of and how it meets the proposed system. This system use Android framework and Mysql as back end technology.

II.4.2 ECONOMIC FEASIBILITY

Economic analysis is most frequent used for evaluating of the effectiveness of the candidate system. More commonly known as cost/benefit analysis the procedure is to determine the benefit and saving that are expected from a candidate system and compare them with the existing system. Except for the initial capital amount and the amount after each financial

year, no other huge amount is needed. The expenses can be handles by any participants. So, the system is economically feasible.

This feasibility involves some questions such as whether the firm can afford to build the system, whether its benefits should substantially exceed its costs, and whether the project has higher priority and profits than other projects that might use the same re- sources. Here there is no problem. This firm has fully equipped hard ware, and fully fledged software, so no need to spend money on these issues. And as the client and the developer are one, there is no further problem in economic issues.

II.4.3 OPERATIONAL FEASIBILITY

Methods of processing and presentation are all according to the needs of clients since they can meet all user requirements here. The proposed system will not cause any problem under any circumstances and will work according to the specifications mentioned. Hence the proposed system is operationally feasible.

People are inherently resistant to change and computer has been known to facilitate changes. The system operation is the longest phase in the development life cycle of a system. So, Operational Feasibility should be given much importance. This system has a user-friendly interface. Thus it is easy to handle.

II.4.4 BEHAVIORAL FEASIBILITY

In today's world, computer is an inevitable entity. As per the definition of behaviour design, many valid points are recognized in this study. This system behaviour changes according to different environment. In order to ensure proper authentication and authorization and security of sensitive data of the admin or employers, login facilities are provided. These are the main feasibility studies tested in this application.

CHAPTER III

SYSTEM ANALYSIS AND DESIGN

III.1 SOFTWARE REQUIREMENT SPECIFICATION

The primary goal of the system analyst is to improve the efficiency of the existing system. For that study of specification of the requirement is very essential. For the development of the new system, a preliminary survey of the existing system will be conducted. An investigation is done whether the up gradation of the system into an application program could solve the problems and eradicate the inefficiency of the existing system. This gives an idea about the system specifications required to develop and install the project ”BigBasket”.

The System Requirements Specification is based on the System Definition. The requirement specifications are primarily concerned with functional and performance aspect of a software product and emphasis are placed on specifying product characteristics implying how the product will provide those characteristics. One of the most difficult tasks is selecting software, once the system requirement is find out then we have to determine whether a particular software package fits for those system requirements. This selection summarizes the application requirement.

III.1.1 HARDWARE REQUIREMENT

- CPU - INTEL(R)PENTIUM(R)
- HARD DISKSPACE - 500 GB
- RAM - 2GB
- DISPLAY - 19 STANDARD RATIO LCDMONITOR
- KEYBOARD - 99-104 KEYS
- CLOCK SPEED - 1.99 GHZ

III.1.2 SOFTWARE REQUIREMENT

- OPERATING SYSTEM – WINDOWS7/810
- WEB SERVER – GODADDY
- FRONT END – ANDROID
- BACK END – SQL

III.2 SYSTEM DESIGN

Designing the system in an effective way leads to the smooth working of any software's. System design is the process of developing specification for a candidate system that meet the criteria established in the system analysis. Major step in the system design is the preparation of the input forms and output reports in a form applicable to the user. The main objective of the system design is to use the package easily by any computer operator. System design is the creative act of invention, developing new inputs, and database, off-line files, method, procedure and output for processing business to meet an organization objective. System design builds information gathered during the system analysis. This system is designed neatly so that user will never get ambiguity while using the system.

III.2.1 NON-FUNCTIONAL REQUIREMENTS

Performance Requirements

For the efficient performance of the application, network must have high bandwidth so that the task of centralized management does not lead to network jam. Also the hard disk capability must be high so that data can be effectively stored and retrieved.

Security Requirements

Security requirements of this application involve authentication using user name and password so that invalid users are restricted from data access. For the security of data,

periodic database backups must be performed so that we can recover data in the case of data loss.

III.3 UNIFIED MODELING LANGUAGE [UML]

UML is a way of visualizing a software program using a collection of diagrams. The notation has evolved from the work of Grady Booch, James Rumbaugh, Ivar JAcobson and the Rational Software Corporation to be used for object-oriented design, but it has since been extended to cover a wider variety of software engineering projects. Today, UML is accepted by the Object Management Group (OMG) as the standard for modeling software development.UML stands for Unified Modeling Language.UML 2.0 helps extend the original UML specification to cover a wider portion of software development efforts including agile practices. Improved integration between structural models like class diagrams and behavior models like activity diagrams. The original UML specified nine diagrams; UML 2.x brings that number up to 13. The four new diagrams are called: communication diagram, composite diagram, interaction overview diagram and timing diagram. It also renamed state chart diagrams to state machine diagrams, also known as state diagrams.

Types of UML diagrams

The current UML standards call for 13 different types of diagrams: class, activity, object, use case, sequence, package, state, component, communication, composite structure, interaction overview, timing and deployment. These diagrams are organized into two distinct groups: structural diagrams and behavioral or interaction diagrams.

Structural UML diagrams

- Class diagram
- Package diagram
- Object diagram
- Component diagram
- Composite structure diagram
- Deployment diagram

Behavioral UML diagrams

- Activity Diagram
- Sequence diagram
- Use case diagram
- State diagram
- Communication diagram
- Interaction overview
- Timing diagram

3.3.1 Use case Diagram

To model a system the most important aspect is capture the dynamic behavior. To modify a bit in details, dynamic behavior of the system when it is running or operating. So only behavior is not sufficient to model a system rather dynamic behavior is more important than static behavior. In UML there are five diagrams available to model dynamic nature and use case diagram is one of them. Now as we have to discuss that the use case diagram is dynamic in nature there should be some internal or external factors for making the interaction. These internal and external agents are known as actors. So use case diagram consists of actors, use case and their relationships. The diagram is used to model the system of an application. A single use case diagram captures a particular functionality of a system.
Use case Diagram objects:

- Actor
- Use case
- System
- Package Actor

Actor

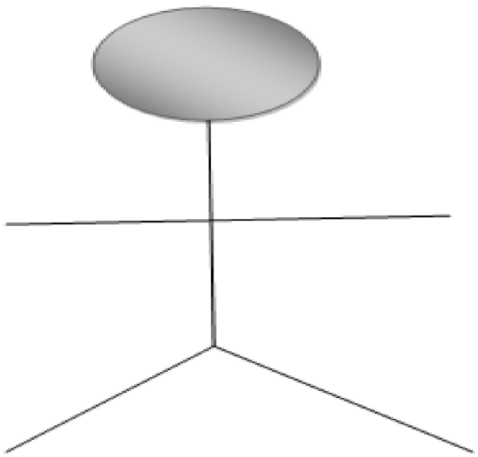

Fig III.1: Actor

Actor is a use case diagram in an entity that performs a role in one given system. This could be a person, organization or an external system usually drawn like skeleton.

Use case

A use case represents a function or an action within the system. It's drawn as an oval and named with the function.

Fig III.2: Use Case

System

System is used to define the scope of the use case and drawn as a rectangle. This is an optional element but useful when your visualizing large systems. For example you can create all the use cases and then use the system object to define the scope covered by your project. Or you can even use it to show the different areas covered in different releases.

Package

Package is another optional element that is extremely useful in complex diagrams. Similar to use class diagrams, packages are used to group together use cases.

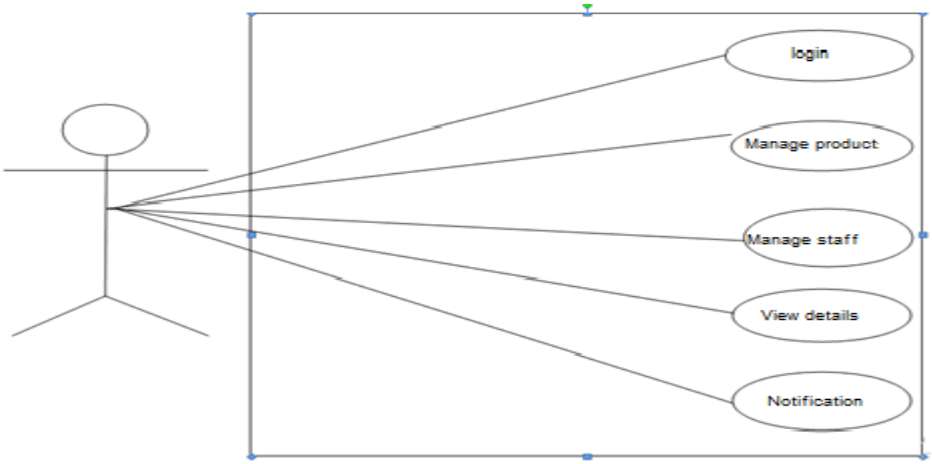

Fig 3.1 Usecase diagram for Shop owner

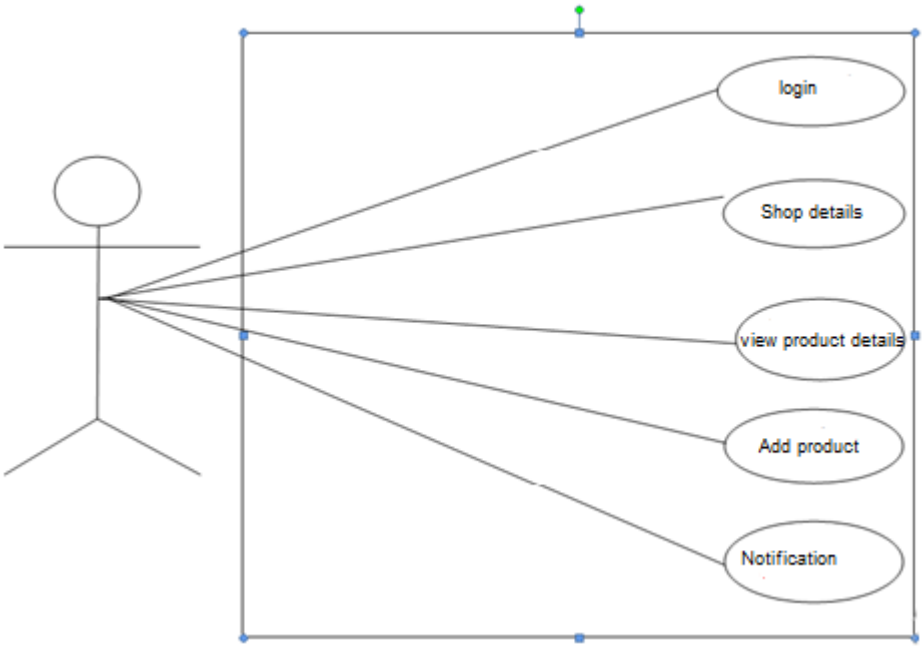

Fig 3.2 Usecase diagram for Customer

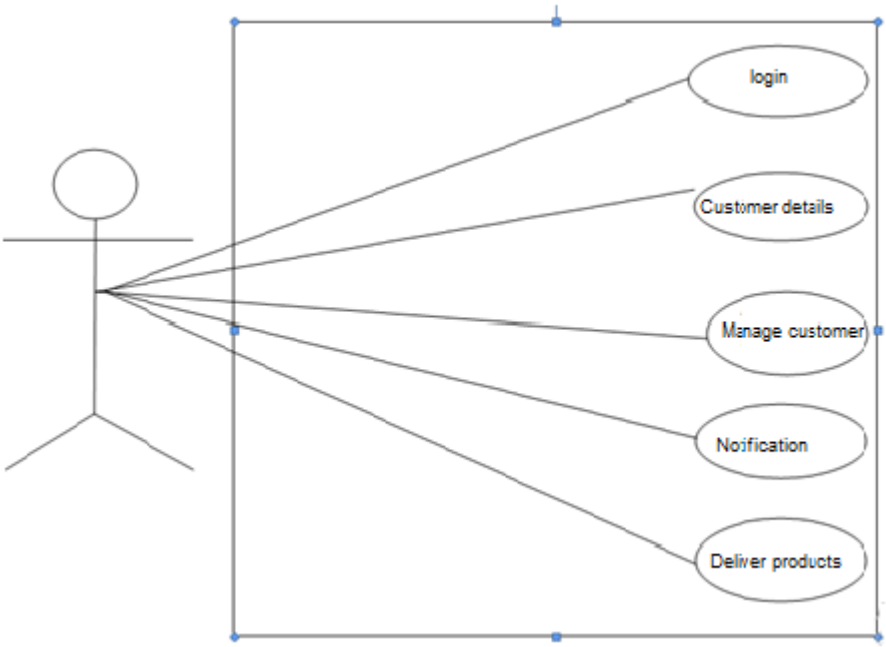

Fig 3.3 Usecase diagram for Staff

III.3.2 Sequence Diagram

UML sequence diagrams are used to represent or model the flow of messages, events and actions between the objects or components of a system. Time is represented in the vertical direction showing the sequence of interaction of the header elements.

