

ATMIYA UNIVERSITY

RAJKOT



A

Report On

E-VEGETABLE MARKET MANAGEMENT

Under subject of

MAJOR PROJECT

B.TECH, Semester – VII

(Computer Engineering)

Submitted by:

Prachi Pravinbhai Ardesna (201002002)

Moxa Prakashbhai Sodha (201002025)

Prof. Nirali Borad

(Faculty Guide)

Prof. Tosai M. Bhalodia

(Head of the Department)

Academic Year

(2020-21)

CANDIDATE'S DECLARATION

We hereby declare that the work presented in this project entitled “**E-VEGETABLE MARKET MANAGEMENT**” submitted towards completion of project in **7th Semester** of B. Tech. (Computer Engineering) is an authentic record of our original work carried out under the guidance of “**Prof. Nirali Borad**”.

We have not submitted the matter embodied in this project for the award of any other degree.

Semester: 7th

Place: Rajkot

Signature:

Prachi Ardesna (201002002)

Moxa Sodha(201002025)

ATMIYA UNIVERSITY

RAJKOT



CERTIFICATE

Date:

This is to certify that the “**E-VEGETABLE MARKET MANAGEMENT**” has been carried out by **Prachi Ardeshta** under my guidance in fulfillment of the subject Mini Project in **COMPUTER ENGINEERING (7th Semester)** of Atmiya University, Rajkot during the academic year 2021.

Prof. Nirali Borad

(Project Guide)

Prof. Tosal M. Bhalodia

(Head of the Department)

ATMIYA UNIVERSITY

RAJKOT



CERTIFICATE

Date:

This is to certify that the “**E-VEGETABLE MARKET MANAGEMENT**” has been carried out by **Moxa Sodha** under my guidance in fulfillment of the subject Mini Project in **COMPUTER ENGINEERING (7th Semester)** of Atmiya University, Rajkot during the academic year 2021.

GUIDE:

Prof. Nirali Borad

Prof. Tosal M. Bhalodia

(Head of the Department)

ACKNOWLEDGEMENT

We have taken many efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. We would like to extend our sincere thanks to all of them.

We are highly indebted to **Prof. Nirali Borad** for their guidance and constant supervision as well as for providing necessary information regarding the Mini Project titled “**E-VEGETABLE MARKET MANAGEMENT**”. We would like to express our gratitude towards staff members of Computer Engineering Department, Atmiya University for their kind co- operation and encouragement which helped us in completion of this project.

We even thank and appreciate to our colleague in developing the project and people who have willingly helped us out with their abilities.

Prachi Ardesna (201002002)

Moxa Sodha(201002025)

ABSTRACT

The paper proposes to organize the vegetable market and bring about a change in the way things work. Vegetable, one of the products that is highly dependent on the rural section of the country has developed into such an efficient system that ensures that every household in the country gets fresh vegetable on-time with all due remunerate onto the producers. The paper proposes on interface for the consumers/buyers of vegetables. The main challenge for the system is to intervene starting with the current set up and incrementally bring the benefit of improved efficiency.

INDEX

Sr. No.	TITLES	Page No.
	Acknowledgement	I
	Abstract	II
	Index	III
	List of Figures	VI
1.	Introduction	1
	1.1 Introduction	1
	1.2 Purpose	1
	1.3 Scope	1
	1.4 Feasibility study	1
	1.4.1 Operational Feasibility	1
	1.4.2 Technical Feasibility	2
	1.4.3 Economical Feasibility	2
2.	Software Requirements Specification	3
	2.1 Hardware Requirement	3
	2.2 Software Requirement	3
3.	Design & Planning	4
	3.1 Software Development Life Cycle Model	4
	3.1.1 Waterfall Model	4
	3.2 DFD(data flow diagram)	4
	3.3 ER-Diagram	6
	3.4 Class Diagram	7
	3.5 Input / Output Interface	8
4.	Implementation Details	14

	4.1	Back End		14
		4.1.1	JAVA	14
		4.1.2	MYSQL	14
5	Testing and Implementation			15
	5.1	Unit Testing		15
		5.1.1	Introduction	15
		5.1.2	Benefits	15
	5.2	Integration Testing		16
		5.2.1	Purpose	16
			5.2.1.1 Top-down And Bottom-up	16
	5.3	Software Verification And Validation		17
		5.3.1	Introduction	17
		5.3.2	Classification Of Methods	17
	5.4	System Testing		17
6.	Limitations			18
7.	Conclusion			19
8.	References			20

LIST OF FIGURES

Figure name	Page no.
Data flow diagram	4
- DFD level 0	4
- DFD level 1	5
- DFD level 2	5
Use case diagram	6
ER-Diagram	7
Class diagram	8

INTRODUCTION

1.1 Introduction

The paper proposes to organize the vegetable market and bring about a change in the way things work. Vegetable, one of the products that is highly dependent on the rural section of the country has developed into such an efficient system that ensures that every household in the country gets fresh vegetable on-time with all due remuneration to the producers. The paper proposes on interface for the consumers/buyers of vegetables. The main challenge for the system is to intervene starting with the current set up and incrementally bring the benefit of improved efficiency.

