

# DogStore

## REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT  
FOR

Six Month Industrial Training

at

**Auribises Technologies, Ludhiana**  
**(from June 5,2017 to December 5,2017)**

Submitted By

Gursimran Singh Basra (1411258)



---

Information Technology Department  
**GURU NANAK DEV ENGINEERING COLLEGE**  
**LUDHIANA, INDIA**

---

## Acknowledgement

We are highly grateful to the Dr. M.S. Saini, Director, Guru Nanak Dev Engineering College (GNDEC), Ludhiana, for providing this opportunity to carry out the major project work at Guru Nanak Dev Engineering College.

The constant guidance and encouragement received from Dr. K.S. Mann H.O.D. IT Department, GNDEC Ludhiana has been of great help in carrying out the project work and is acknowledged with reverential thanks.

We would like to express a deep sense of gratitude and thanks profusely to Prof. Sandeep Kumar Singla, without his wise counsel and able guidance, it would have been impossible to complete the project in this manner.

We express gratitude to other faculty members of Information Technology department of GNDEC for their intellectual support throughout the course of this work.

Finally, we are indebted to all whosoever have contributed in this report work.

**Gursimran Singh Basra**

---

## Abstract

Modern life offers a plethora of options of services and goods for consumers. As a result, peoples expenses have gone up dramatically, e.g., compared to a decade ago, and the cost of living has been increasing day by day. Thus it becomes essential to keep a check on expenses in order to live a good life with a proper budget set up.

The Android OS smartphones is one of the top-selling in the world ,it is apparent that people have been using smartphones as an organizational tool. .

Expense Tracker Mobile Application (Monitary) was developed for Android users to keep track of their expenses and determine whether they are spending as per their set budget. Potential users need to input the required data such as the expense amount, merchant, category, and date when the expense was made. Optional data such as sub-category and extra notes about the expense can be entered as well.The application allows users to track their expenses daily, weekly, monthly, and yearly in terms of summary, bar graphs, and pie-charts.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Introduction to Organization . . . . .	1
1.2	Introduction To Project . . . . .	1
1.3	Project Category . . . . .	1
1.4	Objectives . . . . .	1
1.5	Problem Formulation . . . . .	2
1.6	Identification/Reorganization of Need . . . . .	2
1.7	Existing System . . . . .	2
1.8	Proposed System . . . . .	3
1.9	Unique Features of the System . . . . .	3
1.9.1	Complete Application Built Instantly . . . . .	3
1.9.2	Collective Information . . . . .	3
1.9.3	User Friendly . . . . .	3
<b>2</b>	<b>Requirement Analysis and System Specification</b>	<b>4</b>
2.1	Feasibility Study (Technical,Economical,Operational) . . . . .	4
2.1.1	Types of Feasibility . . . . .	5
2.1.1.1	Technical Feasibility . . . . .	5
2.1.1.2	Economic Feasibility . . . . .	6
2.1.1.3	Operational Feasibility . . . . .	6
2.2	Software Requirement Specification Document . . . . .	7
2.2.1	Data Requirement . . . . .	7
2.2.2	Performance Requirement . . . . .	7
2.2.3	Functional Requirement . . . . .	7
2.3	Intended User . . . . .	7
2.4	Features . . . . .	7
2.5	Expected hurdles . . . . .	8
2.6	SDLC model to be used . . . . .	8
2.6.1	Agile Model . . . . .	8
<b>3</b>	<b>System Design</b>	<b>9</b>
3.1	Function Oriented . . . . .	9
3.2	Detail Design . . . . .	9
3.3	Data Flow Diagram . . . . .	10
3.4	Database Design . . . . .	10
3.4.1	ER Diagrams . . . . .	10
3.4.2	Normalization . . . . .	10
3.4.3	First Normal Form(1NF) . . . . .	11
3.4.4	Second Normal Form(2NF) . . . . .	11
3.4.5	Third Normal Form(3NF) . . . . .	11
3.4.6	Boyce and Codd Normal Form (BCNF) . . . . .	11
3.4.7	Database Manipulation . . . . .	12
3.4.8	Database Connection Controls and Strings . . . . .	13

3.4.9	Methodology . . . . .	13
<b>4</b>	<b>Implementation ,Testing and Maintenance</b>	<b>16</b>
4.1	Introduction to Languages,IDEs,Tools and Technologies used for Implementation	16
4.1.1	Technologies used . . . . .	16
4.1.1.1	Firebase . . . . .	16
4.1.1.2	Android . . . . .	18
4.1.2	Picasso . . . . .	20
4.1.3	API.AI . . . . .	20
4.1.4	Material Design . . . . .	21
4.1.4.1	Usage . . . . .	21
4.1.4.2	Behaviour . . . . .	21
4.1.4.3	Snackbar specs . . . . .	21
4.1.5	Language used . . . . .	21
4.1.5.1	JAVA . . . . .	21
4.1.5.2	XML . . . . .	24
4.1.6	IDE used . . . . .	24
4.1.6.1	Android Studio . . . . .	24
4.1.7	Introduction to $\text{\LaTeX}$ . . . . .	24
4.1.7.1	Typesetting . . . . .	25
4.1.7.2	Installing $\text{\LaTeX}$ on System . . . . .	26
4.1.7.3	Pdftscreen $\text{\LaTeX}$ . . . . .	27
4.2	Coding standards of Language used . . . . .	27
4.2.1	Introduction . . . . .	27
4.2.2	Source file basics . . . . .	27
4.3	GANTT chart . . . . .	28
4.4	Testing Techniques and Test Plans . . . . .	28
4.4.1	xUnit Framework . . . . .	28
4.4.2	JUnit . . . . .	28
4.4.3	Robotium Android Testing Tool . . . . .	29
4.4.4	MonkeyRunner Android App Testing . . . . .	29
<b>5</b>	<b>Results and Discussions</b>	<b>30</b>
5.1	User Interface Representation (Of Respective Project) . . . . .	30
5.1.1	Snapshots of system . . . . .	30
5.2	Back Ends Representation (Database to be used) . . . . .	31
5.2.1	Firebase Realtime Database . . . . .	31
5.2.2	How does it work? . . . . .	31
5.2.3	Snapshots of Database . . . . .	33
<b>6</b>	<b>Conclusion and Future Scope</b>	<b>45</b>
6.1	Conclusion . . . . .	45
6.2	Future Scope . . . . .	45

# List of Figures

2.1	Agile Model . . . . .	8
3.1	Data Flow Diagram . . . . .	10
3.2	digital ocean server to fetch images . . . . .	12
3.3	digital ocean server Console . . . . .	13
3.4	Firebase Authentication . . . . .	14
3.5	Firebase Database . . . . .	14
4.1	Firebase Console . . . . .	16
4.2	Firebase Products . . . . .	17
4.3	Android Anatomy . . . . .	18
4.4	Activity Life Cycle . . . . .	19
4.5	Android Studio . . . . .	24
4.6	L <sup>A</sup> T <sub>E</sub> X Logo . . . . .	25
4.7	Donald Knuth, Inventor Of T <sub>E</sub> X typesetting system . . . . .	25
4.8	Gantt Chart . . . . .	28
5.1	Login Screen . . . . .	31
5.2	Register Screen . . . . .	32
5.3	Items Cart . . . . .	33
5.4	Navigation Drawer . . . . .	34
5.5	Shopping Cart . . . . .	35
5.6	Payment Gateway (PayuMoney) . . . . .	36
5.7	Payment Gateway Login . . . . .	37
5.8	Register to gateway . . . . .	38
5.9	Payment . . . . .	39
5.10	About us . . . . .	40
5.11	Enquiry Bot . . . . .	41
5.12	Share Application . . . . .	42
5.13	Feedback . . . . .	43
5.14	Confirmation Email in Fire base . . . . .	44

# Chapter 1

## Introduction

### 1.1 Introduction to Organization

Forget about not remembering where you spent your money! Keep track of your expenses and get a lot of statistics to know in what you spend more and you can evaluate where you can cut some expenses! Take advantage of this tool that will help you to manage your money and be informed all the time. You can also configure to get notifications when you have to make some payments every month so you don't forget! This Application is made for any user who wants or wish to keep a better track of their expenses in order to keep them in a same place.

### 1.2 Introduction To Project

With the launch and increase in sales of smartphones over the last few years, people are using mobile applications to get their work done, which makes their lives easier. Mobile applications comprise various different categories such as Entertainment, Sports, Lifestyle, Education, Games, Food and Drink, Health and Fitness, Finance, etc. This Expense Tracker application falls in the Finance Category and serves the important purpose of managing finances which is a very important part of one's life. The software product went through the design, development, and the testing phase as a part of the Software Development Lifecycle.

The applications interface is designed using custom art elements, the functionality is implemented using Android SDK, and the phase of testing the product was accomplished successfully. The application is not much user intensive but just comprises of having them enter the expense amount, date, category, merchant and other optional attributes (taking picture of the receipts, entering notes about the expense, adding subcategories to the categories). With this entered information, the user is able to see the expense details about the product, price, quality, quantity or category. All these topics have been explained in detail in their respective chapters

### 1.3 Project Category

Our Project is internet based. Dog lovers can get products, vaccines, chains calcium tablets, bones of their choice. They can get stuff delivered to their homes. This application helps them to find needy products for their pet dogs. They can search products according to need of their pet dogs.

### 1.4 Objectives

- The objective of our project is to provide information to dog lovers about ideal and best products for their pet dogs.

- We want to facilitate by providing them description about different products and their quality.
- One of the biggest objective is to aware the doglovers about different products of different quality so they can choose the right product.
- Implement the various concepts of Android programming together within an application that becomes helpful for intended users.
- Developing an Android based application for the dog lovers and system having user friendly interface.
- Read on for help with choosing a dog's food, training, first aid and vet visits, walks, dog sitters, GPS trackers.
- To make life better. Even better is when they combine to make everything easier, because dog lovers don't need to go to market for dog's food,chains,calcium bones,vaxines.they get these all stuff online.

## 1.5 Problem Formulation

A Problem well defined is a problem half solved. Formulating a Problem Formulation is the first and most important step of a research process. It fits the level of researchers level of research skills, needed resources and time restrictions and can be investigated through the collection and analysis of data Our team will give proper attention to the visitor problems. We are also providing contact system in our project ,if any visitor find something wrong in our content then they can contact us . We will welcome any type of suggestion and if we find it right then we will implement that suggestion in our application.

## 1.6 Identification/Reorganization of Need

Our Dogstore basically targets different segments to fulfill the desired needs of people which are following below:-

- The needs of our project is to ignite the interest in care and love about their pet dogs.
- We want to facilitate people by providing them different forms of products and quality.
- For the enquiries customers also interact to the artificial intelligence system.

## 1.7 Existing System

In the existing system , we have some applications on the internet in which there is least information about products and feed of pet dogs but we are providing that facility too. Some applications does not have pay online facility. We did not find any application on the internet which provide detailed information about different products collectively but we have tried to give facilities like AI system, payment by payU. So, all in all our site is user friendly site.



## **1.8 Proposed System**

To remove the limitations of the existing system , In the proposed system , people can easily get information about all the products of pet dogs. We are providing online ordering in our project so that people can have reach to our facilities easily.

## **1.9 Unique Features of the System**

### **1.9.1 Complete Application Built Instantly**

As soon as you join our application is created for you. Every page has text and photos to help you get started.

### **1.9.2 Collective Information**

We did not find any application on the internet which provide detailed information about different products collectively but we have tried to give facilities like AI system,payment by payU.

### **1.9.3 User Friendly**

Our application is very user friendly .Design is simple and effective. People can reach to different section easily.

# Chapter 2

## Requirement Analysis and System Specification

### 2.1 Feasibility Study (Technical,Economical,Operational)

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spend on it. Feasibility study lets the developer foresee the future of the project and the usefulness. A feasibility study of a system proposal is according to its workability, which is the impact on the organization, ability to meet their user needs and effective use of resources. Carrying out a feasibility study involves information assessment, information collection and report writing. The information assessment phase identifies the information that is required to answer the three questions set out above. Once the information has been identified, you should question information sources to discover the answers to these questions Thus when a new application is proposed it normally goes through a feasibility study before it is approved for development.