Sequence Diagrams are used primarily to design, document and validate the architecture, interfaces and logic of the system by describing the sequence of actions that need to be performed to complete a task. UML sequence diagrams are useful design tools because they provide a dynamic view of the system behavior which can be difficult to extract from static diagrams or specifications.

Although UML sequence diagrams are typically used to describe object-oriented software systems, they are also extremely useful as system engineering tools to design system architectures in business process, as message sequence charts and call flows for telecoms or wireless system design, and for protocol stack design and analysis.

A sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. It is a construct of a message sequence chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence Diagrams are typically associated with use case realizations in the logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

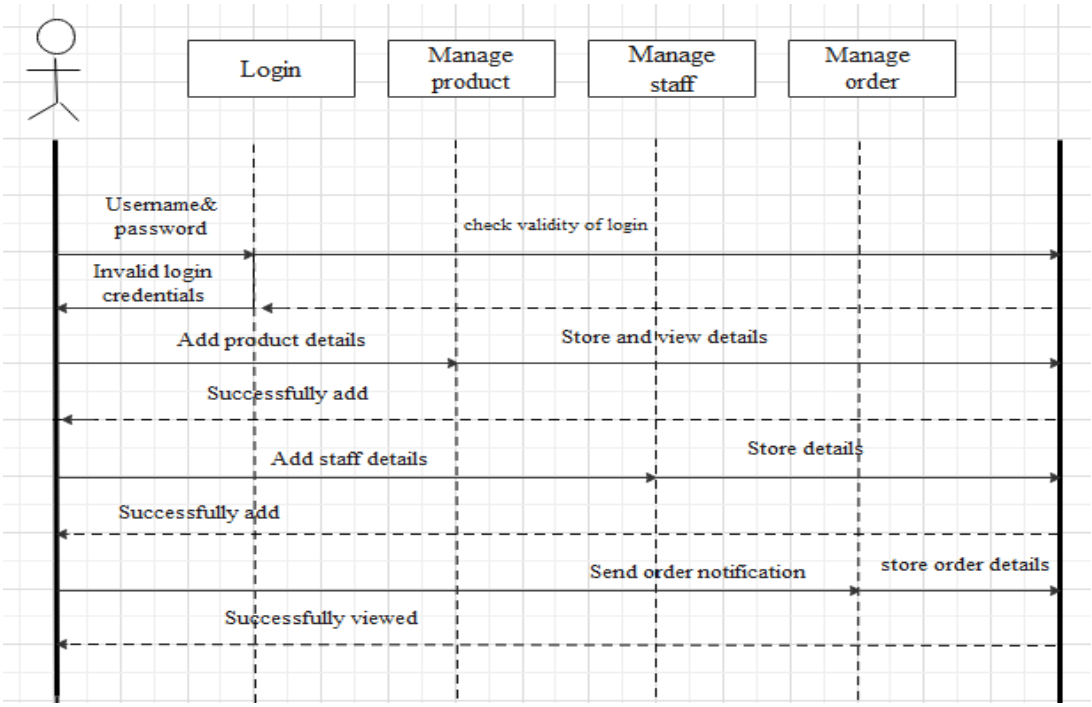

Figure III.1: Sequence diagram for Admin

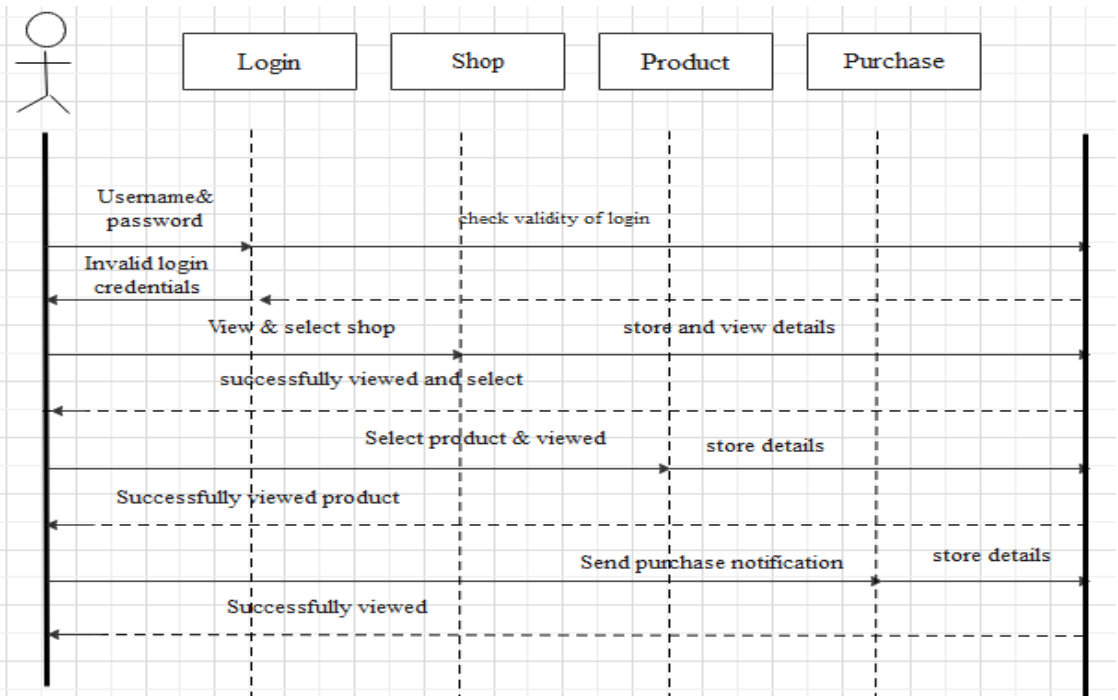

Figure III.2: Sequence diagram for Customer

III.3.3 Activity Diagram

The basic purposes of activity diagrams are similar to other four diagrams. It captures the dynamic behavior of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.

It does not show any message flow from one activity to another. Activity diagram is sometimes considered as the flowchart. Although the diagrams look like a flowchart, they are not. It shows different flows such as parallel, branched, concurrent, and single. Activity diagrams are mainly used as a flowchart that consists of activities performed by the system. Activity diagrams are not exactly flowcharts as they have some additional capabilities. These additional capabilities include branching, parallel flow, swim lane, etc.

Before drawing an activity diagram, we must have a clear understanding about the elements used in activity diagram. The main element of an activity diagram is the activity itself. An activity is a function performed by the system. After identifying the activities, we need to understand how they are associated with constraints and conditions.
Before drawing an activity diagram, we should identify the following elements –