1.2 Purpose

This document is meant to delineate the features of E-Vegetable market Management, so as to serve as a guide to the developers on one hand and a software validation document for the prospective client on the other. The E-Vegetable Market Management for Vegetable item shop Android application is intended to provide complete solutions for vendors as well as customers through a single get way using the internet. It will enable vendors to setup online selling, customer to browse through each registered vendor and purchase them online without having to visit the shop physically. The administration module will enable a system administrator to approve and reject requests for new vendors and maintain various lists of vegetable.

1.3 Scope

This system allows the customers to get their required product and can communicate directly to vendors and vice-e-versa.

1.4 Feasibility study

A feasibility study is a high-level capsule version of the entire System analysis and Design Process. The study begins by classifying the problem definition. Feasibility is to determine if it's worth doing. Once an acceptance problem definition has been generated, the analyst develops a logical model of the system. A search for alternatives is analyzed carefully. There are 3 parts in feasibility study.

- 1) Operational Feasibility
- 2) Technical Feasibility
- 3) Economical Feasibility

- **Operational Feasibility:**

- The System is to be developed for vendor and customers who wants to use it. We want our system user friendly and easy to use.

- The administrator also may be non-technical, so the user interface will be designed in such a way that it gets comfortable for non-technical person to operate easily.
- **Economic Feasibility:**
- Economic feasibility is a measure of cost effectiveness of a project or solution.
- For declaring that the system is economically feasible, the benefits from the project should exceed or at least to the equal to the cost of development.

- **Technical Feasibility:**
- This involves questions such as whether the technology needed for the system exists, how difficult it will be to build, and whether the firm has enough experience using that technology.
- The assessment is based on outline design of system requirements in terms of input, processes, output, fields, programs and procedures. This can be qualified in terms of volume of data, trends, frequency of updating in order to give an introduction to the technical system.

CHAPTER 2

SOFTWARE REQUIREMENTS SPECIFICATION

2.1 Hardware Requirements

- Minimum 2.27 Ghz Processor
- RAM: 4GB minimum
- 100GB free space in Hard Disk storage