A feasibility study is designed to provide an overview of the primary issues related to a business idea. The purpose is to identify any make or break issues that would prevent your business from being successful in the marketplace. In other words, a feasibility study determines whether the business idea makes sense. A thorough feasibility analysis provides a lot of information necessary for the business plan. For example, a good market analysis is necessary in order to determine the projects feasibility. This information provides the basis for the market section of the business plan.

The document provide the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as Technical, Economic and Operational feasibilities. Feasibility is defined as the practical extent to which a project can be performed successfully. To evaluate feasibility, a feasibility study is performed, which determines whether the solution considered to accomplish the requirements is practical and workable in the software. Information such as resource availability, cost estimation for software development, benefits of the software to the organization after it is developed and cost to be incurred on its maintenance are considered during the feasibility study. The objective of the feasibility study is to establish the reasons for developing the software that is acceptable to users, adaptable to change and conformable to established standards.

Objectives of feasibility study are listed below.

- To analyze whether the software will meet organizational requirements
- To determine whether the software can be implemented using the current technology and within the specified budget and schedule
- To determine whether the software can be integrated with other existing software.

### 2.1.1 Types of Feasibility

Various types of feasibility that are commonly considered include technical feasibility, operational feasibility, and economic feasibility.

#### 2.1.1.1 Technical Feasibility

Technical feasibility is one of the first studies that must be conducted after the project has been identified. In large engineering projects consulting agencies that have large staffs of engineers and technicians conduct technical studies dealing with the projects.

When writing a feasibility report, the following should be taken to consideration:

- A brief description of the business to assess more possible factors which could affect the study
- The part of the business being examined
- The human and economic factor
- The possible solutions to the problem

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs and procedures. Having identified an outline system, the investigation must go on to suggest the type of equipment, required method developing the system, of running the system once it has been designed. Technical feasibility assesses the current resources (such as hardware and software) and technology, which are required to accomplish user requirements in the software within the allocated time and budget. For this, the software development team ascertains whether the current resources and technology can be upgraded or added in the software to accomplish specified user requirements. Technical feasibility also performs the following tasks.

- Analyzes the technical skills and capabilities of the software development team members
- Determines whether the relevant technology is stable and established
- Ascertains that the technology chosen for software development has a large number of users so that they can be consulted when problems arise or improvements are required.

Technical issues raised during the investigation are:

- Does the existing technology sufficient for the suggested one?
- Can the system expand if developed?

The project should be developed such that the necessary functions and performance are achieved within the constraints. The project is developed within latest technology. Through the technology may become obsolete after some period of time, due to the fact that never version of same software supports older versions, the system may still be used. So there are minimal constraints involved with this project. The system has been developed using Java the project is technically feasible for development.

#### 2.1.1.2 Economic Feasibility

The purpose of the economic feasibility assessment is to determine the positive economic benefits to the organization that the proposed system will provide. It includes quantification and identification of all the benefits expected. This assessment typically involves a cost/benefits analysis.

Software is said to be economically feasible if it focuses on the issues listed below.

- Cost incurred on software development to produce long-term gains for an organization.
- Cost required to conduct full software investigation (such as requirements elicitation and requirements analysis).
- Cost of hardware, software, development team, and training.

The following are some of the important financial questions asked during preliminary investigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

#### 2.1.1.3 Operational Feasibility

If the system is not easy to operate, than operational process would be difficult. The operator of the system should be given proper training. The system should be made such that the user can interface the system without any problem.

Operational feasibility is a measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development. The operational feasibility assessment focuses on the degree to which the proposed development projects fits in with the existing business environment and objectives with regard to development schedule, delivery date, corporate culture, and existing business processes.

Operational feasibility also performs the following tasks.

- Determines whether the problems anticipated in user requirements are of high priority.
- Determines whether the solution suggested by the software development team is acceptable.
- Analyzes whether users will adapt to a new software.
- Determines whether the organization is satisfied by the alternative solutions proposed by the software development team.

This includes the following questions:

- Is there sufficient support for the users?
- Will the proposed system cause harm?
- The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible.

This system proposed will have a very user friendly interface, a nave user will be able to understand the user interface in seconds. Itll have basic yet intriguing interface with a list various pdf categories, fab button which indicates that pdf upload using it and many other functions which is easily understandable.

## **2.2 Software Requirement Specification Document**

### **2.2.1 Data Requirement**

For this project, no specific data collections were required, all that was needed for testing the project. Initial Products and Prices or was uploaded to check sytem works in right way or not.

### **2.2.2 Performance Requirement**

Basis on resource requirements, this application will be able to run on pretty low-end devices with 512MB of ram and 800Mhz processor.

### **2.2.3 Functional Requirement**

The application will have different kind of functions like buy things,pay by cards or cash on delivery etc of various.User can buy products easily and quickly.

## **2.3 Intended User**

This Application is made for any user who wants to buy products for their pets. Easy platform for the user who loves and care their pets.

## **2.4 Features**

- Buy products easily.
- For the enquiries user can talk to a AI system.
- Provide email when users orders something.
- Feedback option given so that user can share his views.

## 2.5 Expected hurdles

- An algorithm for working of feedback notification to admins is one of the biggest hurdle in the project.
- Libraries for Android studio needs to be added into the imported from various sources such as GitHub, Firebase.
- Dependencies of the different libraries used in the project.

## 2.6 SDLC model to be used

### 2.6.1 Agile Model

Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements. In Agile, the tasks are divided to time boxes (small time frames) to deliver specific features for a release. An Agile software life cycle is much different as compared to traditional software development frameworks like Waterfall. In Agile, more emphasis is given to sustained and quick development of product features rather than spending more time during the initial project planning, and analyzing the actual requirements. The Agile team develops the product through a series of iterative cycles known as sprints. Besides development activity, other aspects pertaining to development such as product analysis, designing the product features, developing the functionality, and testing the development for bugs are also carried out during the sprints. The incremental cycles should always produce a shippable product release that can be readily deployed.

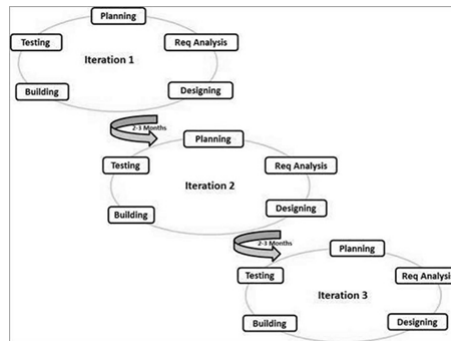


Figure 2.1: Agile Model

# Chapter 3

## System Design

### 3.1 Function Oriented

Object-oriented design is the process of planning a system of interacting objects for the purpose of solving a software problem. It is one approach to software design. An object contains encapsulated data and procedures grouped together to represent an entity. The 'object interface' defines how the object can be interacted with. An object-oriented program is described by the interaction of these objects. Object-oriented design is the discipline of defining the objects and their interactions to solve a problem that was identified and documented during object-oriented analysis.

### 3.2 Detail Design

The design phase involves converting the informational, functional, and network requirements identified during the initiation and planning phases into unified design specifications that developers use to script programs during the development phase. Program designs are constructed in various ways. Using a top-down approach, designers first identify and link major program components and interfaces, then expand design layouts as they identify and link smaller subsystems and connections. Using a bottom-up approach, designers first identify and link minor program components and interfaces, then expand design layouts as they identify and link larger systems and connections.

Contemporary design techniques often use prototyping tools that build mock-up designs of items such as application screens, database layouts, and system architectures. End users, designers, developers, database managers, and network administrators should review and refine the prototyped designs in an iterative process until they agree on an acceptable design. Designers should carefully document completed designs. Detailed documentation enhances a programmers ability to develop programs and modify them after they are placed in production. The documentation also helps management ensure final programs are consistent with original goals and specifications. Organizations should create initial testing, conversion, implementation, and training plans during the design phase. Additionally, they should draft user, operator, and maintenance manuals.

For design of the website project:

- First Database has to be designed which can be used to handle all the requirements of the users.
- The basic structure of the website has to be designed.
- The main template to be used for the website is designed.

### 3.3 Data Flow Diagram

Information moves through software, it is modified by a series of transformations. Data flow diagram is a graphical representation that depicts information flow and the transforms that are applied as data move from input to output. The basic form of a data flow diagram, also known as a data flow graph or a bubble chart. The data flow diagram may be used to represent a system or software at any level of abstraction. DFDs may be partitioned into levels that represent increasing information flow and functional detail. The DFD provides a mechanism for functional modeling as well as information flow modeling.

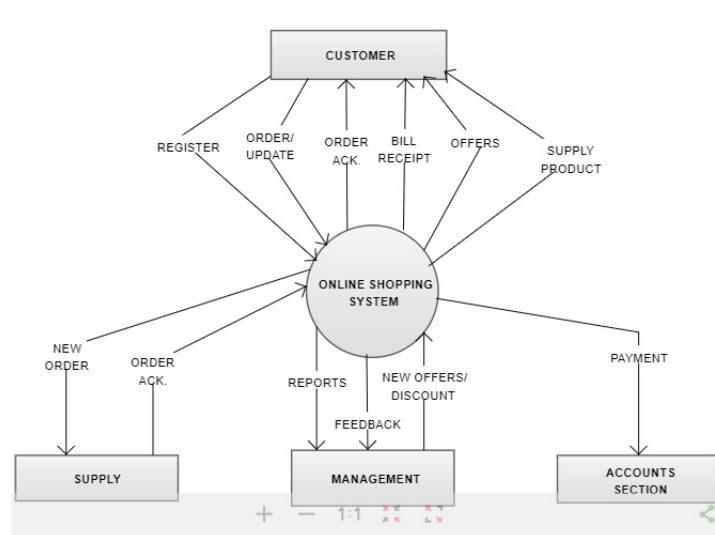


Figure 3.1: Data Flow Diagram

### 3.4 Database Design

#### 3.4.1 ER Diagrams

An entity-relationship (ER) diagram, a graphical representation of entities and their relationships to each other, typically used in computing in regard to the organization of data within databases or information systems. An entity is a piece of data—an object or concept about which data is stored. A relationship is how the data is shared between entities. Entities may be characterized not only by relationships, but also by additional properties (attributes), which include identifiers called "primary keys". Diagrams created to represent attributes as well as entities and relationships may be called entity-attribute-relationship diagrams, rather than entity-relationship models. There is a tradition for ER/data models to be built at two or three levels of abstraction. Note that the conceptual-logical-physical hierarchy below is used in other kinds of specification, and is different from the three schema approach to software engineering.

#### 3.4.2 Normalization

Database Normalisation is a technique of organizing the data in the database. Normalization is a systematic approach of decomposing tables to eliminate data redundancy and undesirable



characteristics like Insertion, Update and Deletion Anamolies. It is a multi-step process that puts data into tabular form by removing duplicated data from the relation tables. Normalization is used for mainly two purpose:

- Eliminating reduntant(useless) data.
- Ensuring data dependencies make sense i.e data is logically stored.

Normalization rule are divided into following normal form.

### **3.4.3 First Normal Form(1NF)**

As per First Normal Form, no two Rows of data must contain repeating group of information i.e each set of column must have a unique value, such that multiple columns cannot be used to fetch the same row. Each table should be organized into rows, and each row should have a primary key that distinguishes it as unique.

The Primary key is usually a single column, but sometimes more than one column can be combined to create a single primary key.

### **3.4.4 Second Normal Form(2NF)**

As per the Second Normal Form there must not be any partial dependency of any column on primary key. It means that for a table that has concatenated primary key, each column in the table that is not part of the primary key must depend upon the entire concatenated key for its existence. If any column depends only on one part of the concatenated key, then the table fails Second normal form.

### **3.4.5 Third Normal Form(3NF)**

Third Normal form applies that every non-prime attribute of table must be dependent on primary key, or we can say that, there should not be the case that a non-prime attribute is determined by another non-prime attribute. So this transitive functional dependency should be removed from the table and also the table must be in Second Normal form.