- Activities
- Association
- Conditions
- Constraints

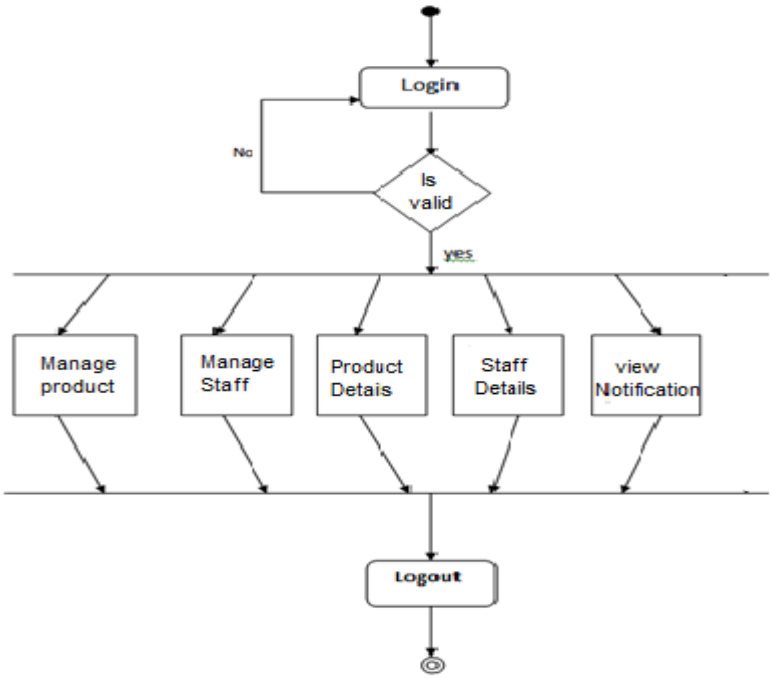

Figure III.3: Activity diagram for Admin

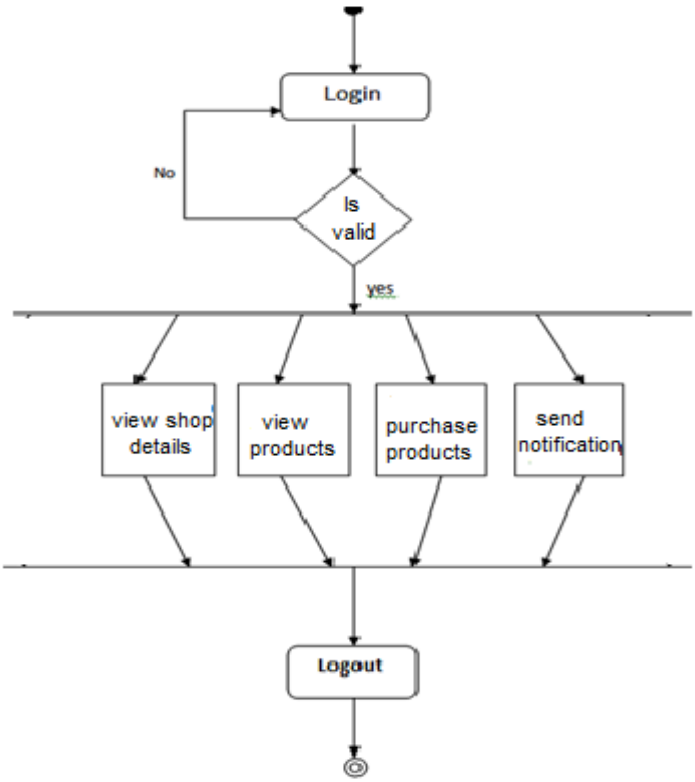

Figure III.4: Activity diagram for Customer

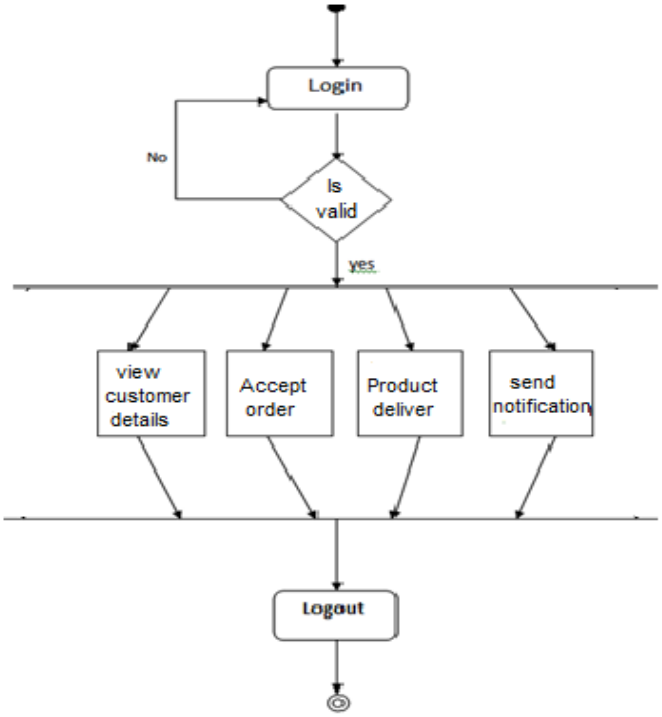

Figure III.5: Activity diagram for Staff

III.4 SYSTEM DESIGN

The most creative and challenging phase of the system life cycle is the system design. The term design describes a final system and the process by which it is developed. It refers to the technical specification that will be applied in implementing the candidate system. In system design, we move from the logical to the physical aspects of the life cycle.

The first step is to determine how the output is to be produced and in what format. Then input data and master files have to be designed as the next step and finally the impact of the candidate system on the user and organization are documented and evaluated by the management. After identifying the problem and the limitation of the existing system, a detailed design of the proposed system is conducted.

Free flow personnel interview and reference to previous records prepared manually were the only methods taken to collect necessary information. At present, all organizations are on the path of computerization process. Design is the phase that indicates the final system. It is the solution, the translation of requirements into ways of meeting them. In this phase the following elements were designed namely, data flow, data stores, processes, procedures was

formulated in a manner that meet the project requirements. After logical design physical construction of the system is done.

The database tables, input screens, output screens, output reports are designed. After analyzing the various functions involved in the system the database, labels as dictionaries designed. Care is taken for the field name to be in self-explanatory form. Unnecessary fields are avoiding so as not affecting the storage system. Care must be taken to design the input screen in the most user-friendly way so as to help even the novice users to make entries approximately in the right place. This is being accomplished by the use of giving online help messages, which are brief and cleanly prompts users for appropriate action.

Design is the only way that we can accurately translate a customer's requirements into a finished software product or system. Without design, risk of building an unstable system exist one that will fail when small changes are made, one that will be difficult to test.
All input screens in the system are user friendly and are designed in such a way that even a layman can operate. The sizes of all screens are standardized. Reports generated in this software give the finer accepts of the required information, which helps in taking vital decision.

The importance of the software design can be stated with a single word quality. Design is a place where quality is fostered in software development. Design is the only way where requirements are actually translated into a finished software product or system.

Mainly this project consists of 3 Modules:

- **Admin**
- **Customer**
- **Staff**

Admin Module

Administrator is the main actor in this system. He has the entire control of the system which includes adding all the details to generate the reports .Brief description about the functionalities performed by the admin is given below. After the admin successfully login to this website the admin can perform the functionalities including:

Admin Login

- Login
- Product details
- Staff details
- Manage staff
- Send notifications to staff

Customer

- Login
- View shop
- View product
- Purchase product
- Send notifications to shop owner

Staff

- Login
- Customer details
- Accept the order
- Deliver the products to the customer
- Send notifications to shop owner

III.4.1 Input Design

Input design is the process of converting a user oriented description of the inputs to a computer based business system into a programmer oriented specification. The design decision for handling input specify how data are accepted for computer processing. Input design is a part of overall design that needs careful attention. The collection of input data is considered to be the most expensive part of the system design. Since the inputs have to be planned in such a way so as to get the relevant information, extreme care is taken to obtain the pertinent information. If the data going into the system is incorrect then the processing and outputs will magnify these errors. The goal of designing input data is to make data entry as easy, logical and free from errors as possible. The following are the objectives of input design:

- To produce a cost effective method of input.
- To ensure validation

Effort has been made to ensure that input data remains accurate from the stage at which it is recorded and documented to the stage at which it is accepted by the computer. Validation procedures are also present to detect errors in data input, which is beyond control procedures. Validation procedures are designed to check each record, data item or field against certain criteria.

In my proposed system Image Compression, data has to be accurate and complete. If not, error messages are displayed to the user and he is unable to proceed to the next stage of action unless he corrects his data. Duplicate entries are not allowed. The data validation, a procedure of the proposed system, provides program checks for the completeness, consistency, reasonableness and sequence of the system.

Maximum care has been taken to ensure that user types in only minimum data into the system, as all he/she will have to do is to move and click the mouse or strike a key to select the desired data at the desired position.

The screens are designed in such a way that the user can find the needed like options, actions etc. with ease of use. The needed columns, where interaction is needed, like labels, buttons are also simple. The related data columns are clubbed together as groups, so that the user can understands the related data easily.

The input design is the link between the information system and the user. It comprises developing specifications and procedures for data preparation and those steps that are necessary to put input data into a usable form for processing data entry.

The design of inputs focuses on controlling the amount of inputs required, controlling errors, avoiding delay, avoiding extra steps and keeping the process simple.

III.4.2 Data Flow Diagram

Data flow diagram is the graphical representation of the system. It is a network that uses special symbols to describe the flow of data and process that transforms data throughout the system.