### **3.4.6 Boyce and Codd Normal Form (BCNF)**

Boyce and Codd Normal Form is a higher version of the Third Normal form. This form deals with certain type of anomaly that is not handled by 3NF. A 3NF table which does not have multiple overlapping candidate keys is said to be in BCNF. For a table to be in BCNF, following conditions must be satisfied:

- R must be in 3rd Normal Form
- and, for each functional dependency ( X Y ), X should be a super Key.

### 3.4.7 Database Manipulation

Data manipulation is the process of changing data in an effort to make it easier to read or be more organized. For example, a log of data could be organized in alphabetical order, making individual entries easier to locate. Data manipulation is often used on web server logs to allow a website owner to view their most popular pages as well as their traffic sources.

Users in the Accounting field or other fields that work with numbers often manipulate data to figure out costs of products, trends in sales, potential tax obligations, or how well merchandise is selling per week or month. Stock market analysts are frequently using data manipulation to predict trends in the stock market and how stocks might perform in the near future.

Computers may also use data manipulation to display information to users in a more meaningful way, based on code in a software program, web page, or data formatting defined by a user. The manipulation portion of SQL consists of its 'Insert', 'Update', 'Merge', 'Truncate', and 'Delete' operators. Only SQL's 'Select' operator is used for querying. (Other portions of SQL are for concurrency control, security, data definition, etc.)

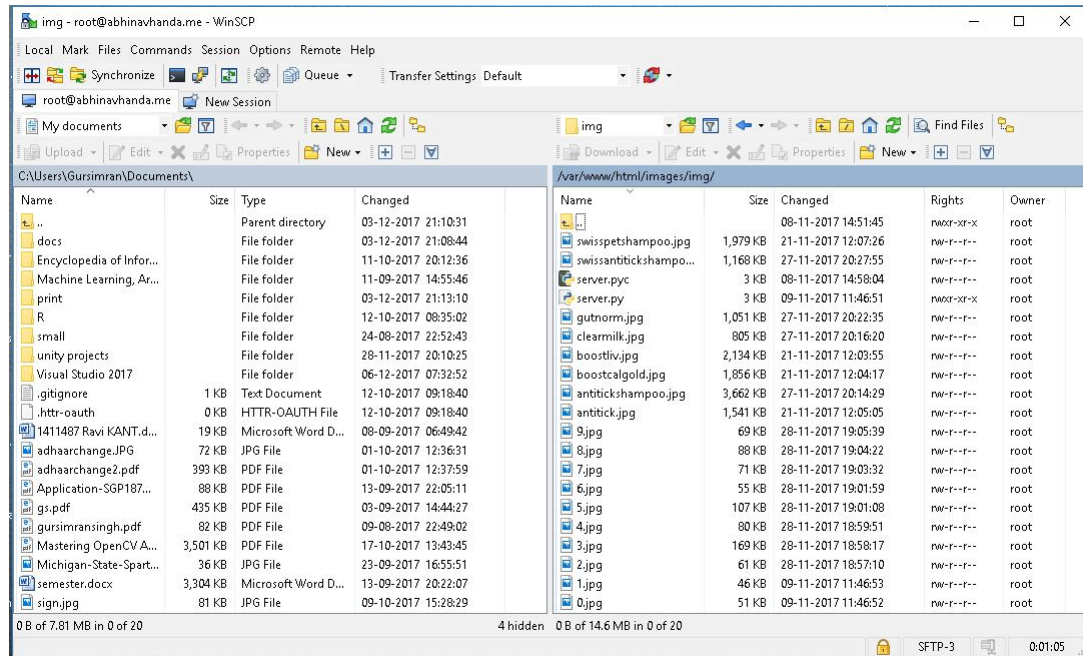


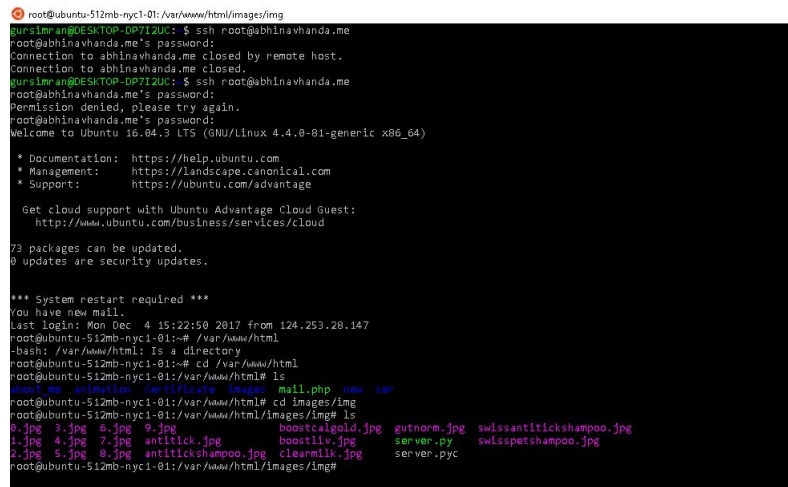
Figure 3.2: digital ocean server to fetch images

DigitalOcean, Inc. is an American cloud infrastructure provider[2] headquartered in New York City with data centers worldwide. DigitalOcean provides developers cloud services that help to deploy and scale applications that run simultaneously on multiple computers. As of December 2015, DigitalOcean was the second largest hosting company in the world in terms of web-facing computers. In 2003, Ben and Moisey Uretsky who had founded ServerStack, a managed hosting business, wanted to create a new product which would combine the web hosting and virtual servers. The Uretskys, having surveyed the cloud hosting market felt that most hosting companies were targeting enterprise client leaving the entrepreneurial software developers market underserved. In 2011 the Uretskys founded DigitalOcean, a company which would provide server provisioning and cloud hosting for software developers.

In 2012 the Uretskys met co-founder Mitch Wainer following Wainer's response to a

Craigslist job listing. The company launched their beta product in January 2012. By mid-2012, the founding team consisted of Ben Uretsky, Moisey Uretsky, Mitch Wainer, Jeff Carr, and Alec Hartman. After DigitalOcean was accepted into TechStars 2012's startup accelerator in Boulder, Colorado, the founders moved to Boulder to work on the product.[11] By the end of the accelerator program[when?], the company had signed up 400 customers and launched around 10,000 cloud server instances.

On January 15, 2013, DigitalOcean became one of the first cloud-hosting companies to offer SSD-based virtual machines. Following a TechCrunch review which was syndicated by Hacker News, DigitalOcean saw a rapid increase in customers. In December 2013, DigitalOcean opened its first European data center located in Amsterdam. By the end of December 2013, Netcraft reported that DigitalOcean was the fastest growing cloud hosting service in the world in terms of web-facing computer count. During 2014, the company continued its expansion, opening new data centers in Singapore and London. By May 2015, DigitalOcean became the second largest hosting provider in the world according to a report by Netcraft. During 2015 DigitalOcean expanded further with a data center in Toronto, Canada. Later in 2016 they continued expansion to Bangalore, India. As of July 2017, the company has 12 data centers in various parts of the globe.



```

root@ubuntu-512mb-nyc1-01: /var/www/html/images/img#
gursimran@DESKTOP-DP712UC: $ ssh root@abhinavhanda.me
root@abhinavhanda.me's password:
Connection to abhinavhanda.me closed by remote host.
gursimran@DESKTOP-DP712UC: $ ssh root@abhinavhanda.me
root@abhinavhanda.me's password:
Permission denied, please try again.
root@abhinavhanda.me's password:
welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.4.0-81-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Get cloud support with Ubuntu Advantage Cloud Guest:
http://www.ubuntu.com/business/services/cloud

73 packages can be updated.
0 updates are security updates.

*** System restart required ***
You have new mail.
Last login: Mon Dec 4 15:22:58 2017 from 124.253.28.147
root@ubuntu-512mb-nyc1-01:~# /var/www/html/
-bash: /var/www/html: is a directory
root@ubuntu-512mb-nyc1-01:~# cd /var/www/html
root@ubuntu-512mb-nyc1-01:/var/www/html# ls
css  fonts  images  js  mail.php  new  var
root@ubuntu-512mb-nyc1-01:/var/www/html# cd images/img
root@ubuntu-512mb-nyc1-01:/var/www/html/images/img# ls
0.jpg  3.jpg  6.jpg  9.jpg  boostcalgold.jpg  gutnorm.jpg  swissantitickshampoo.jpg
1.jpg  4.jpg  7.jpg  antitick.jpg  boostilv.jpg  server.py  swisspetshampoo.jpg
2.jpg  5.jpg  8.jpg  antitickshampoo.jpg  clearmill.jpg  server.pyc
root@ubuntu-512mb-nyc1-01:/var/www/html/images/img#

```

Figure 3.3: digital ocean server Console

### 3.4.8 Database Connection Controls and Strings

### 3.4.9 Methodology

Database design methodology provided in this Topic is based on the guideline proposed by Connolly and Begg (2005). They have introduced three main phases of database design methodology, namely: conceptual, logical and physical database design. In this section we provide a description of what a design methodology is, and give a brief overview of these three main phases of database design.

Design methodology is an approach taken in designing or building things and it serves as a guideline on how things are done. Normally a design methodology is broken down into phases or stages and for each phase the detailed steps are outlined, and appropriate tools and techniques are specified. A design methodology is able to support and facilitate designers in

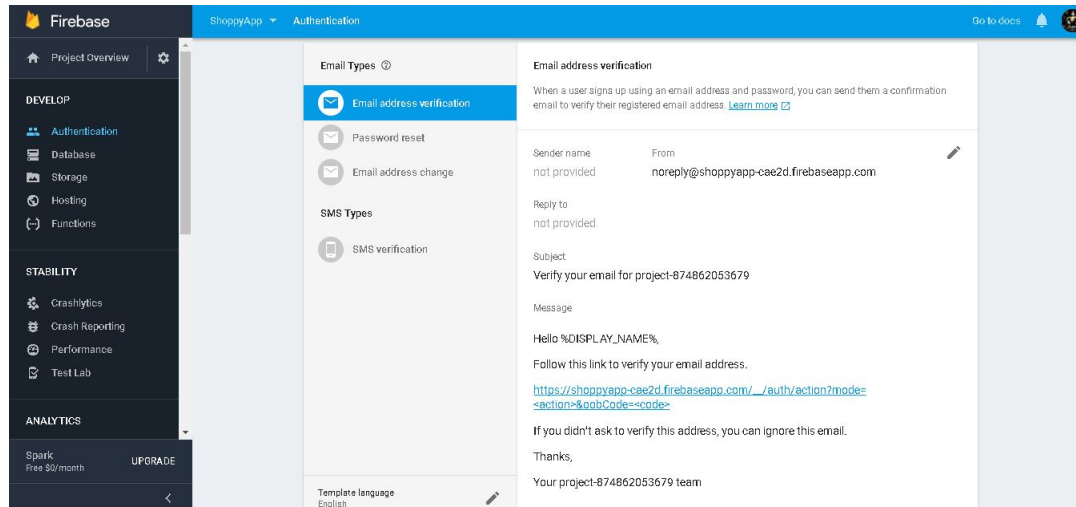


Figure 3.4: Firebase Authentication

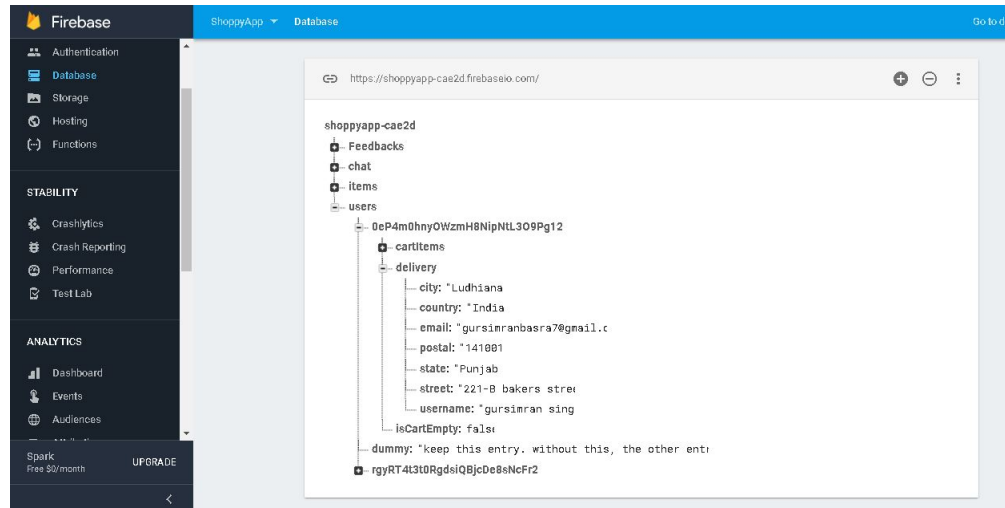


Figure 3.5: Firebase Database

planning, modeling, and managing a database development project in a structured and systematic manner. Validation is one of the key aspects in the design methodology as it helps to ensure that the produced models accurately represent the user requirement specifications. As mentioned earlier, we are going to adopt the database design methodology proposed by Connolly and Begg for our discussion in this Topic. The methodology consists of three main phases, starting with conceptual, then logical and lastly the physical database design phase. The conceptual database design is aimed to produce a conceptual representation of the required database. The core activity in this phase involves the use of ER modelling in which the entities, relationship and attributes are defined. For the logical design phase, the aim is to map the conceptual model which is represented by the ER model to the logical structure of the database. Among the activities involved in this phase is the use of normalisation process to derive and validate relations. In the physical design phase, the emphasis is to translate the logical structure to the physical implementation of the database using the defined database management system. Besides the above three main phases, this methodology has also outlined eight core steps. The Step 1 is focused on the conceptual database design phase, the Step 2

is focused on the logical database design phase, and the Step 3 to Step 8 are focused on the physical database design phase.

# Chapter 4

## Implementation ,Testing and Maintenance

### 4.1 Introduction to Languages,IDEs,Tools and Technologies used for Implementation

#### 4.1.1 Technologies used

##### 4.1.1.1 Firebase

Firebase helps you build better mobile apps and grow your business.

Firebase is a mobile platform from Google offering a number of different features that you can pick n mix from. Specifically, these features revolve around cloud services, allowing users to save and retrieve data to be accessed from any device or browser. This can be useful for such things as cloud messaging, hosting, crash reporting, notifications, analytics and even earning money through AdMob.

It works with Android apps, iOS apps and web apps and best of all: its free!

- **Setting up a project**

Before you can do anything with Firebase, you first need to create an account. You can do this over at [firebase.google.com](https://firebase.google.com/).

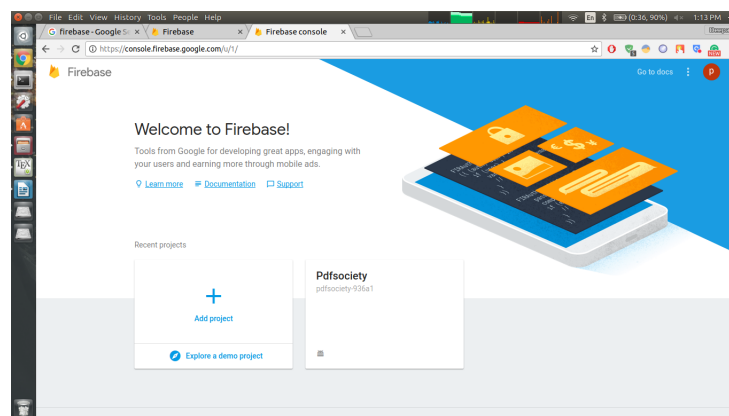


Figure 4.1: Firebase Console

- **Firebase Products**

- **Firebase Authentication**

Firebase Authentication aims to make building secure authentication systems easy, while improving the sign-in and onboarding experience for end users. It provides an

Mix and match complementary products



Figure 4.2: Firebase Products

end-to-end identity solution, supporting email and password accounts, phone auth, and Google, Twitter, Facebook, and GitHub login, and more.

Most apps need to know the identity of a user. Knowing a user's identity allows an app to securely save user data in the cloud and provide the same personalized experience across all of the user's devices. Firebase Authentication provides backend services, easy-to-use SDKs, and ready-made UI libraries to authenticate users to your app. It supports authentication using passwords, phone numbers, popular federated identity providers like Google, Facebook and Twitter, and more.

Firebase Authentication integrates tightly with other Firebase services, and it leverages industry standards like OAuth 2.0 and OpenID Connect, so it can be easily integrated with your custom backend.

#### – **Firebase Realtime Database**

Store and sync data with our NoSQL cloud database. Data is synced across all clients in realtime, and remains available when your app goes offline.

The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in realtime to every connected client. When you build cross-platform apps with our iOS, Android, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data.

#### – **Cloud Storage**

Cloud Storage is built for app developers who need to store and serve user-generated content, such as photos or videos.

Cloud Storage for Firebase is a powerful, simple, and cost-effective object storage service built for Google scale. The Firebase SDKs for Cloud Storage add Google security to file uploads and downloads for your Firebase apps, regardless of network



quality. You can use our SDKs to store images, audio, video, or other user-generated content. On the server, you can use Google Cloud Storage, to access the same files.

#### 4.1.1.2 Android

Android provides a rich application framework that allows you to build innovative apps and games for mobile devices in a Java language environment. The documents listed in the left navigation provide details about how to build apps using Android's various APIs. The various fundamental concepts about the Android app framework:



Figure 4.3: Android Anatomy

Apps provide multiple entry points. Android apps are built as a combination of distinct components that can be invoked individually. For instance, an individual activity provides a single screen for a user interface, and a service independently performs work in the background. From one component you can start another component using an intent. You can even start a component in a different app, such as an activity in a maps app to show an address. This model provides multiple entry points for a single app and allows any app to behave as a user's "default" for an action that other apps may invoke.

Apps adapt to different devices, Android provides an adaptive app framework that allows you to provide unique resources for different device configurations. For example, you can create different XML layout files for different screen sizes and the system determines which layout to apply based on the current device's screen size. You can query the availability of device features at runtime if any app features require specific hardware such as a camera. If necessary, you can also declare features your app requires so app markets such as Google Play Store do not allow installation on devices that do not support that feature. Android comes with an Android market which is an online software store. It was developed by Google.

It allows Android users to select, and download applications developed by third party developers and use them. There are around 2.0 lakh+ games, application and widgets available on the market for users. Android applications are written in java programming language. Android is available as open source for developers to develop applications which can be further used for selling in android market. There are around 200,000 applications developed for android with over 3 billion+ downloads. Android relies on Linux version 2.6 for core system services.



such as security, memory management, process management, network stack, and driver model. For software development, Android provides Android SDK (Software development kit).

- **Activity Lifecycle** Activities in the system are managed as an activity stack. When a new activity is started, it is placed on the top of the stack and becomes the running activity. The previous activity always remains below it in the stack, and will not come to the foreground again until the new activity exits. An activity has essentially four states: If an activity is in the foreground of the screen (at the top of the stack), it is active or running.

If an activity has lost focus but is still visible (that is, a new non-full-sized or transparent activity has focus on top of your activity), it is paused. A paused activity is completely alive (it maintains all state and member information and remains attached to the window manager), but can be killed by the system in extreme low memory situations.

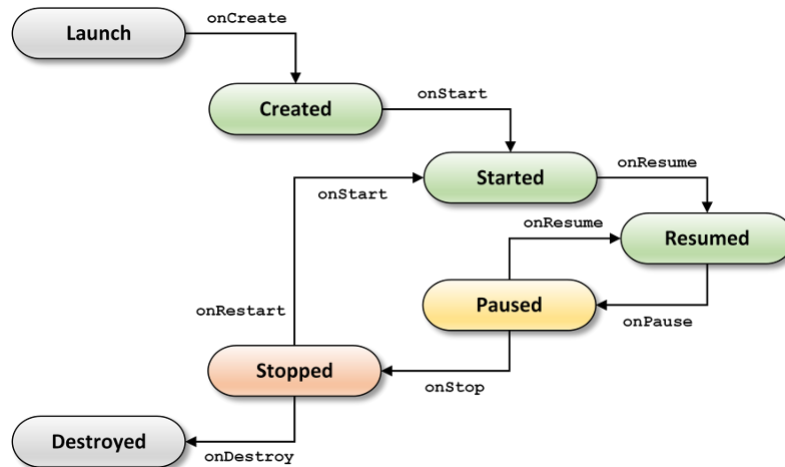


Figure 4.4: Activity Life Cycle

If an activity is completely obscured by another activity, it is stopped. It still retains all state and member information, however, it is no longer visible to the user so its window is hidden and it will often be killed by the system when memory is needed elsewhere.

If an activity is paused or stopped, the system can drop the activity from memory by either asking it to finish, or simply killing its process. When it is displayed again to the user, it must be completely restarted and restored to its previous state.

- **Libraries used in making this application**

- **Android Support Library** Android Support Library, Card View, RecyclerView, Material : The application will follow Material design guidelines.

- **Impliment UI for Each Activity and Fragment**

- UI for MainActivity
- UI for Expenses Activity
- UI for Expense Detail that will show last known expenses in the same category as a summary.

- UI for Categories Fragment
- UI for Statistics fragment
- **Expenses Fragment** Implement Business Logic for Adding and removing Expenses. Including removing from recycler view:
  - \* Add expense or income
  - \* Remove expense or income from recycler view after enter to the detail page.
- **Categories Fragment**
  - \* Categories add and remove from recycler view.
  - \* Detail of category with graph showing last used in the present week.
- **Statistics Fragment** Picker to select dates to make the query.
  - \* Show Graph for expenses made that month
  - \* Show graph for expenses made by category

### 4.1.2 Picasso

Displaying images is easiest using a third party library such as Picasso from Square which will download and cache remote images and abstract the complexity behind an easy to use DSL. Picasso allows for hassle-free image loading in your application often in one line of code!

```
Picasso.with(context).load("http://i.imgur.com/DvpvklR.png").into(imageView)
```

Many common pitfalls of image loading on Android are handled automatically by Picasso:

- Handling ImageView recycling and download cancelation in an adapter.
- Complex image transformations with minimal memory use.
- Automatic memory and disk caching.

### 4.1.3 API.AI

We'll use DialogFlow previously called API.AI to process natural language, that is understand what users want. Chat bot will communicate to it's customers via the Facebook Messenger. And in the second part we'll use Node.js to upgrade the bot. So the basic knowledge of javascript and Node.js is needed.

Chatbots are computer programs that you chat with, for example within messaging apps. Imagine you wanted to buy something from an online retailer. Today you would go to their website, look around until you found what you wanted, and then place an order. But if said store had a chat bot, you would simply be able to send them a message that you are looking for a particular item, and the bot would respond. You would be having a conversation, if you will, with the stores bot. Such an experience mimics that of going into an actual physical store and speaking with a salesperson.

Facebook and Microsoft, among others, are emerging as leaders in this space, dedicating serious resources so developers can create these bots. In the past six months, Facebook Messenger rolled out over 30,000 bots with more on the way. Facebook believes Messenger

can become a primary channel for businesses to interact with their customers, consequently replacing 1-800 numbers with a mix of artificial intelligence and human intervention.

But it turns out that chatbots arent a new technology. The idea of a humans and machines interacting with one another isnt a radical concept. Over several decades weve been trying to create machines that mimic human behavior. The idea of a chat bot within our messaging apps is actually quite logical given the history.

Today's bots arent trying to pass for humans, unlike their predecessors, rather their job is to provide information or complete tasks for the humans they interact with. But wait, how does that work? But, how do these bots work? How do they know how to talk to people and answer questions? Isnt that artificial intelligence?

There are two types of chatbots, one functions based on a set of rules, and the other more advanced version uses machine learning.

Chatbot that functions based on rules are very limited. It can only respond to very specific commands. If you say the wrong thing, it doesnt know what you mean.

Chatbot that functions using machine learning, on the other hand, has an artificial brain AKA artificial intelligence. It understands language, not just commands. This bot continuously gets smarter as it learns from conversations it has with people.

It is the latter bots that will become smarter as more and more energy is spent on developing them.

#### **4.1.4 Material Design**

Snackbars contain a single line of text directly related to the operation performed. They may contain a text action, but no icons.

Toasts (Android only) are primarily used for system messaging. They also display at the bottom of the screen, but may not be swiped off-screen.