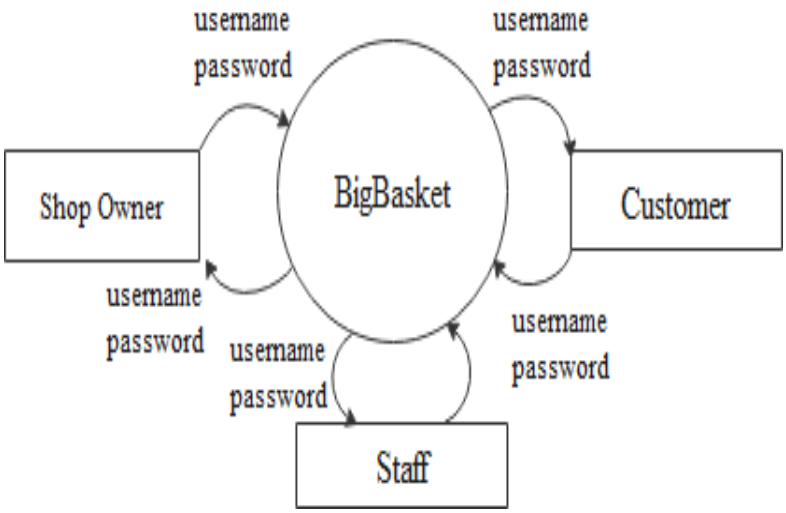

Figure III.6: Level 0 DFD

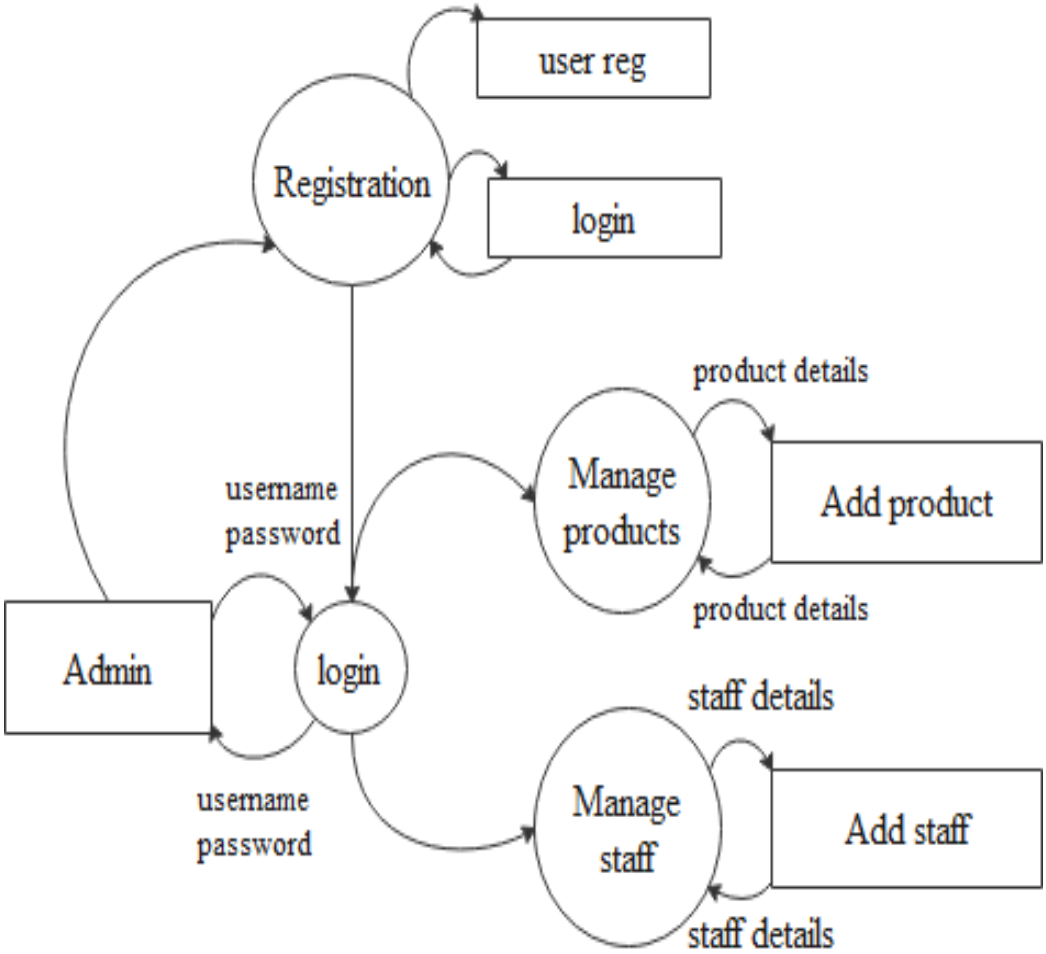

Figure III.7: Level 1 DFD

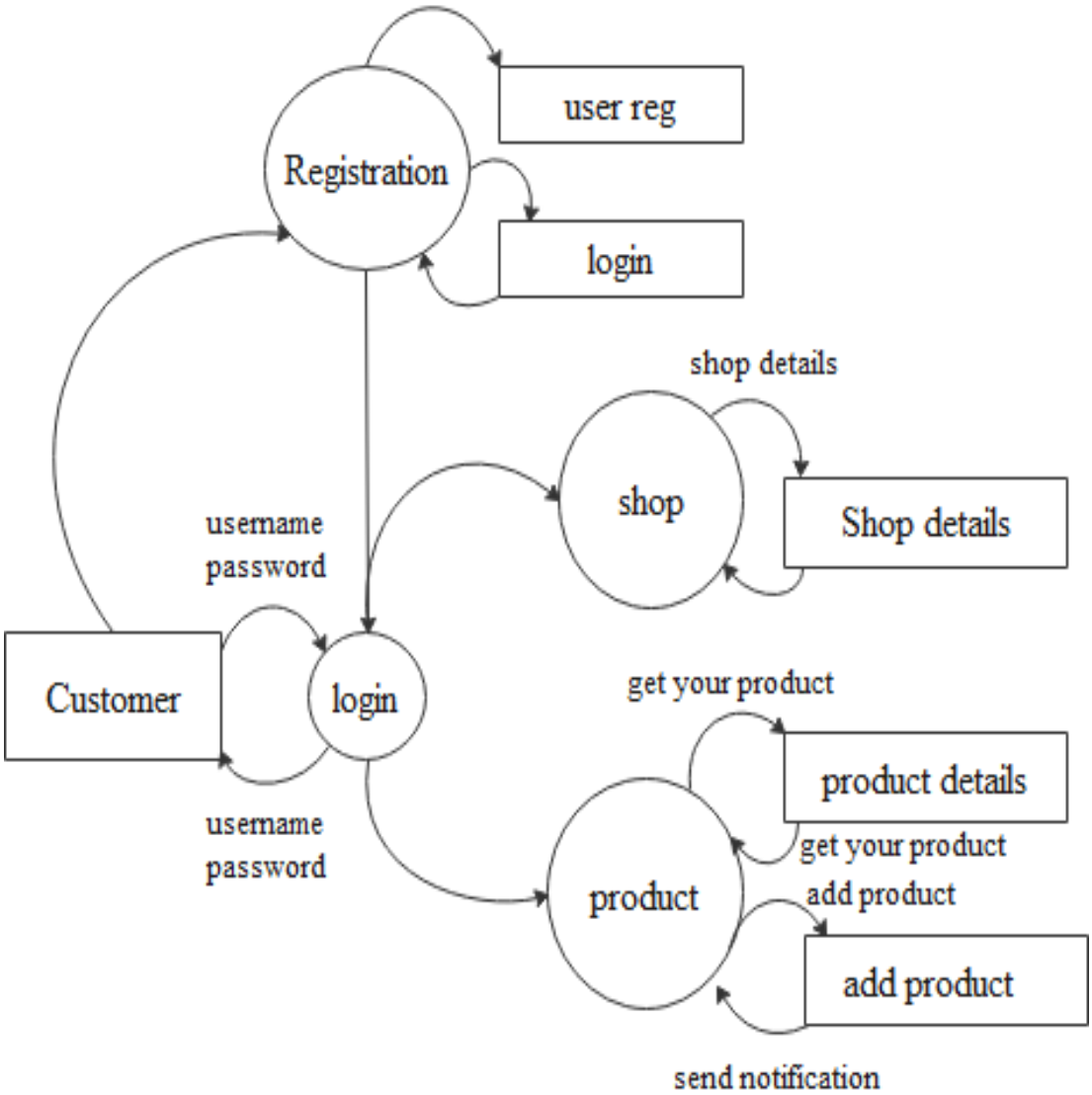

Figure III.8: Level 2 DFD

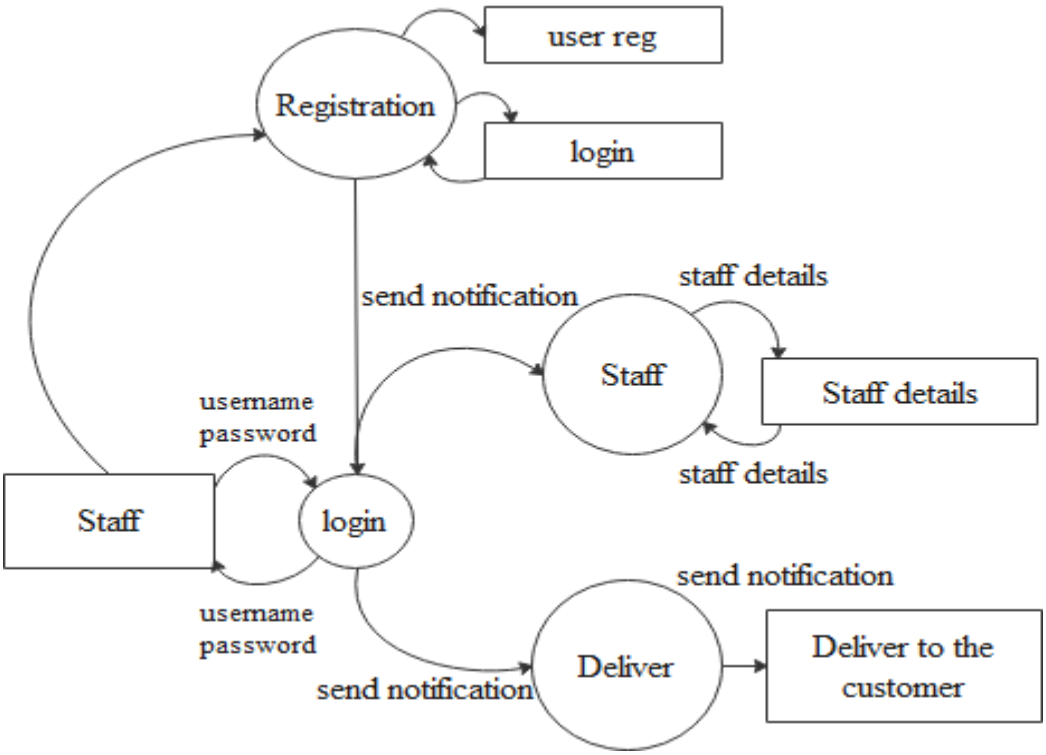

Figure III.9: Level 3 DFD

III.4.3 Output Design

The output design phase of the system design is concerned with the conveyance of information to the end users in user friendly manner. The output design should be efficient, intelligible so that the system relationship with the end user is improved and thereby enhancing the process of decision making. The output design is an ongoing activity almost from the beginning of the project, efficient and well defined output design improves the relation of the system and the user. The primary considerations in the design of the output are the requirement of the information and the objective of the end user. There are various types of outputs required by most of the systems, but outputs of Image Compression are purely interactive outputs which involve the user in communicating with the computer.

The system output may be of any of the following

- A report
- A document
- A message

The output design specification is made in such a way that it is unambiguous and comprehensive. The approach to output design is very dependent on the type of output and nature of data. Special attention has to be made to data editing. The choice of appropriate output medium is also an important task. The output designed must be specified and documented, data items have to be accurately defined and arranged for clarity. The layout of the output will be normally specified on a layout chart. The final design layout must be approved by the user, communicated in detailed to the programmer. The user's requirements are quite different from that of the programmer. Before preparing a specification for the programmer, it is prudent to ensure that the design is acceptable to the user.

III.4.4 Database Design

TABLES

ADMIN LOGIN

Column name	Data type	Description
Id	Int	Id of Admin
Name	Varchar(20)	Name of Admin
Email	Varchar(50)	Email of the admin
Phone	Varchar(10)	Phone
Shop_name	Varchar(200)	Shop_name
Building_address	Varchar(2000)	Building_address
Password	Varchar(30)	password
Device_token	Varchar(1000)	Device_token

Primary key: Id

PRODUCT

Column name	Data type	Description
Product_id	Int	Id of product
Product_name	Varchar(500)	Name of product
Quantity	Varchar(1000)	Quantity of product
Brand	Varchar(500)	Brand name
Price	Int	Price of the product
Rake_no	Varchar(500)	Rake number
Photo	Varchar(500)	Product Photo
Id	Int	Id of admin

Primary key: product_id

CUSTOMER DETAILS

Column name	Data type	Description
Customer_id	Int	Id of customer
Name	Varchar(500)	Customer Name
Email	Varchar(500)	Email of the customer
Phone	Varchar(500)	Phone number of customer
Password	Varchar(500)	Password
Device_token	Varchar(1000)	Device_token

Primary key: Customer_id

ORDER DETAILS

Column name	Data type	Description
Id	Int	Id of Admin
Order_id	Int	Id of order details
Product_id	Int	Id of product details
Customer_id	Int	Id of customer details
Price	Int	Price of the product
Quantity	Varchar(200)	Quantity
Flag	Int	Flag

Primary key: order_id

STAFF DETAILS

Column name	Data type	Description
Staff_id	Int	Id of staff
Name	Varchar(500)	Name of the staff
Photo	Varchar(500)	Staff photo
Emp_id	Varchar(500)	Employee id
Phone	Varchar(200)	Phone
Pin	Varchar(50)	Pin
Id	Int	Id of admin
Status	Varchar(100)	Current Status
Device_token	Varchar(1000)	Device_token

Primary key: Staff_id

III.5 TOOLS AND PLATFORMS

III.5.1 Android

Android is a mobile operating system developed by Google, based on the Linux kernel and designed primarily for touch screen mobile devices such as smart phones and tablets. Android's user interface is mainly based on direct manipulation, using touch gestures that loosely correspond to real-world actions, such as swiping, tapping and pinching, to manipulate on-screen objects, along with a virtual keyboard for text input. Android's source code is released by Google under an open source license, although most Android devices ultimately ship with a combination of free and open source and proprietary software, including proprietary software required for accessing Google services. Applications ("apps"), which extend the functionality of devices, are written using the Android software development kit (SDK) and often, the Java programming language. The SDK includes a comprehensive set of development tools, including a debugger, software libraries, a handset emulator based on QEMU, documentation, sample code, and tutorials. Initially, Google's supported integrated development environment (IDE) was Eclipse using the Android Development Tools (ADT) plugin. Android has a growing selection of third-party applications, which can be acquired by users by downloading and installing the application's APK (Android application package) file, or by downloading them using an application store program that allows users to install, update, and remove applications from their devices. Google Play Store is the primary application store installed on Android devices that comply with Google's compatibility requirements and license the Google Mobile Services software.

Features

- Auto correction and dictionary
- Voice based features
- Multitouch
- Multitasking
- Screen capture
- Multiple language support
- Accessibility

Android Environment:

Returns the current state of the primary shared/external storage media. Returns the current state of the shared/external storage media at the given path. Return root of the "system" partition holding the core **Android** OS. Return root directory where all external storage devices will be mounted.

Android Language:

Android's user interface is mainly based on direct manipulation, using touch gestures that loosely correspond to real-world actions, such as swiping, tapping and pinching, to manipulate on-screen objects, along with a virtual keyboard for text input. Android's source code is released by Google under an open source license, although most Android devices ultimately ship with a combination of free and open source and proprietary software, including proprietary software required for accessing Google services. Applications ("apps"), which extend the functionality of devices, are written using the Android software development kit (SDK) and often, the Java programming language.

Android tools:

Android SDK Platform-Tools is a component for the Android SDK. It includes tools that interface with the Android platform, such as a db, fastboot and systrace. These tools are required for Android app development. They're also needed if you want to unlock your device bootloader and flash it with a new system image. Android developers are able to churn out all these apps much faster and more efficiently thanks to a profusion of sophisticated, integrated development environments (IDEs) as well as other time-saving tools and applications. Flow Up allows you to monitor the performance of all your production apps. Handy dashboards let you keep track of your stats and metrics, including CPU and disk usage, memory usage, frames per second, bandwidth, and more.

Flow Up is a monthly subscription-based solution with pricing determined by the total number of users in the company.

Android Studio is known as the **best** and fastest **tool** to **develop** an **Android** operating system based applications. Developers are enabled to download this **tool** on Windows, macOS, and Linux based operating systems. It is developed by Google and JetBrains and written in Java, Kotlin, C++.

Android Application:

Although an Android app can be made available by developers through their websites, most Android apps are uploaded and published on the Android Market, an online store dedicated to these applications. The Android Market features both free and priced apps.

Android apps are written in the Java programming language and use Java core libraries. They are first compiled to Dalvik executables to run on the Dalvik virtual machine, which is a virtual machine specially designed for mobile devices.

Developers may download the Android software development kit (SDK) from the Android website. The SDK includes tools, sample code and relevant documents for creating Android apps.

Android activity

If you have worked with C, C++ or Java programming language then you must have seen that your program starts from **main()** function. Very similar way, Android system initiates its program with in an **Activity** starting with a call on onCreate() callback method.

Android XML

XML stands for Extensible Mark-up Language.XML is a very popular format and commonly used for sharing data on the internet. This chapter explains how to parse the XML file and extract necessary information from it.

Android provides three types of XML parsers which are **DOM, SAX and XMLPullParser**. Among all of them android recommend XMLPullParser because it is efficient and easy to use. So we are going to use XMLPullParser for parsing XML.

The first step is to identify the fields in the XML data in which you are interested in.For example. In the XML given below we interested in getting temperature only.

The method **getEventType** returns the type of event that happens. eg: Document start , tag start etc. The method **getName** returns the name of the tag and since we are only interested in temperature, so we just check in conditional statement that if we got a temperature tag , we call the method **getAttributeValue** to return us the value of temperature tag.

Android UI layout

The basic building block for user interface is a **View** object which is created from the View class and occupies a rectangular area on the screen and is responsible for drawing and event handling. View is the base class for widgets, which are used to create interactive UI components like buttons, text fields, etc.

The **ViewGroup** is a subclass of **View** and provides invisible container that hold other Views or other View Groups and define their layout properties.

At third level we have different layouts which are subclasses of ViewGroup class and a typical layout defines the visual structure for an Android user interface and can be created either at run time using **View/ViewGroup** objects or you can declare your layout using simple XML file **main_layout.xml** which is located in the res/layout folder of your project.

Android Fragment

You create fragments by extending **Fragment** class and You can insert a fragment into your activity layout by declaring the fragment in the activity's layout file, as a **<fragment>** element.

Prior to fragment introduction, we had a limitation because we can show only a single activity on the screen at one given point in time. So we were not able to divide device screen and control different parts separately. But with the introduction of fragment we got more flexibility and removed the limitation of having a single activity on the screen at a time. Now we can have a single activity but each activity can comprise of multiple fragments which will have their own layout, events and complete life cycle.

Following is a typical example of how two UI modules defined by fragments can be combined into one activity for a tablet design, but separated for a handset design.

Android content provider

Content providers let you centralize content in one place and have many different applications access it as needed. A content provider behaves very much like a database where you can query it, edit its content, as well as add or delete content using insert(), update(), delete(), and query() methods. In most cases this data is stored in an **SQlite** database.

A content provider is implemented as a subclass of **ContentProvider** class and must implement a standard set of APIs that enable other applications to perform transactions.

Android Intent - Filter

An Android **Intent** is an abstract description of an operation to be performed. It can be used with **startActivity** to launch an Activity, **broadcastIntent** to send it to any interested BroadcastReceiver **startService(Intent)** or **bindService(Intent, ServiceConnection, int)** to communicate with a background Service.

Intend objects

An Intent object is a bundle of information which is used by the component that receives the intent as well as information used by the Android system.

An Intent object can contain the following components based on what it is communicating or going to perform

Action

This is mandatory part of the Intent object and is a string naming the action to be performed — or, in the case of broadcast intents, the action that took place and is being reported. The action largely determines how the rest of the intent object is structured. The Intent class defines a number of action constants corresponding to different intents. Here is a list of The action in an Intent object can be set by the setAction() method and read by getAction().

Extras

This will be in key-value pairs for additional information that should be delivered to the component handling the intent. The extras can be set and read using the putExtras() and getExtras() methods respectively.

Component Name

This optional field is an android **ComponentName** object representing either. Activity, Service or Broadcast Receiver class. If it is set, the Intent object is delivered to an instance of the designated class otherwise Android uses other information in the Intent object to locate a suitable target.

The component name is set by setComponent(), setClass(), or setClassName() and read by getComponent().

Explicit Intent

Explicit intent going to be connected internal world of application,suppose if you wants to connect one activity to another activity, we can do this quote by explicit intent, below image is connecting first activity to second activity by clicking button. These intents designate the target component by its name and they are typically used for application.

Implicit Intent

These intents do not name a target and the field for the component name is left blank. Implicit intents are often used to activate components in other applications. You have seen how an Intent has been used to call an another activity. Android OS uses filters to pinpoint the set of Activities, Services, and Broadcast receivers that can handle the Intent with help of specified set of action, categories, data scheme associated with an Intent.

Android Libraries

This category encompasses those Java-based libraries that are specific to Android development. Examples of libraries in this category include the application framework libraries in addition to those that facilitate user interface building, graphics drawing and database access. A summary of some key core Android libraries available to the Android developer is as follows −

- android.app − Provides access to the application model and is the cornerstone of all Android applications.
- android.content − Facilitates content access, publishing and messaging between applications and application components.
- android.database − Used to access data published by content providers and includes SQLite database management classes.
- android.opengl − A Java interface to the OpenGL ES 3D graphics rendering API.
- android.os − Provides applications with access to standard operating system services including messages, system services and inter-process communication.
- android.text − Used to render and manipulate text on a device display.
- android.view − The fundamental building blocks of application user interfaces.
- android.widget − A rich collection of pre-built user interface components such as buttons, labels, list views, layout managers, radio buttons etc.
- android.webkit − A set of classes intended to allow web-browsing capabilities to be built into applications.

Having covered the Java-based core libraries in the Android runtime, it is now time to turn our attention to the C/C++ based libraries contained in this layer of the Android software stack.

Android Runtime

This is the third section of the architecture and available on the second layer from the bottom. This section provides a key component called **Dalvik Virtual Machine** which is a kind of Java Virtual Machine specially designed and optimized for Android.

The Dalvik VM makes use of Linux core features like memory management and multi-threading, which is intrinsic in the Java language. The Dalvik VM enables every Android application to run in its own process, with its own instance of the Dalvik virtual machine.

The Android runtime also provides a set of core libraries which enable Android application developers to write Android applications using standard Java programming language.

CHAPTER IV
SYSTEM TESTING

IV.1 TESTING METHODOLOGIES AND STRATEGIES

Software testing is an integral part of to ensure software quality, some software organizations are reluctant to include testing in their software cycle, because they are afraid of the high cost associated with the software testing .There are several factors that attribute the cost of software testing. Creating and maintaining large number of test cases is a time consuming process. Furthermore, it requires skilled and experienced testers to develop great quality test cases.

Even with the wide availability of automation tools for testing, the degree of automation mostly remains at the automated test script level and generally significant amount of human intervention is required in testing. In addition data collected, as testing is conducted provides a good indication of software quality as a while. The debugging process is the most unpredictable part of testing process. Testing begins at the module level and work towards the integration of entire computer based system. No testing is completed without verification and validation part.

The goal of verification and validation activities are to access and improve the quality of work products generated during the development and modification of the software. Testing plays a vital role in determining the reliability and efficiency of the software and hence is very important stage in software development. Tests are to be conducted on the software to evaluate its performance under a number of conditions. Ideally, it should do so at the level of each module and also when all of them are integrated to form the completed system.

In the project ”BigBasket” the testing has been successfully handled with the modules. The test data was given to each and every module in all respect and got the desired output. Each module that has been tested is found working properly.

IV.1.1 Unit Testing

Here we test each module individually and integrated the overall system. Unit testing focuses verification efforts even in the smallest unit of software design in each module. This is known as ”module testing”. The modules of the ”OEMS” are tested separately. This testing is carried out in the programming style itself. In this testing each module is focused to work satisfactorily as regard to expected output from the module. There are some validation checks for the fields. Unit testing gives stress on the modules of ”OEMS” independently of one another, to find errors. Different modules are tested against the specifications produced during the design of the modules. Unit testing is done to test the working of individual modules with test servers. Program unit is usually small enough that the programmer who developed it can test it in a great detail. Unit testing focuses first on that the modules to locate errors. These error are verified and corrected and so that the unit perfectly fits to the project.

IV.1.2 Integration Testing

Data can be lost across an interface, one module can have an adverse effect on the other sub-functions, when combined they may not perform the desired functions. Integrated testing is the systematic testing to uncover the errors within the interface. This testing is done with simple data and the developed system has run successfully with this simple data. The need for integrated system is to find the overall system performance. The Modules of this project are connected and tested.

After splitting the programs into units, the units were tested together to see the defects between each module and function. It is testing to one or more modules or functions together with the intent of finding interface defects between the modules or functions. Testing completed at as part of unit or functional testing, integration testing can involve

putting together of groups of modules and functions with the goal of completing and verifying meets the system requirements.

IV.1.3 system Testing

System testing focuses on testing the system as a whole. System Testing is a crucial step in Quality Management Process. In the Software Development Life Cycle, System Testing is the first level where the System is tested as a whole. The System is tested to verify whether it meets the functional and technical requirements. The application/System is tested in an environment that closely resembles the production environment where the application will be finally deployed.

The perquisites for System Testing are:-

- All the components should have been successfully Unit Tested.
- All the components should have been successfully integrated.
- Testing should be completed in an environment closely resembling the production environment. When necessary iterations of System Testing are done in multiple environments.

IV.1.4 User Acceptance Testing

The system was tested by a small client community to see if the program met the requirements defined the analysis stage. It was fond to be satisfactory. In this phase, the system is fully tested by the client community against the requirements defined in the analysis and design stages, corrections are made as required, and the production system is built. User acceptance of the system is key factor for success of the system.

TEST CASE

Test Step	Expected Result	Actual Result	Status
Click on the Login button without entering user name or password	Messages like "Please enter User Name" and "Please Enter Password" should appear.	Messages "Please enter User Name" and "Please Enter Password" appear.	Pass
Enter a non-existing user name password and click on the Login button	Message like "Invalid User Name" should appear	A message "Invalid User Name" appears	Pass
Enter a valid user name but wrong password and click on the Login button	Message like "Wrong Password" should appear	A message "Wrong Password" appears	pass
Enter a valid user name and password and click on the Login button	The page should be navigated to the home page	The page is navigated to the home page	pass

Test case for login page

Test Step	Expected Result	Actual Result	Status
Enter all fields and click Register button	The page should navigated to the Login page	The page is navigated to the login page	Pass
Enter all fields, but some fields are invalid	Message like "Invalid entry"	A message "Invalid Entry" appears	Pass
Click on Register button without filling all fields that are required	Messages like "Please fill all required fields" should appear.	A message "Please fill all required fields" appears	pass

Test case for registeration page

Test Step	Expected Result	Actual Result	Status
Enter all fields with valid entries & click SUBMIT button	Message like "Successfully Loaded" & The page should refresh	A message "Successful!!" appears and page is refreshed	Pass
Enter fields with invalid entries	Message like "Invalid entries" should appear	A message "Invalid Entry" appears	Pass
No fields are filled but click SUBMIT button	Message like "Fill all the fields" should appear	A message "Fill all required fields" appears	Pass

Test case for product page

CHAPTER V

SYSTEM IMPLEMENTATION

The implementation is one phase of software development. Implementation is that stage in the project where theoretical design is turned into working system. Implementation involves placing the complete and tested software system into actual work environment. Implementation is concerned with translating design specification with source code. The primary goal of implementation is to write the source code to its specification that can be achieved by making the source code clear and straight forward as possible. Implementation means the process of converting a new or revised system design into operational one. The three types of implementation are:-implementation of a computerized system to replace a manual system, implementation of a new system to replace existing one and implementation of a modified system to replace an existing one.