##### **4.1.4.1 Usage**

Only one snackbar may be displayed at a time. Each snackbar may contain a single action, neither of which may be Dismiss or Cancel.

##### **4.1.4.2 Behaviour**

Snackbars animate upwards from the bottom edge of the screen.

##### **4.1.4.3 Snackbar specs**

- Action button: Roboto Medium 14sp, all-caps text.
- Mobile height: 48dp (single-line), 80dp (multi-line).
- Desktop snackbar height: 48dp

#### **4.1.5 Language used**

##### **4.1.5.1 JAVA**

Java is a platform-independent programming language used to create secure and robust application that may run on a single computer or may be distributed among servers and clients

over a network.

Java features such as platform-independency and portability ensure that while de-veloping Java EE enterprise applications, you do not face the problems related to hardware , network , and the operating system.

Java was started as a project called "Oak" by James Gosling in June 1991. Gosling's goals were to implement a virtual machine and a language that had a familiar C like notation but with greater uniformity and simplicity than C/C++. The First publication of Java 1.0 was released by Sun Microsystems in 1995. It made the promise of "Write Once, Run Anywhere", with free runtimes on popular platforms. In 2006-2007 Sun released java as open source and andplatform independent soft-ware. Over time new enhanced versions of Java have been released. The current version of Java is Java 1.7 which is also known as Java 7.he Java virtual machine (JVM) is a software implementation of a computer that executes programs like a real machine. The Java virtual machine is written specifically for a specific operating system, e.g. for Linux a special implementation is required as well as for Windows.

Java programs are compiled by the Java compiler into bytecode. The Java virtual machine interprets this bytecode and executes the Java program. The Java runtime environment (JRE) consists of the JVM and the Java class li- braries and contains the necessary functionality to start Java programs. The JDK contains in addition the development tools necessary to create Java pro-grams. The JDK consists therefore of a Java compiler, the Java virtual machine, and the Java class libraries.

The characteristics and features of java are as follows :

- **Simple** Simple Java is a simple language because of its various features, Java Doesn't Support Pointers , Operator Overloading etc. It doesnt require unreferenced object because java support automatic garbage collection. Java provides bug free system due to the strong memory management.
- **OOPS** Object-Oriented Programming Language (OOPs) is the methodology which provide software development and maintenance by using object state, behavior Programming Lan-guage must , and have properties. the Object Oriented following characteristics.
  - Encapsulation
  - Polymor-phism
  - Inheritance
  - Abstraction

As the languages like Objective C, C++ fulls the above four characteristics yet they are not fully object oriented lan-guages because they are structured as well as object oriented languages.In java everything is an Object. Java can be easily extended since it is based on the Object model.

- **Secure** Secure Java is Secure Language because of its many features it enables to de-velop virus-free, tamper-free systems. Authentication techniques are based on public-key encryption. Java does not support pointer explicitly for the memory. All Program Run under the sandbox.

- **Robust** Robust Java was created as a strongly typed language. Data type issues and problems are resolved at compile-time, and implicit casts of a variable from one type to another are not allowed.
- **Platform-independent** Platform-independent Java Language is platform-independent due to its hardware and software environment. Java code can be run on multiple platforms e.g. windows, Linux, sun Solaris, Mac/Os etc. Java code is compiled by the compiler and converted into byte code. This byte code is a platform independent code because it can be run on multiple platforms i.e. Write Once and Run Anywhere(WORA).
- **Architectural Neutral** Architecture neutral It is not easy to write an application that can be used on Windows , UNIX and a Macintosh. And its getting more complicated with the move of windows to non Intel CPU architectures. Java takes a different approach. Because the Java compiler creates byte code instructions that are subsequently interpreted by the java interpreter, architecture neutrality is achieved in the implementation of the java interpreter for each new architecture.
- **Portable** Portable Java code is portable. It was an important design goal of Java that it be portable so that as new architectures(due to hardware, operating system, or both) are developed, the java environment could be ported to them. In java, all primitive types(integers, longs, floats, doubles, and so on) are of defined sizes, regardless of the machine or operating system on which the program is run. This is in direct contrast to languages like C and C++ that leave the sized of primitive types up to the compiler and developer. Additionally, Java is portable because the compiler itself is written in Java.
- **Dynamic** Dynamic Because it is interpreted , Java is an extremely dynamic language, At runtime, the java environment can extends itself by linking in classes that may be located on remote servers on a network(for example, the internet) At runtime, the java interpreter performs name resolution while linking in the necessary classes. The Java interpreter is also responsible for determining the placement of object in memory. These two features of the Java interpreter solve the problem of changing the definition of a class used by other classes.
- **Interpreted** We all know that Java is an interpreted language as well. With an interpreted language such as Java, programs run directly from the source code. The interpreter program reads the source code and translates it on the fly into computations. Thus, Java as an interpreted language depends on an interpreter program. The versatility of being platform independent makes Java to outshine from other languages. The source code to be written and distributed is platform independent. Another advantage of Java as an interpreted language is its error debugging quality. Due to this any error occurring in the program gets traced. This is how it is different to work with Java.
- **High performance** High performance For all but the simplest or most infrequently used applications, performance is always a consideration for most applications, including graphics-intensive ones such as are commonly found on the world wide web, the performance of java is more than adequate.
- **Multithreading** Writing a computer program that only does a single thing at a time is an artificial constraint that lived with in most programming languages. With java, we no longer have to live with this limitation. Support for multiple, synchronized threads is

built directly into the Java language and runtime environment. Synchronized threads are extremely useful in creating distributed, network-aware applications. Such as application may be communicating with a remote server in one thread while interacting with a user in a different thread.

- **Distributed** Java facilitates the building of distributed application by a collection of classes for use in networked applications. By using `java.net.URL` (Uniform Resource Locator) class, an application can easily access a remote server. Classes also are provided for establishing socket-level connections.

#### 4.1.5.2 XML

Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable through use of tags that can be created and defined by users. Much like natural language is extensible (that is, can grow) when speakers create new words and agree on what they mean, XML is a markup language that can grow when users create new elements and agree on what they mean. For example, XML can markup machine-readably that apples and bananas are types of fruit, which is semantically deeper than the purpose of HTML. However, HTML is useful for display of content; often HTML is used to display XML content after transformation with XSL.

#### 4.1.6 IDE used

##### 4.1.6.1 Android Studio



Figure 4.5: Android Studio

Android Studio is Android’s official IDE. It is purpose built for Android to accelerate your development and help you build the highest-quality apps for every Android device.

It offer tools custom-tailored for Android developers, including rich code editing, debugging, testing, and profiling tools.

#### 4.1.7 Introduction to $\text{\LaTeX}$

$\text{\LaTeX}$ , I had never heard about this term before doing this project, but when I came to know about it’s features, it is just excellent.  $\text{\LaTeX}$ (pronounced /letk/, /letx/, /ltx/, or /ltk/)

Figure 4.6:  $\text{\LaTeX}$  Logo

is a document markup language and document preparation system for the  $\text{\TeX}$  typesetting program. Within the typesetting system, its name is styled as  $\text{\LaTeX}$ .

Figure 4.7: Donald Knuth, Inventor Of  $\text{\TeX}$  typesetting system

Within the typesetting system, its name is styled as  $\text{\LaTeX}$ . The term  $\text{\LaTeX}$  refers only to the language in which documents are written, not to the editor used to write those documents. In order to create a document in  $\text{\LaTeX}$ , a `.tex` file must be created using some form of text editor. While most text editors can be used to create a  $\text{\LaTeX}$  document, a number of editors have been created specifically for working with  $\text{\LaTeX}$ .

$\text{\LaTeX}$  is most widely used by mathematicians, scientists, engineers, philosophers, linguists, economists and other scholars in academia. As a primary or intermediate format, e.g., translating DocBook and other XML-based formats to PDF,  $\text{\LaTeX}$  is used because of the high quality of typesetting achievable by  $\text{\TeX}$ . The typesetting system offers programmable desktop publishing features and extensive facilities for automating most aspects of typesetting and desktop publishing, including numbering and cross-referencing, tables and figures, page layout and bibliographies.

$\text{\LaTeX}$  is intended to provide a high-level language that accesses the power of  $\text{\TeX}$ .  $\text{\LaTeX}$  essentially comprises a collection of  $\text{\TeX}$  macros and a program to process  $\text{\LaTeX}$  documents. Because the  $\text{\TeX}$  formatting commands are very low-level, it is usually much simpler for end-users to use  $\text{\LaTeX}$ .

#### 4.1.7.1 Typesetting

$\text{\LaTeX}$  is based on the idea that authors should be able to focus on the content of what they are writing without being distracted by its visual presentation. In preparing a  $\text{\LaTeX}$  document, the author specifies the logical structure using familiar concepts such as chapter, section, table, figure, etc., and lets the  $\text{\LaTeX}$  system worry about the presentation of these structures. It therefore encourages the separation of layout from content while still allowing manual typesetting adjustments where needed.

```
\documentclass[12pt]{article}
\usepackage{amsmath}
\title{\LaTeX}
\begin{document}
  \maketitle
  \LaTeX{} is a document preparation system
  for the \TeX{} typesetting program.
  \par
   $E=mc^2$ 
\end{document}
```

#### 4.1.7.2 Installing L<sup>A</sup>T<sub>E</sub>X on System

Installation of L<sup>A</sup>T<sub>E</sub>X on personal system is quite easy. As i have used L<sup>A</sup>T<sub>E</sub>X on Ubuntu 13.04 so i am discussing the installation steps for Ubuntu 13.04 here:

- Go to terminal and type

*sudo apt-get install texlive-full*

- Your Latex will be installed on your system and you can check for manual page by typing.

*man latex*

in terminal which gives manual for latex command.

- To do very next step now one should stick this to mind that the document which one is going to produce is written in any type of editor whether it may be your most common usable editor Gedit or you can use vim by installing first vim into your system using command.

*sudo apt-get install vim*

- After you have written your document it is to be embedded with some set of commands that Latex uses so as to give a structure to your document. Note that whenever you wish your document to be looked into some other style just change these set of commands.
- When you have done all these things save your piece of code with .tex format say test.tex. Go to terminal and type

*latex path of the file test.tex Or pdflatex path of the file test.tex*

*eg: pdflatex test.tex*

for producing pdf file simultaneously.

After compiling it type command

*evince filename.pdf*

*eg: evince test.pdf*

To see output pdf file.



### 4.1.7.3 Pdfscreen L<sup>A</sup>T<sub>E</sub>X

There are some packages that can help to have unified document using L<sup>A</sup>T<sub>E</sub>X. Example of such a package is pdfscreen that let the user view its document in two forms-print and screen. Print for hard copy and screen for viewing your document on screen. Download this package from [www.ctan.org/tex-archive/macros/latex/contrib/pdfscreen/](http://www.ctan.org/tex-archive/macros/latex/contrib/pdfscreen/). Then install it using above mention method.

To test it the test code is given below:-

Just changing print to screen gives an entirely different view. But for working of pdfscreen another package required are comment and fancybox.

The fancybox package provides several different styles of boxes for framing and rotating content in your document. Fancybox provides commands that produce square-cornered boxes with single or double lines, boxes with shadows, and round-cornered boxes with normal or bold lines. You can box mathematics, floats, center, flushleft, and flushright, lists, and pages.

Whereas comments package selectively include/excludes portions of text. The comment package allows you to declare areas of a document to be included or excluded. One need to make these declarations in the preamble of your file. The package uses a method for exclusion that is pretty robust, and can cope with ill-formed bunches of text.

So these extra packages needed to be installed on system for the proper working of pdfscreen package.

## 4.2 Coding standards of Language used

### 4.2.1 Introduction

This document is the definition of Google's coding standards for source code in the Java Programming Language. A Java source file is described as being in Google Style if and only if it adheres to the rules herein. Like other programming style guides, the issues covered span not only aesthetic issues of formatting, but other types of conventions or coding standards as well. However, this document focuses primarily on the hard-and-fast rules that we follow universally, and avoids giving advice that isn't clearly enforceable (whether by human or tool).

### 4.2.2 Source file basics

- File name, the source file name consists of the case-sensitive name of the top-level class it contains (of which there is exactly one), plus the .java extension.
- File encoding: UTF-8 Source files are encoded in UTF-8. Aside from the line terminator sequence, the ASCII horizontal space character (0x20) is the only whitespace character that appears anywhere in a source file. This implies that:
- All other whitespace characters in string and character literals are escaped.
- Tab characters are not used for indentation.

## 4.3 GANTT chart

A Gantt chart, commonly used in project management, is one of the most popular and useful ways of showing activities (tasks or events) displayed against time. On the left of the chart is a list of the activities and along the top is a suitable time scale. Each activity is represented by a bar; the position and length of the bar reflects the start date, duration and end date of the activity. Today, Gantt charts are most commonly used for tracking project schedules. For this it is useful to be able to show additional information about the various tasks or phases of the project, for example how the tasks relate to each other, how far each task has progressed, what resources are being used for each task and so on.

### Project Planner

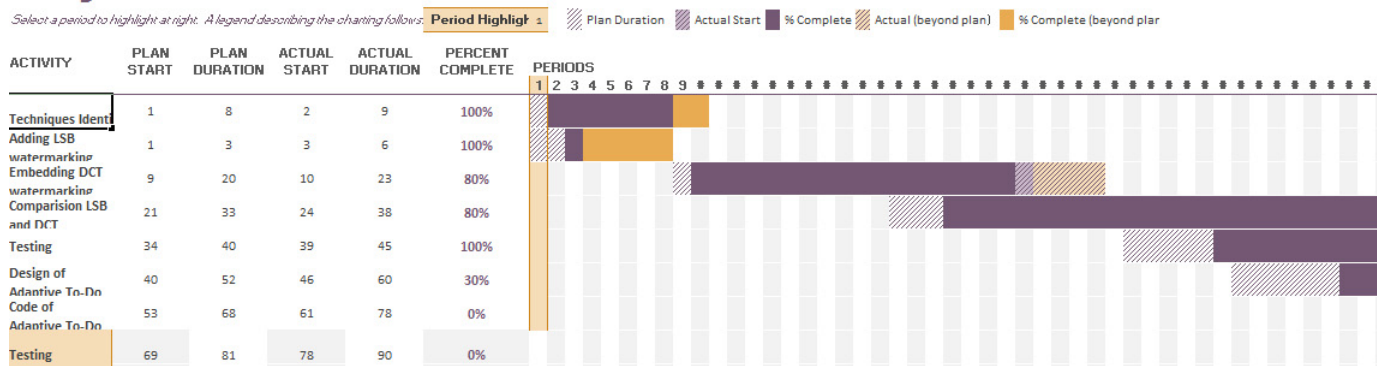


Figure 4.8: Gantt Chart

## 4.4 Testing Techniques and Test Plans

### 4.4.1 xUnit Framework

xUnit is the collective name for several unit testing frameworks that derive their structure and functionality from Smalltalk's SUnit. SUnit, designed by Kent Beck in 1998, was written in a highly structured object-oriented style, which lent easily to contemporary languages such as Java and C. Following its introduction in Smalltalk the framework was ported to Java by Kent Beck and Erich Gamma and gained wide popularity, eventually gaining ground in the majority of programming languages in current use. The names of many of these frameworks are a variation on "SUnit", usually replacing the "S" with the first letter (or letters) in the name of their intended language ("JUnit" for Java, "RUnit" for R etc.). These frameworks and their common architecture are collectively known as "xUnit".

### 4.4.2 JUnit

JUnit is a unit testing framework for the Java programming language. JUnit has been important in the development of test-driven development, and is one of a family of unit testing frameworks which is collectively known as xUnit that originated with SUnit. JUnit is linked as a JAR at compile-time; the framework resides under package junit.framework for JUnit 3.8

and earlier, and under package org.junit for JUnit 4 and later. A research survey performed in 2013 across 10,000 Java projects hosted on GitHub found that JUnit, (in a tie with slf4j-api), was the most commonly included external library. Each library was used by 30.7 percentage of projects.

### 4.4.3 Robotium Android Testing Tool

Robotium is one the first and frequently utilized automated testing tools for software supported on Android. Robotium is a free Android UI testing tool. It is suitable for tests automation for different Android versions and sub-versions. Software developers often describe it as Selenium for Android. Tests created by Robotium are written in Java. In fact, Robotium is a library for unit tests. But it takes much time and efforts to create tests by means of Robotium, as one must work with the program source code in order to automate tests. The tool is also unsuitable for interaction with system software; it cannot lock and unlock a smartphone or a tablet. There is no Record and Play function in Robotium, and it does not provide screenshots.

### 4.4.4 MonkeyRunner Android App Testing

MonkeyRunner Android App Testing MonkeyRunner is one of popular Android Testing tools used for automating of functional tests for Android software. This tool is more low-level than Robotium is. One does not have to deal with the source code in order to automate tests. The tests are written in Python, one may use a recording tool for creating tests. MonkeyRunner can run tests on real devices connected to a PC or emulators. The tool has an API what allows it to control a smartphone, a tablet or an emulator from outside of Android code. A significant disadvantage of the mobile app testing tool is that it is necessary to write scripts for each device. Another problem of MonkeyRunner is that the tests require adjustments each time when user interface of the tested program is changed.

# Chapter 5

## Results and Discussions

### 5.1 User Interface Representation (Of Respective Project)

No matter how new technologies change our lifestyle, people will always shop. Developing and designing mobile apps for ecommerce businesses has its own tricks. And online shopping follows adheres to the same principles as offline shopping. If customers dont like what they see, they wont buy. Thats why intuitive interfaces and catchy designs are invaluable for any ecommerce app.

We analyse user behaviour and design apps that cater to our users needs. For this article, we asked our design department to share their tips and tricks for designing an ideal ecommerce app.

Ecommerce apps contain three essential components: a catalogue (list of goods), search and filter options, and a shopping cart (or checkout).

#### 5.1.1 Snapshots of system

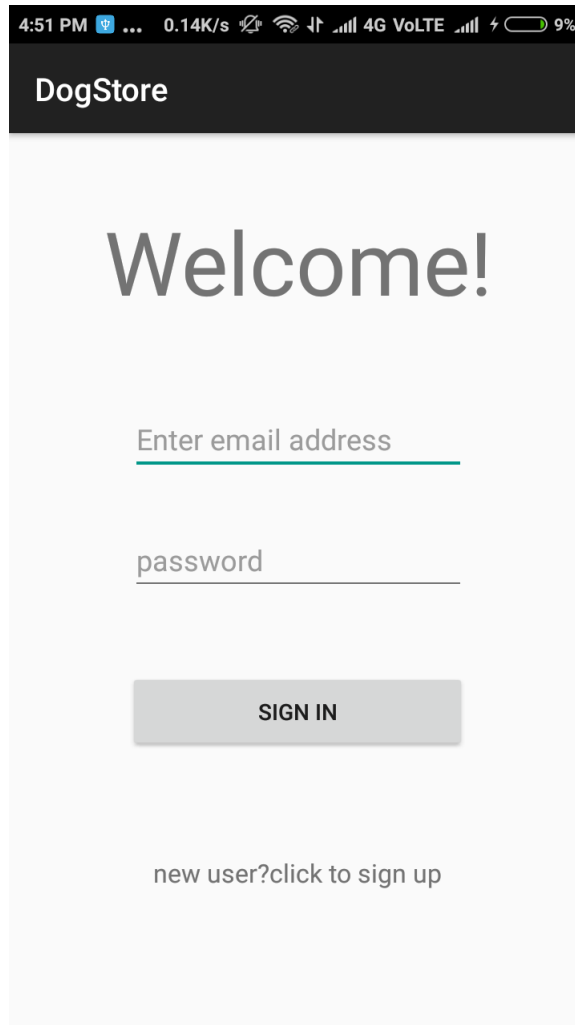


Figure 5.1: Login Screen

Here the back end connectivity used is firebase. When a user logs in to the account, the new account is made in firebase. In order to log in, one has to confirm the email in his account. Then he/she can log in to the application.

## 5.2 Back Ends Representation (Database to be used)

### 5.2.1 Firebase Realtime Database

Store and sync data with our NoSQL cloud database. Data is synced across all clients in realtime, and remains available when your app goes offline.

The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in realtime to every connected client. When you build cross-platform apps with our iOS, Android, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data.

### 5.2.2 How does it work?

The Firebase Realtime Database lets you build rich, collaborative applications by allowing secure access to the database directly from client-side code. Data is persisted locally, and even

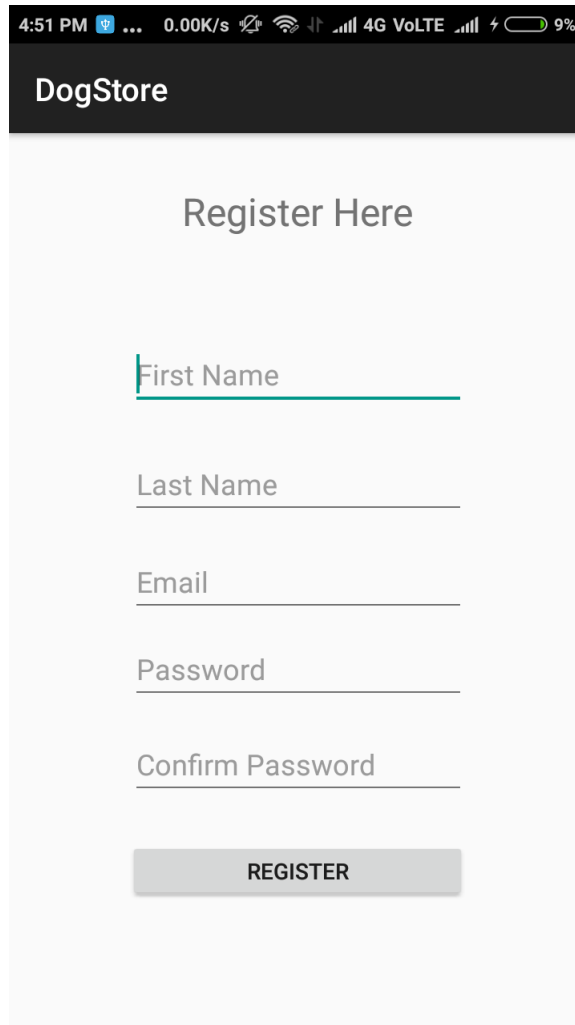
The image is a screenshot of a mobile application interface for 'DogStore'. At the top, there is a status bar showing the time as 4:51 PM, signal strength, and battery level at 9%. Below the status bar is a dark header with the text 'DogStore' in white. The main content area has a light gray background and is titled 'Register Here' in a bold, dark font. Below the title, there are five input fields, each with a label and a horizontal line for text entry: 'First Name', 'Last Name', 'Email', 'Password', and 'Confirm Password'. The 'First Name' field has a blue cursor. At the bottom of the form is a gray button with the text 'REGISTER' in white capital letters.

Figure 5.2: Register Screen

If the user does not have previous account details or he/she is a new user, he has to register first to login.

while offline, realtime events continue to fire, giving the end user a responsive experience. When the device regains connection, the Realtime Database synchronizes the local data changes with the remote updates that occurred while the client was offline, merging any conflicts automatically.

The Realtime Database provides a flexible, expression-based rules language, called Firebase Realtime Database Security Rules, to define how your data should be structured and when data can be read from or written to. When integrated with Firebase Authentication, developers can define who has access to what data, and how they can access it.

The Realtime Database is a NoSQL database and as such has different optimizations and functionality compared to a relational database. The Realtime Database API is designed to only allow operations that can be executed quickly. This enables you to build a great realtime experience that can serve millions of users without compromising on responsiveness. Because of this, it is important to think about how users need to access your data and then structure it accordingly.