The implementation is the final stage and it is an important phase. It involves the individual programming; system testing, user training, and the operational running of developed proposed system that constitute the application subsystem. The implementation phase of the software development is concerned with translating design specification in the source code. The user tests the developed system and the changes are according to the needs. Before implementation, several tests have been conducted to ensure no errors encountered during the operation. The implementation phase ends with an evaluation of the system after placing it into operation of time. The validity and proper functionality of all the modules of the developed application is assured during the process of implementation. Implementation is the process of assuring that the information system is operational and then allowing user to take over its operation for use and evaluation. Implementation is the stage in the project where the theoretical design is turned into a working system. The implementation phase constructs, installs and operated the new system. The most crucial stage in achieving a new successful system is that it works effectively and efficiently.

CHAPTER VI

CONCLUSION

In conclusion, BigBasket has to do with making appropriate effort to stop the rising problem

to all manual supermarket operation in order to enhance the operation of such supermarket. In this project, the software or system that can be used to aid all supermarkets that is still operating manually have been successfully developed.

Technology has made significant progress over the years to provide consumers a better online shopping experience and will continue to do so for years to come. With the rapid growth of products and brands, people have speculated that online shopping will overtake in-store shopping.

While this has been the case in some areas, there is still demand for brick and mortar stores in market areas where the consumer feels more comfortable seeing and touching the product being bought. Online shopping is the best way to purchase any item but be careful because there may be some take products on different sites.

However, the availability of online shopping has produced a more educated consumer that can shop around with relative ease without having to spend a large amount of time. In exchange, online shopping has opened up doors to many small retailers that would never be in business if they had to incur the high cost of owning a brick and mortar store. At the end, it has been a win-win situation for both consumer and sellers.