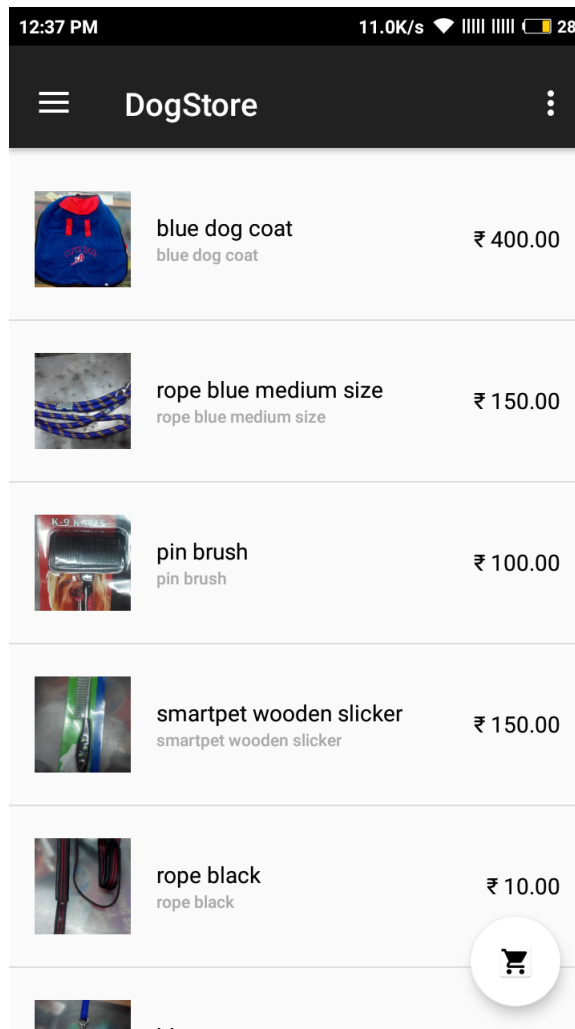


Figure 5.3: Items Cart

This photo represents the items which can be purchased from the application. this contain a custom list view with an adapter.

### 5.2.3 Snapshots of Database

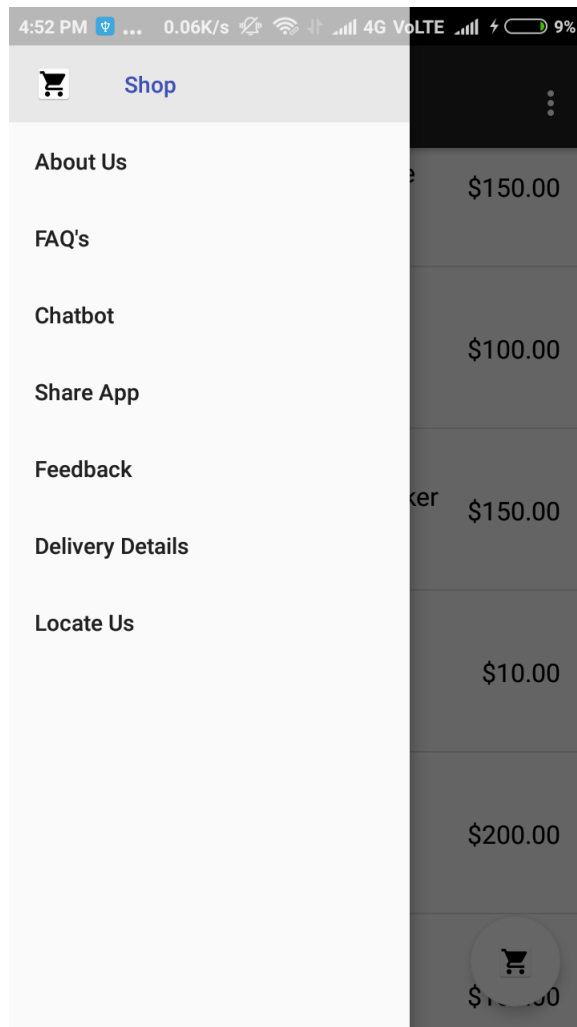


Figure 5.4: Navigation Drawer

This photo shows the navigation drawer. The library used in the development is of mike penz which completely solves our purpose to show the attached links of the project.



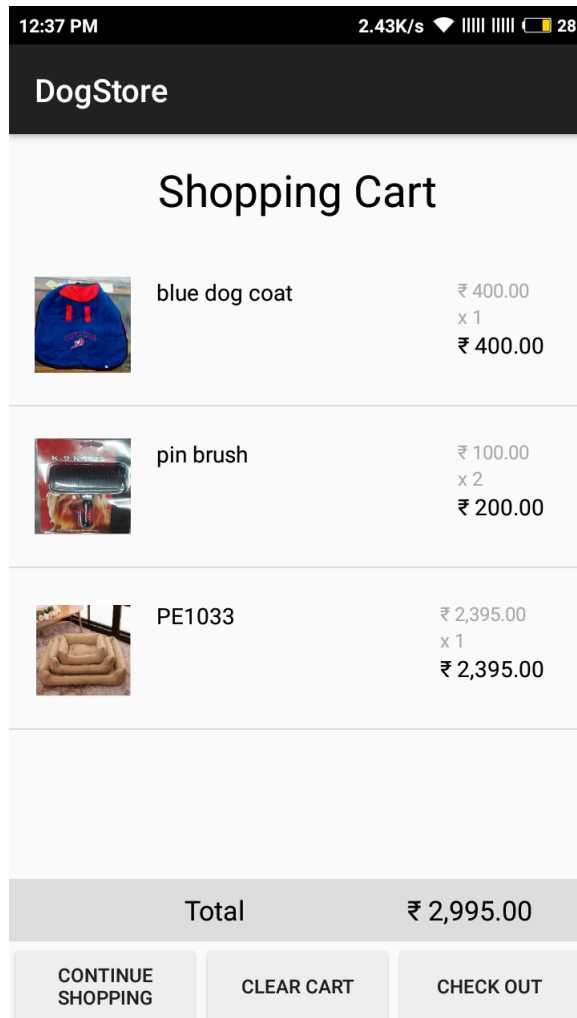


Figure 5.5: Shopping Cart

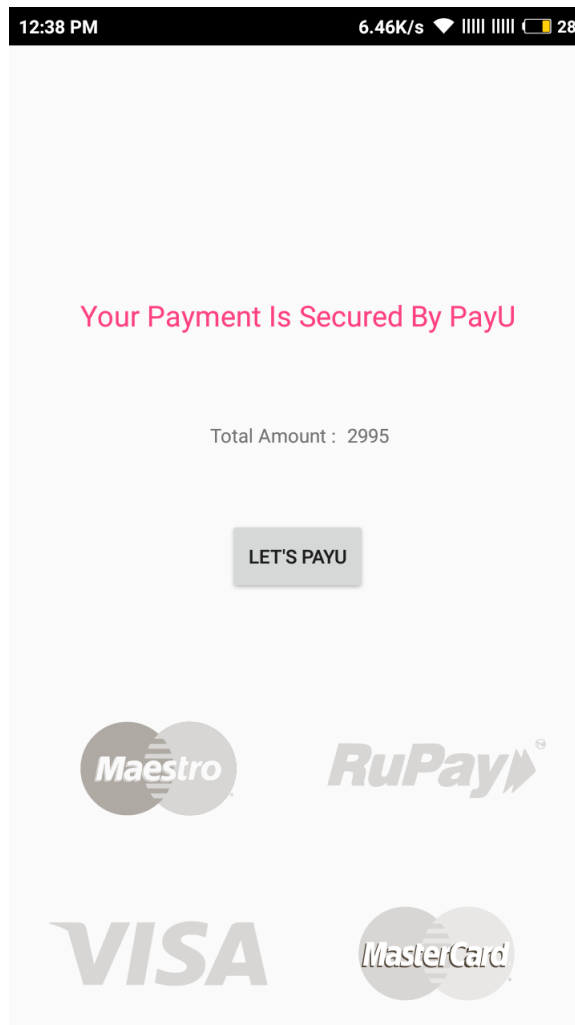


Figure 5.6: Payment Gateway (PayuMoney)

Payumoney is an affordable on line payment gateway provider in India. It enable small business to integrate on line payment gateway services without any set-up cost.

4:53 PM 0.00K/s 4G VoLTE 9%

DogStore

LOGIN SIGN UP

**PayUmoney**  
The best way to pay online

Email

☐ Guest Login

☐ Login with Password

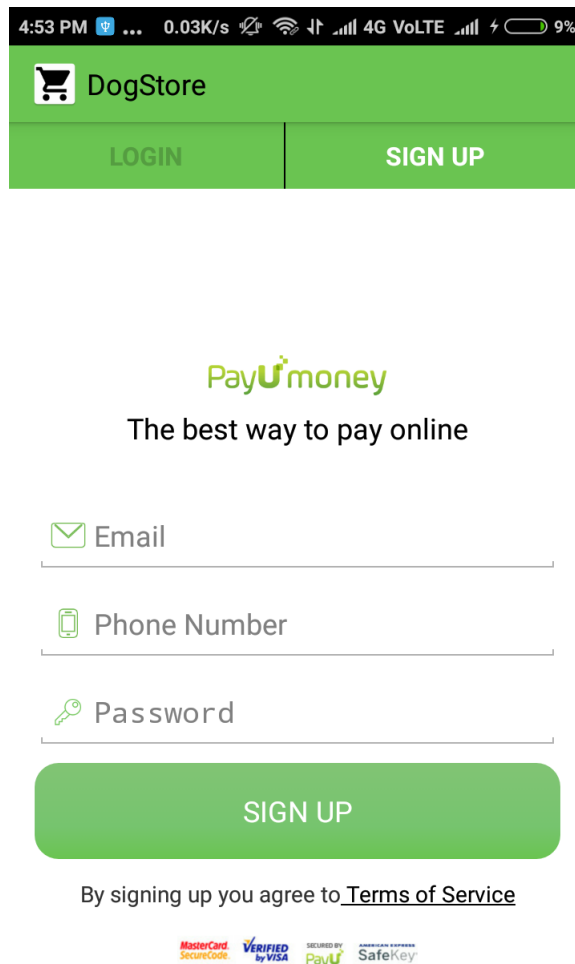
LOGIN

[Forgot Password?](#)


MasterCard SecureCode VERIFIED by VISA SECURED BY PayU SafeKey

Figure 5.7: Payment Gateway Login

In order to complete the payment successfully one has to fill his email id. This email id is taken to send the status of the payment to the email.





4:53 PM 0.03K/s 4G VoLTE 9%


 DogStore

LOGIN SIGN UP

**PayUmoney**  
The best way to pay online

 Email

 Phone Number

 Password

SIGN UP

By signing up you agree to [Terms of Service](#)










Figure 5.8: Register to gateway

You can also make account in the payment gateway for easy and fast transaction processing.  
You can also save your card which make payment more faster.

7:21 PM
0.00K/s
19

Amount to Pay  
₹ 2619.00
[Order Summary](#)

---

^ Debit Card

XXXX-XXXX-XXXX

MM/YYYY

CVV

PAY NOW

---

v Credit Card

---

v Net Banking

Figure 5.9: Payment

We can use debit card, credit card, net banking method for payment of the products purchased.

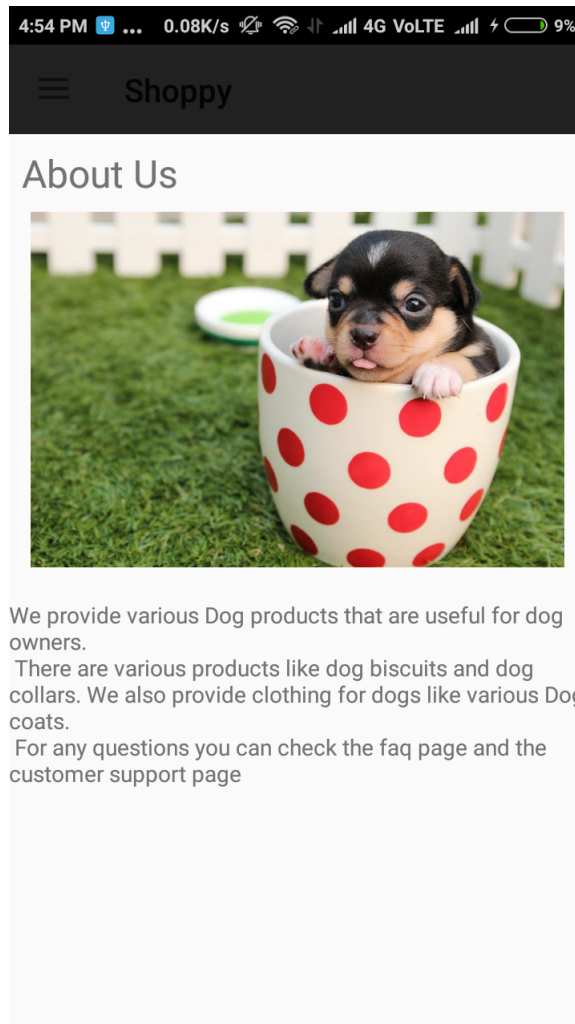


Figure 5.10: About us

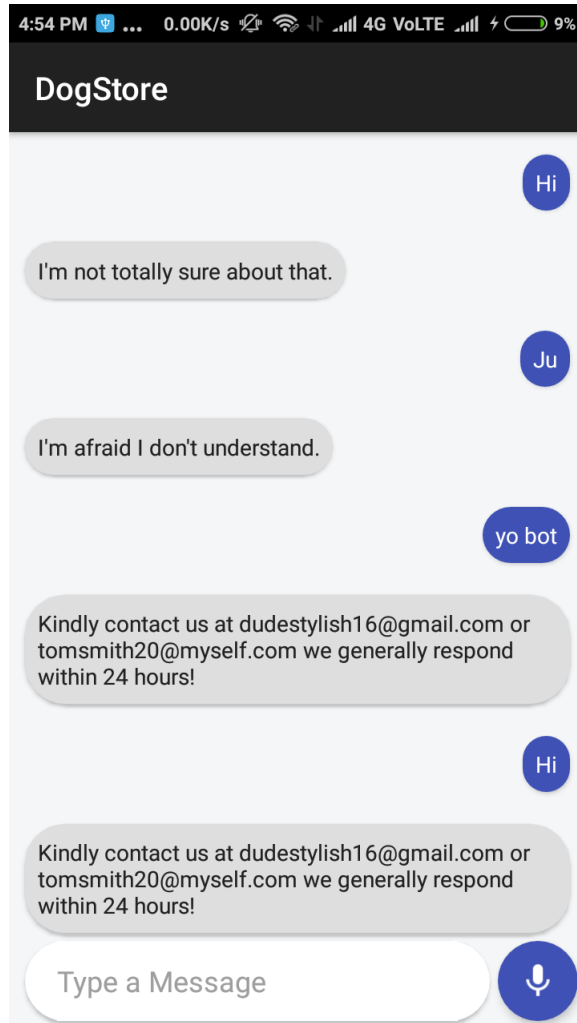


Figure 5.11: Enquiry Bot

This bot is developed in Api.ai (Dialogflow). It give users new ways to interact with your product by building engaging voice and text-based conversational interfaces powered by AI. Connect with users on the Google Assistant, Amazon Alexa, Facebook Messenger, and other popular platforms and devices.

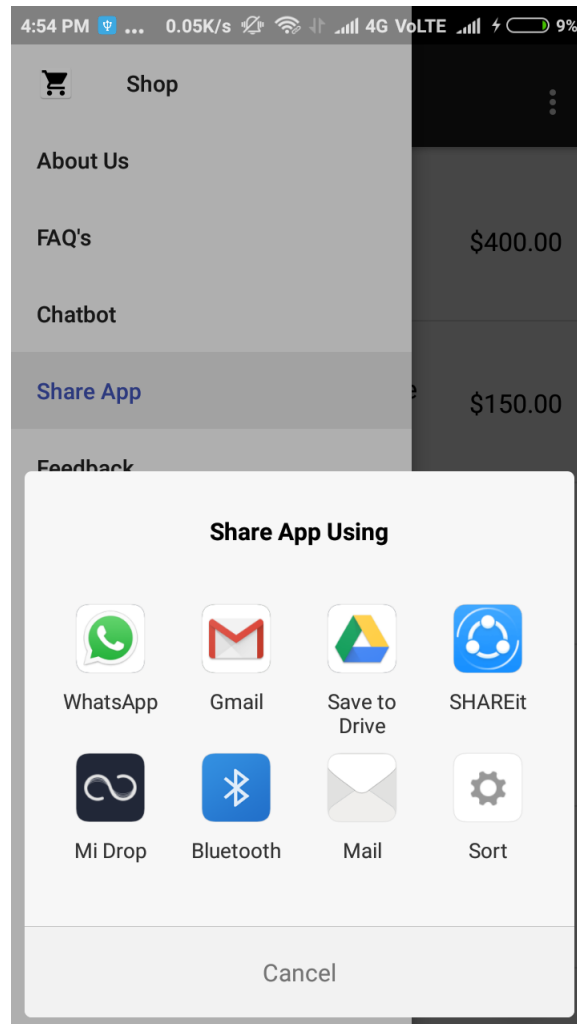
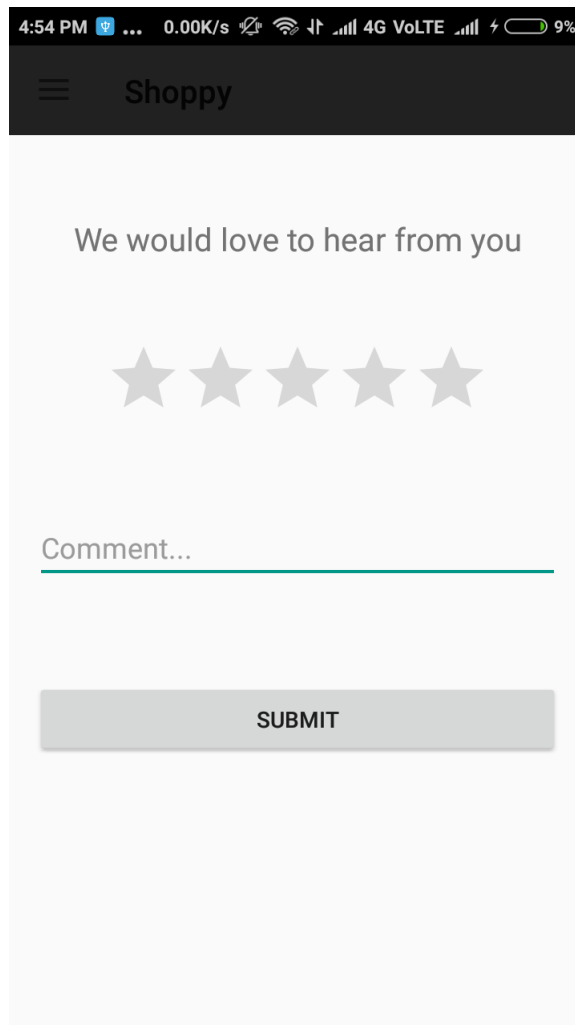


Figure 5.12: Share Application  
We can share our application with approximately all the platforms easily.





The screenshot shows a mobile application interface for a feedback form. At the top, a dark navigation bar contains a hamburger menu icon and the app name 'Shoppy'. Below this, the text 'We would love to hear from you' is centered. Underneath the text is a row of five gray stars for rating. A text input field with the placeholder 'Comment...' is positioned below the stars. At the bottom of the form is a gray rectangular button with the text 'SUBMIT' in all caps. The status bar at the very top of the screen shows the time as 4:54 PM, various connectivity icons, and a 9% battery level.

Figure 5.13: Feedback

We are ready to take the feedback from our user for the app improvement and UI/UX interaction for growth.

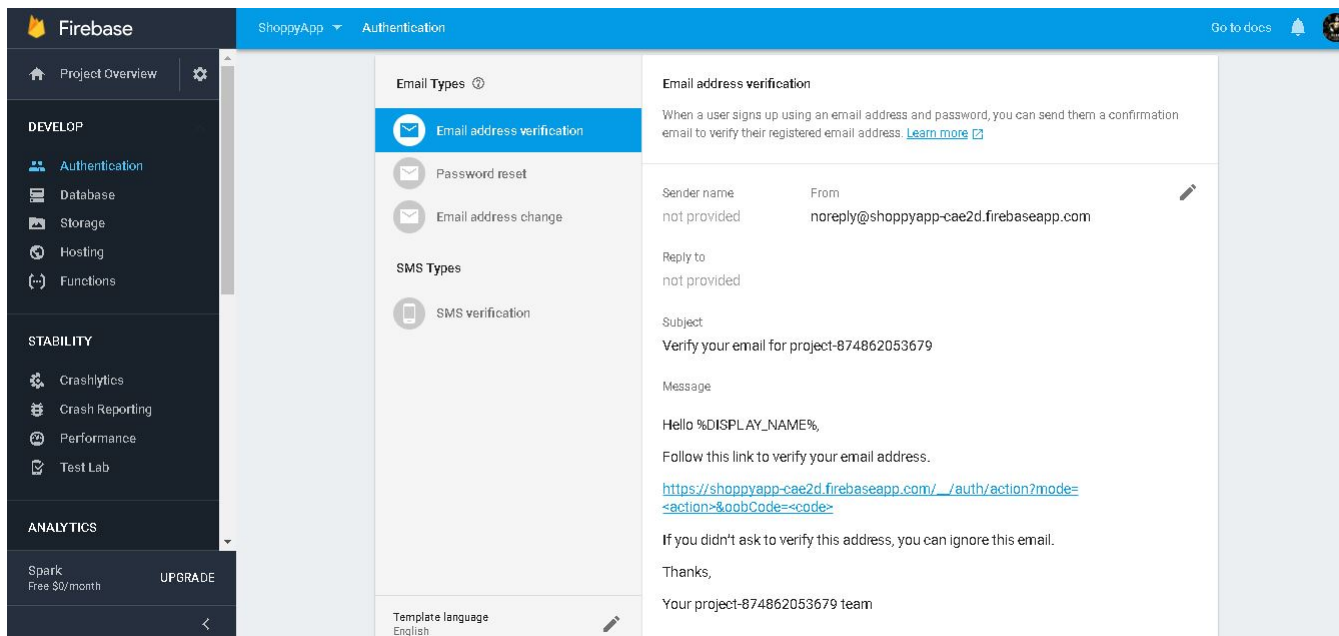


Figure 5.14: Confirmation Email in Fire base

# Chapter 6

## Conclusion and Future Scope

### 6.1 Conclusion

We have successfully developed the Application DOGSTORE With the help of various links and tools, we have been able to provide a mobile application which is live and running successfully. This project also provide the values and ethics of how to deal with the real customer. We have been successful in our attempt to take care of the needs of both the visitors as well as the administrator. Finally we hope that this will go a long way in popularizing the organization. The system has reached a steady state where all the bugs have been eliminated. The website is developed at a high level of efficiency and all the visitors associated with the application understand its advantage. The system solves the problem it was intended to solve as requirement specification.

### 6.2 Future Scope

The app uses android technology which has evergreen scope. The app obviously has a bright future scope as there is test which includes different level and type of questions. Moreover in Future, one can see his/her earlier records regarding the payments and can checkout the product with the use of image processing, by simply placing the camera on previous bought product. The platform used is android. Nowadays Android has become very popular which is an open-source, Linux-based operating system mainly designed by Google for smart-phones and tablets.

Many mobile Apps development industries are considering Android Application Development as one of the best business opportunities, for this they need to hire a lot of knowledgeable mobile application developer in future. This adds a big sign of scope of mobile Apps in future.

In the current job market of mobile application development, the need for inventive App developers is huge and still increasing. Android Apps development can also be taken up as a part time job. You can create your own applications at home and submit it to the Google Play store which can be downloaded by smart-phone users.

# Bibliography

- [1] Introduction to Android: Available at <http://developer.android.com/guide/index.html>
- [2] Android Training: Available at <http://developer.android.com/training/index.html>
- [3] Firebase Documentation: Available at <https://firebase.google.com/docs/android/setup/>
- [4] Picasso: Available at <http://square.github.io/picasso/>
- [5] Firebase Authentication: Available at <https://firebase.google.com/>
- [6] GmailBackGround: Available at <https://github.com/luongvo/GmailBackground/>
- [7] Material Drawer: Available at <https://github.com/mikepenz/MaterialDrawer/>
- [8] Payumoney: Available at <https://payumoney.com/>
- [9] Digitalocean Server: Available at <https://www.digitalocean.com/>