REFERENCES

1.https://www.tutorialspoint.com/android/index.htm

2. https://www.w3adda.com/android-tutorial

3. https://xhtml2pdf.readthedocs.io/en/latest/

4. https://www.w3schools.com/

5. https://stackoverflow.com/

APPENDIX A

A.1 SCREEN SHOTS

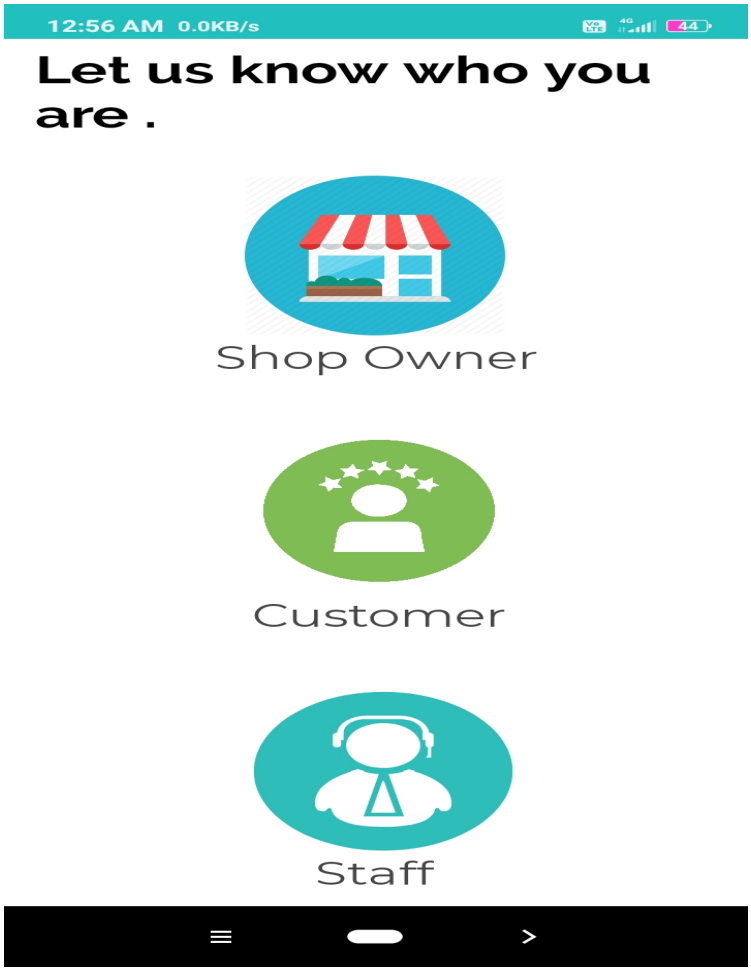

Figure A.1: Home Page

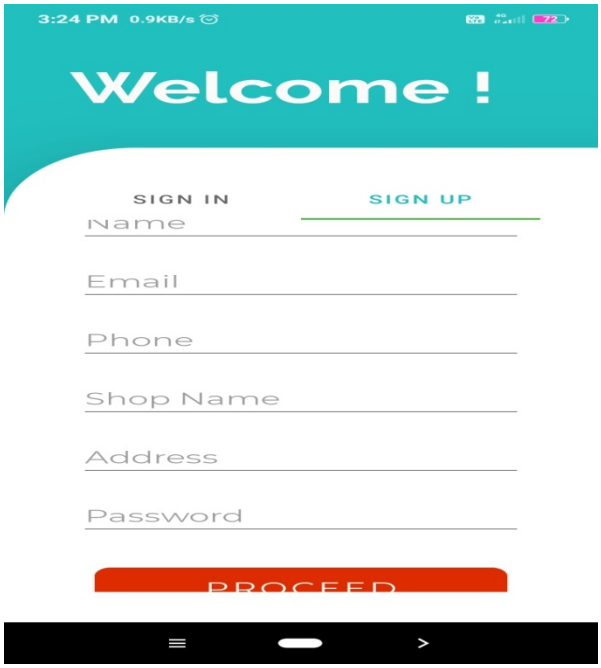

Figure A.2: Shop owner signup page

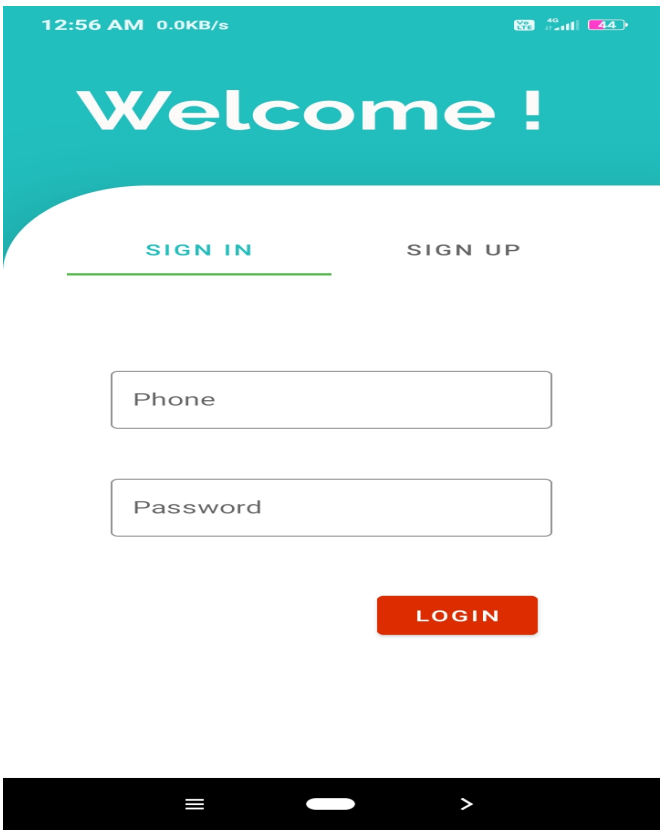

Figure A.3: Shop owner login

Figure A.4: Add product page

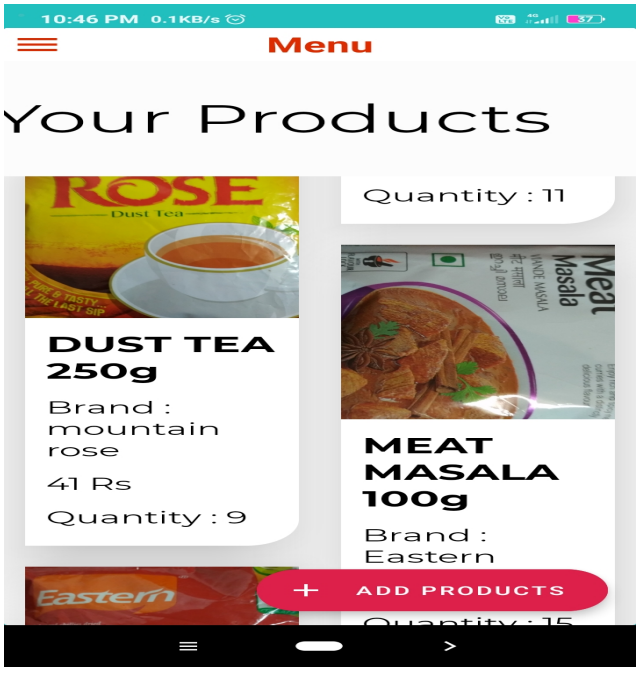

Figure A.5: Product page

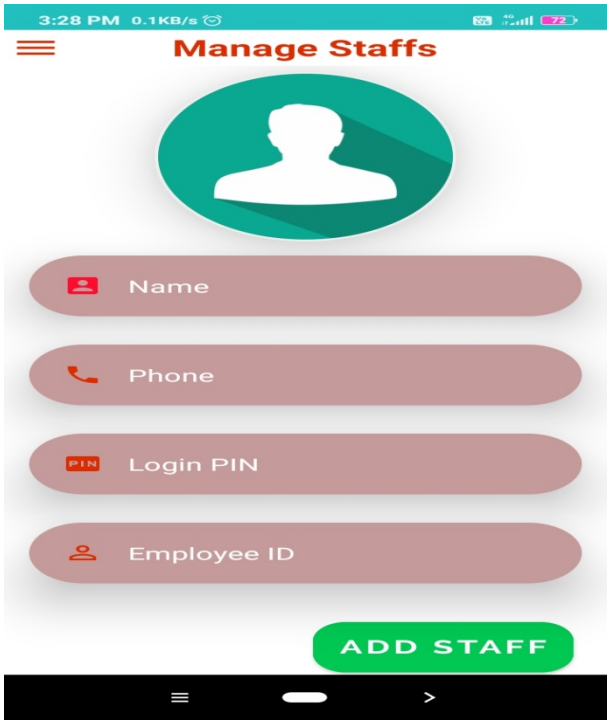

Figure A.6: Add staff page

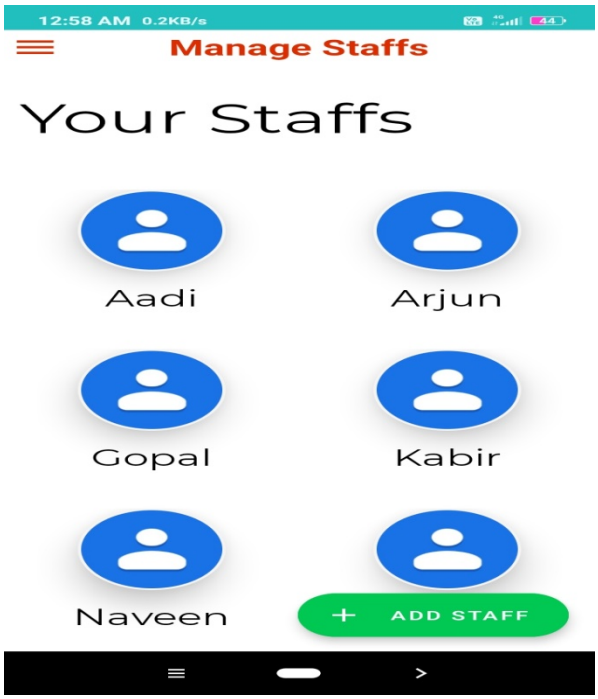

Figure A.7: staff page

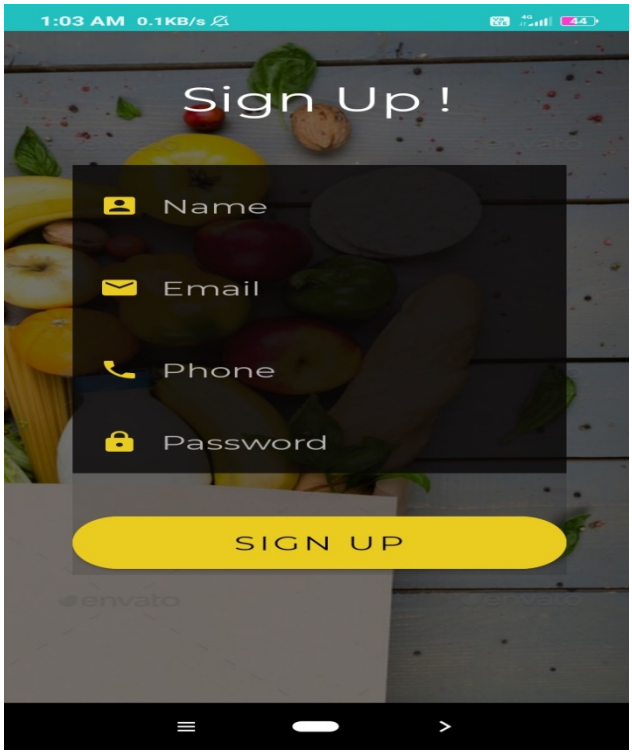

Figure A.8: customer signup page

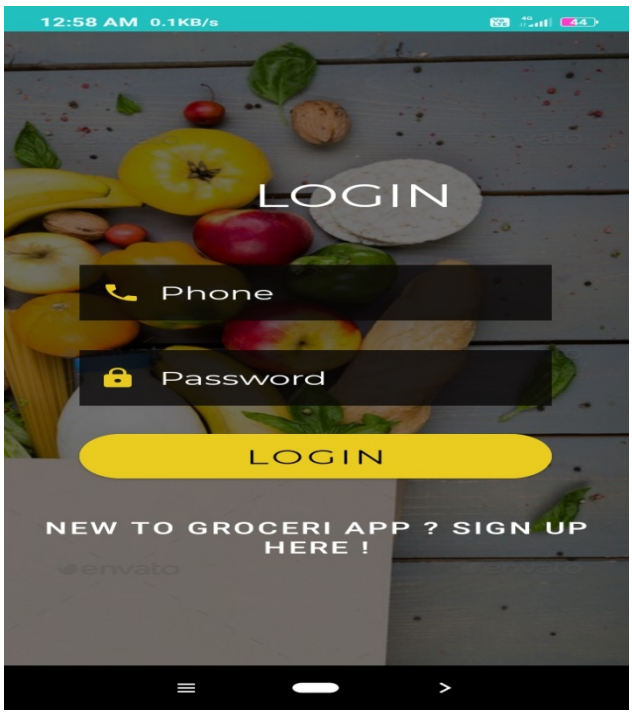

Figure A.9: customer login page

Figure A.10: view shop details

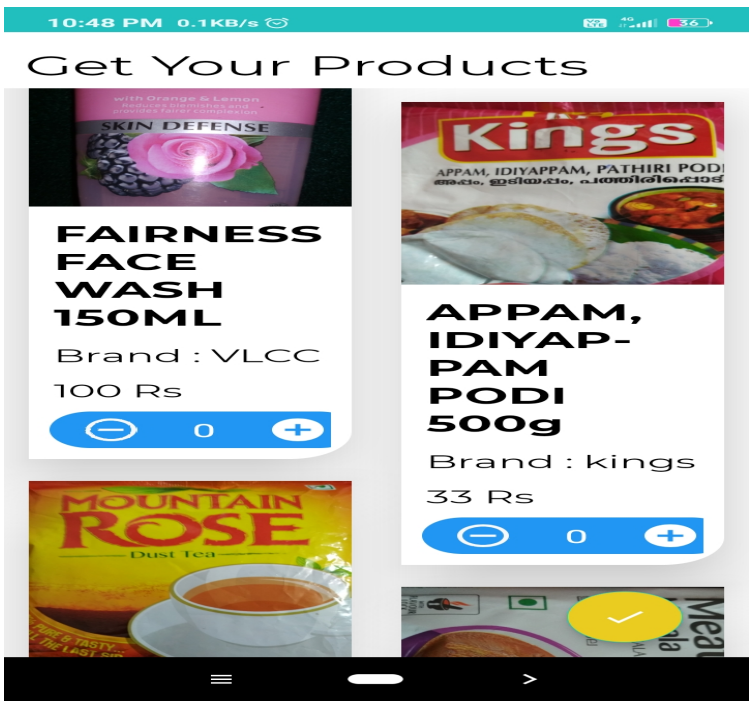

Figure A.11: view product page

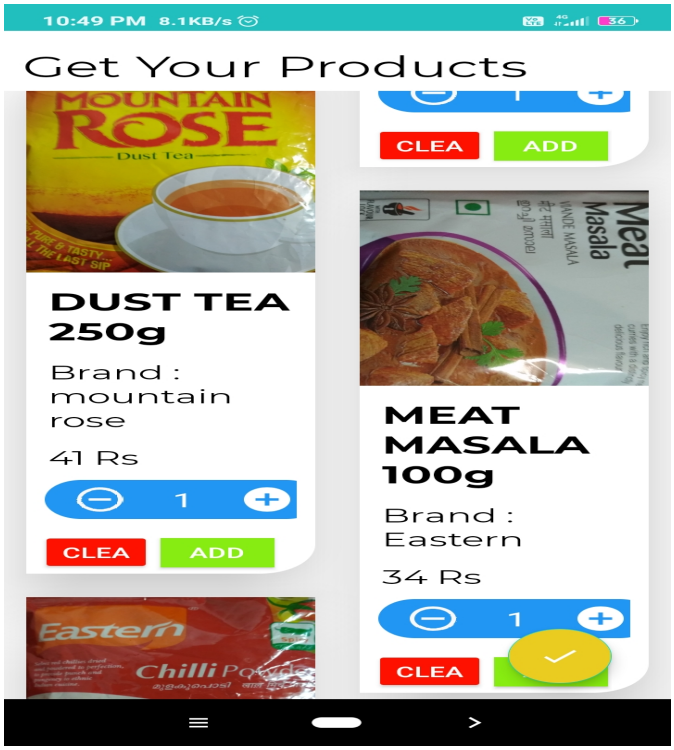

Figure A.12: product purchase page

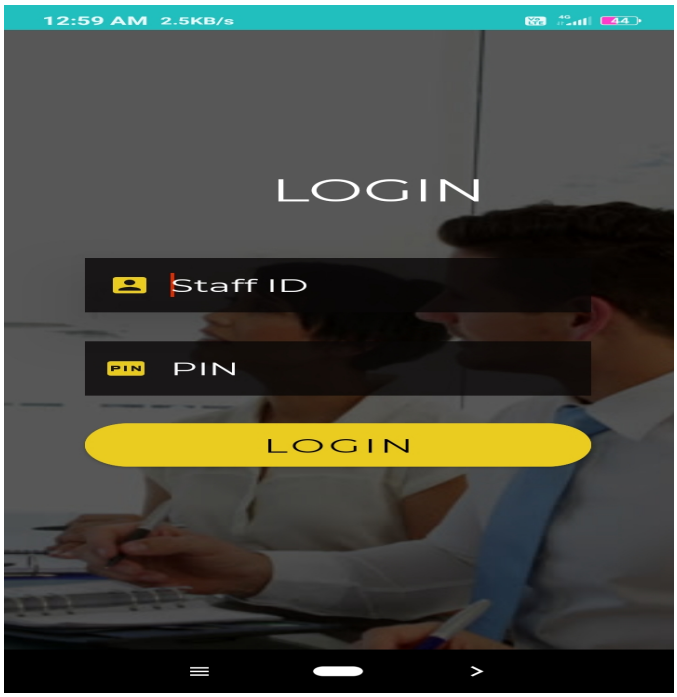

Figure A.13: staff login page

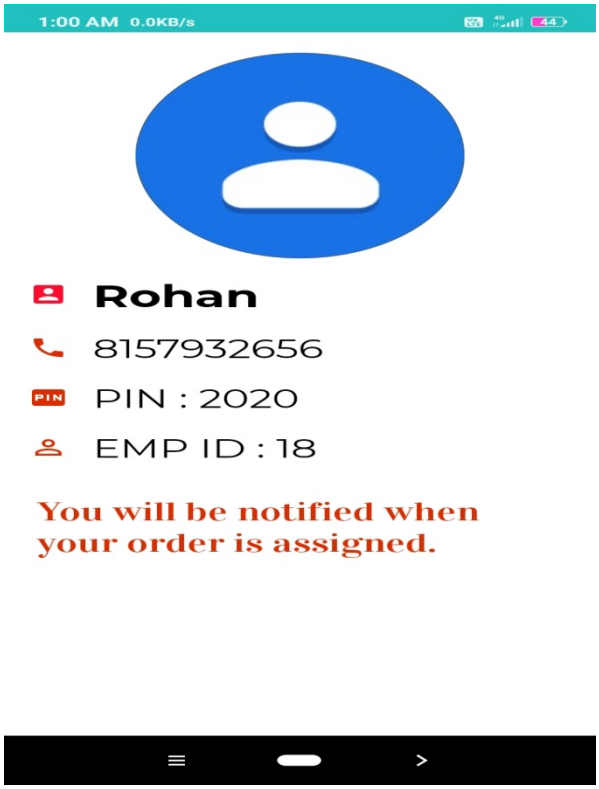

Figure A.14: staff details page

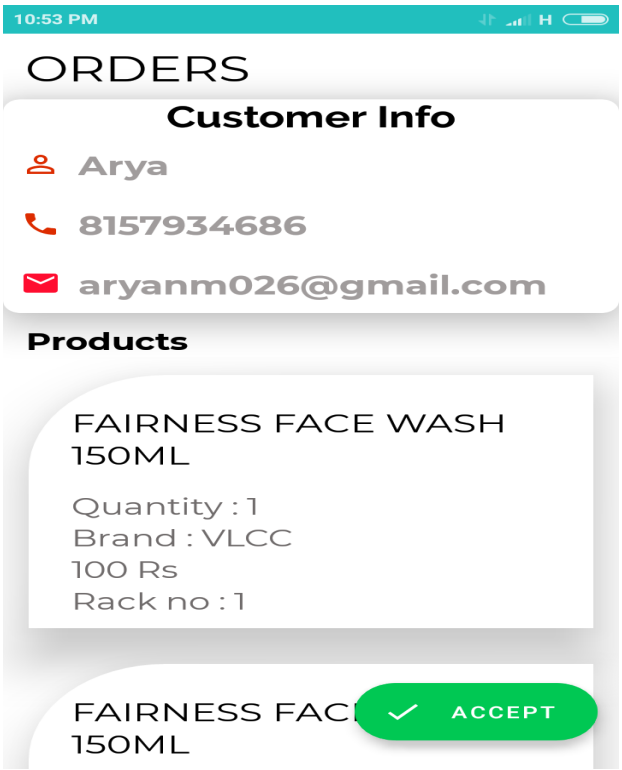

Figure A.15: staff accept order page

A.2 SAMPLE CODE

Home page

```
import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentManager;

import android.os.Bundle;

import com.hp.groceriapp.R;
import com.hp.groceriapp.Shopowner.Fragments.ProductsFragment;
import com.hp.groceriapp.Shopowner.Fragments.StaffsFragment;
import com.hp.groceriapp.Utils.IOnBackPressed;
import com.shrikanthravi.customnavigationdrawer2.data.MenuItem;
import com.shrikanthravi.customnavigationdrawer2.widget.SNavigationDrawer;

import java.util.ArrayList;
import java.util.List;
public class HomeDrawer extends AppCompatActivity {
    SNavigationDrawer sNavigationDrawer;
    Class fragmentClass;
    public static Fragment fragment;

    @Overrid
  protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home_drawer);
        sNavigationDrawer = findViewById(R.id.navigationDrawer);

  //Creating a list of menu Items
List<MenuItem> menuItems = new ArrayList<>();

        //Use the MenuItem given by this library and not the default one.
```

//First parameter is the title of the menu item and then the second parameter is the image which will be the background of the menu item.

```
menuItems.add(new MenuItem("Manage Products",R.drawable.news_bg));
menuItems.add(new MenuItem("Manage Staffs",R.drawable.feed_bg));
//  menuItems.add(new MenuItem("Messages",R.drawable.message_bg));
//  menuItems.add(new MenuItem("Music",R.drawable.music_bg));

//then add them to navigation drawer

sNavigationDrawer.setMenuItemList(menuItems);
fragmentClass =  ProductsFragment.class;
try {
   fragment = (Fragment) fragmentClass.newInstance();
} catch (Exception e) {
   e.printStackTrace();
}
if (fragment != null) {
   FragmentManager fragmentManager = getSupportFragmentManager();
fragmentManager.beginTransaction().setCustomAnimations(android.R.animator.fade_in,
android.R.animator.fade_out).replace(R.id.frameLayout, fragment).commit();
}
```

//Listener to handle the menu item click. It returns the position of the menu item clicked. Based on that you can switch between the fragments.

```
                                        sNavigationDrawer.setOnMenuItemClickListener(new
SNavigationDrawer.OnMenuItemClickListener() {
   @Override
   public void onMenuItemClicked(int position) {
      System.out.println("Position "+position);

      switch (position){
case 0:{
            fragmentClass = ProductsFragment.class;
```

```
            break;
        }
        case 1:{
            fragmentClass = StaffsFragment.class;
            break;
        }

    }

    //Listener for drawer events such as opening and closing.
     sNavigationDrawer.setDrawerListener(new SNavigationDrawer.DrawerListener()
{

        @Override
        public void onDrawerOpened() {

        }

        @Override
        public void onDrawerOpening(){

        }

        @Override
        public void onDrawerClosing(){
            System.out.println("Drawer closed");

            try {
                fragment = (Fragment) fragmentClass.newInstance();
            } catch (Exception e) {
                e.printStackTrace();
            }

            if (fragment != null) {
```

```
                    FragmentManager fragmentManager = getSupportFragmentManager();
fragmentManager.beginTransaction().setCustomAnimations(android.R.animator.fade_in,
android.R.animator.fade_out).replace(R.id.frameLayout, fragment).commit();

                }
            }
        @Override
            public void onDrawerClosed() {
            }
            @Override
            public void onDrawerStateChanged(int newState) {
               System.out.println("State "+newState);
            }
         });
      }
   });
  }
  @Override public void onBackPressed() {
Fragment fragment = getSupportFragmentManager().findFragmentById(R.id.frameLayout);
                    if(!(fragment instanceof IOnBackPressed) || !((IOnBackPressed)
fragment).onBackPressed()) {
      super.onBackPressed();
    }
  }
}
```

Shop owner signup

```
import androidx.appcompat.app.AppCompatActivity;
import androidx.viewpager.widget.ViewPager;

import android.os.Bundle;

import com.google.android.material.tabs.TabLayout;
import com.hp.groceriapp.R;
import com.hp.groceriapp.Shopowner.Fragments.SigninFragment;
import com.hp.groceriapp.Shopowner.Fragments.SignupFragment;
import com.hp.groceriapp.Shopowner.Adapters.TabAdapter;

public class ShopOwnerSignUp extends AppCompatActivity {
    private ViewPager mViewPager;


    private TabLayout mTabLayout;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.signupspare);
        //Tabs
        mViewPager = (ViewPager) findViewById(R.id.main_tabPager);
        mTabLayout = (TabLayout) findViewById(R.id.main_tabs);

//        mSectionsPagerAdapter = new
SectionsPagerAdapter(getSupportFragmentManager());
//
//        mViewPager.setAdapter(mSectionsPagerAdapter);
//
//        mTabLayout.setupWithViewPager(mViewPager);
```

```
    TabAdapter adapter = new TabAdapter(getSupportFragmentManager());
        adapter.addFragment(new SigninFragment(), "Sign in");
        adapter.addFragment(new SignupFragment(), "Sign Up");
        mViewPager.setAdapter(adapter);
        mTabLayout.setupWithViewPager(mViewPager);
    }
}
```

Free staff

```
import android.os.Bundle;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.GridLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.harishpadmanabh.apppreferences.AppPreferences;
import com.hp.groceriapp.R;
import com.hp.groceriapp.Retro.Retro;
import com.hp.groceriapp.Shopowner.Adapters.FreeStaffsAdapter;
import com.hp.groceriapp.Shopowner.Model.FreeStaffModel;

import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class FreeStaffs extends AppCompatActivity {

    private RecyclerView freeStaffsRV;
    String adminID;
```

```
    private AppPreferences appPreferences;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_free_staffs);
        initView();
        appPreferences = AppPreferences.getInstance(getApplicationContext(),
getResources().getString(R.string.app_name));
        adminID=appPreferences.getData("adminid");
        new Retro().getApi().freestaffs(adminID).enqueue(new
Callback<FreeStaffModel>() {
            @Override
            public void onResponse(Call<FreeStaffModel> call,
Response<FreeStaffModel> response) {
                FreeStaffModel freeStaffModel=response.body();
                if(freeStaffModel.getStatus().equalsIgnoreCase("success"))
                {

                    freeStaffsRV.setLayoutManager(new
GridLayoutManager(FreeStaffs.this,2));
                    freeStaffsRV.setAdapter(new
FreeStaffsAdapter(FreeStaffs.this,freeStaffModel,adminID));
                }else
                {
                    Toast.makeText(FreeStaffs.this, " No staffs are free .",
Toast.LENGTH_SHORT).show();
                }
            }
```

```
        @Override
        public void onFailure(Call<FreeStaffModel> call, Throwable t) {
            Toast.makeText(FreeStaffs.this, "Free staff api fail "+t,
Toast.LENGTH_SHORT).show();
        }
      });
   }

   private void initView() {
      freeStaffsRV = findViewById(R.id.freeStaffsRV);
   }
}
```

Customer home page

```
import android.os.Bundle;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.harishpadmanabh.apppreferences.AppPreferences;
import com.hp.groceriapp.Customer.Adapters.ShopList_Adapter;
import com.hp.groceriapp.Customer.CustomerModels.ShopListModel;
import com.hp.groceriapp.R;
import com.hp.groceriapp.Retro.Retro;

import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
```

```
public class CustomerHome extends AppCompatActivity {

    AppPreferences appPreferences;
    private RecyclerView shoplistRV;
    ShopListModel shopListModel;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_customer_home);
        appPreferences = AppPreferences.getInstance(this,
getResources().getString(R.string.app_name));

        initView();
        new Retro().getApi().SHOP_LIST_MODEL_CALL().enqueue(new
Callback<ShopListModel>() {
            @Override
            public void onResponse(Call<ShopListModel> call, Response<ShopListModel>
response) {

                shopListModel = response.body();
                if (shopListModel.getStatus().equalsIgnoreCase("success")) {

                    shoplistRV.setLayoutManager(new
LinearLayoutManager(getApplicationContext(), RecyclerView.VERTICAL, false));
                    shoplistRV.setAdapter(new ShopList_Adapter(shopListModel,
CustomerHome.this));

                } else {
```

```
                Toast.makeText(CustomerHome.this, "No Shops found",
Toast.LENGTH_SHORT).show();
            }

        }

        @Override
        public void onFailure(Call<ShopListModel> call, Throwable t) {
            Toast.makeText(CustomerHome.this, "API failure"+t,
Toast.LENGTH_SHORT).show();
        }
      });


  }

  private void initView() {
    shoplistRV = findViewById(R.id.shoplistRV);
  }
}
```

Customer login

```
import android.content.Intent;
import android.os.Bundle;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;
```

```
import com.google.android.material.button.MaterialButton;
import com.google.android.material.snackbar.BaseTransientBottomBar;
import com.google.android.material.snackbar.Snackbar;
import com.harishpadmanabh.apppreferences.AppPreferences;
import com.hp.groceriapp.Customer.CustomerModels.Cust_LoginModel;
import com.hp.groceriapp.R;
import com.hp.groceriapp.Retro.Retro;

import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class CustomerLogin extends AppCompatActivity {



    private EditText customerPhn;
    private EditText customerPass;
    private MaterialButton proceed;
    private MaterialButton customerSignupBTN;
    private AppPreferences appPreferences;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.customer_login);
        initView();
        appPreferences = AppPreferences.getInstance(this,
getResources().getString(R.string.app_name));

        //.....dummy cred
```

```
//  customerPhn.setText("7012069385");
  //customerPass.setText("qwerty");

  //.....dummy cred ends


  customerSignupBTN.setOnClickListener(view -> {
     startActivity(new Intent(CustomerLogin.this, CustomerSignup.class));
  });

  proceed.setOnClickListener(view -> {
     if (customerPhn.getText().toString().isEmpty() ||
           customerPass.getText().toString().isEmpty()) {
        Snackbar.make(proceed, "Fill all Fields!",
BaseTransientBottomBar.LENGTH_LONG).show();
     } else {
        new
Retro().getApi().CUST_LOGIN_MODEL_CALL(customerPhn.getText().toString(),
           customerPass.getText().toString()).enqueue(new
Callback<Cust_LoginModel>() {
           @Override
           public void onResponse(Call<Cust_LoginModel> call,
Response<Cust_LoginModel> response) {
              Cust_LoginModel cust_loginModel = response.body();
              if (cust_loginModel.getStatus().equalsIgnoreCase("success")) {

appPreferences.saveData("customer_id",cust_loginModel.getUser_data().getCustomer_
id());
                 startActivity(new Intent(CustomerLogin.this,CustomerHome.class));
```

```
                } else {
                    Snackbar.make(proceed, "Wrong Creentials",
BaseTransientBottomBar.LENGTH_LONG).show();

                }
            }

            @Override
            public void onFailure(Call<Cust_LoginModel> call, Throwable t) {
                Snackbar.make(proceed, "Cust_LoginModel API FAILURE :" + t,
BaseTransientBottomBar.LENGTH_LONG).show();


            }
        });

    }
  });
}

private void initView() {

    customerPhn = findViewById(R.id.customer_phn);
    customerPass = findViewById(R.id.customer_pass);
    proceed = findViewById(R.id.proceed);
    customerSignupBTN = findViewById(R.id.customer_signupBTN);
}}
```