2.2 Software Requirements

- Device must have browser
- Android studio
- Visual Studio
- Eclipse IDE

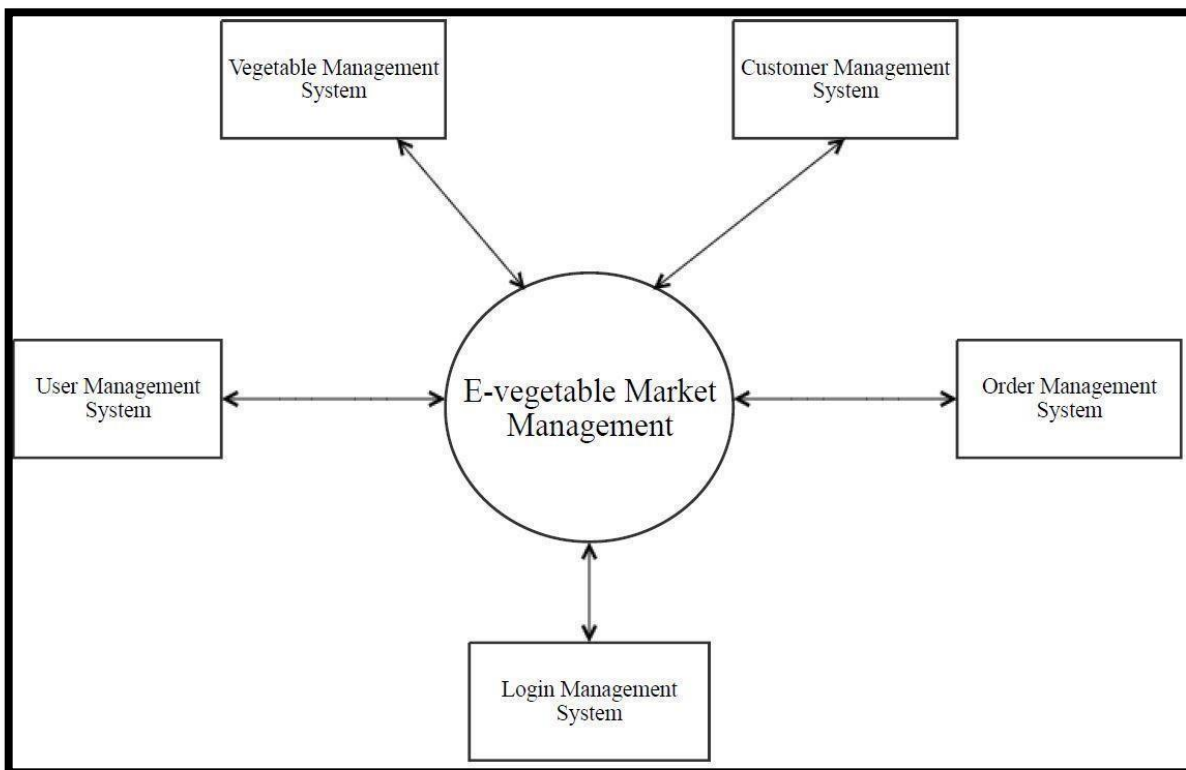
CHAPTER 3

DESIGNING & PLANNING

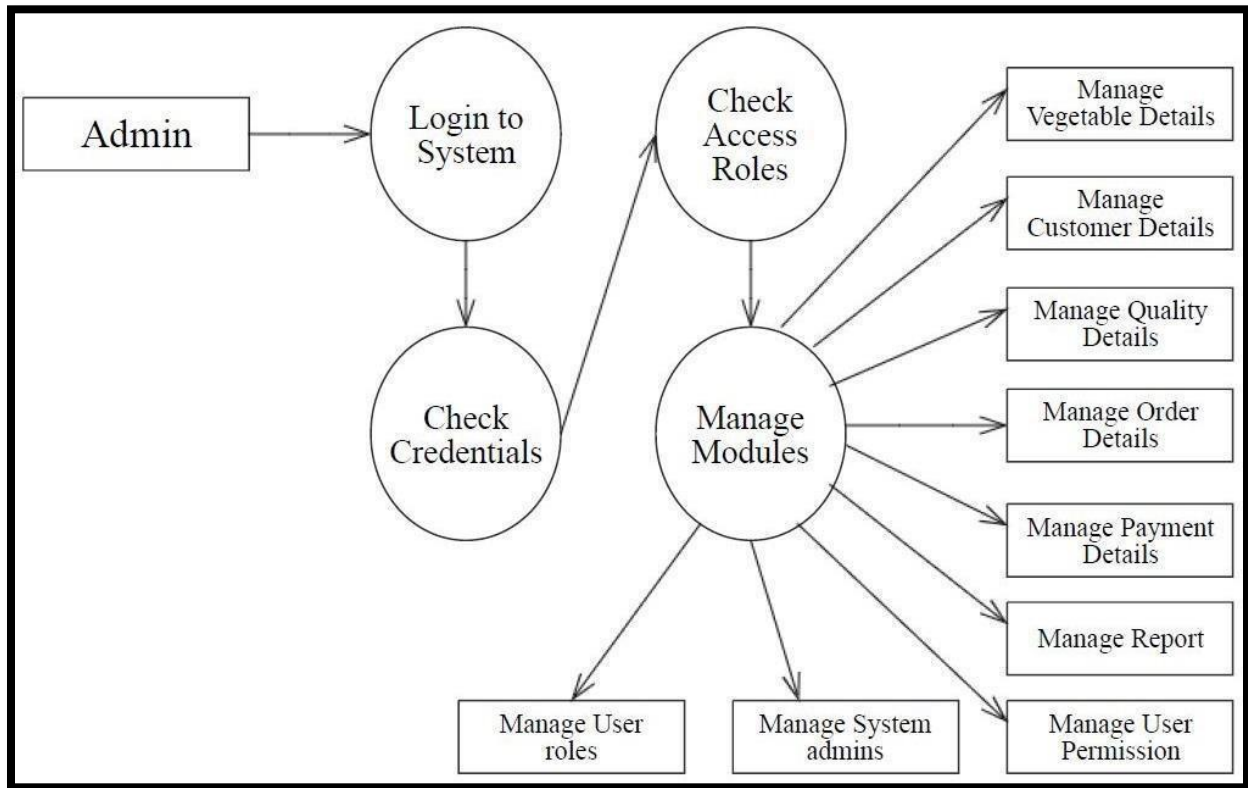
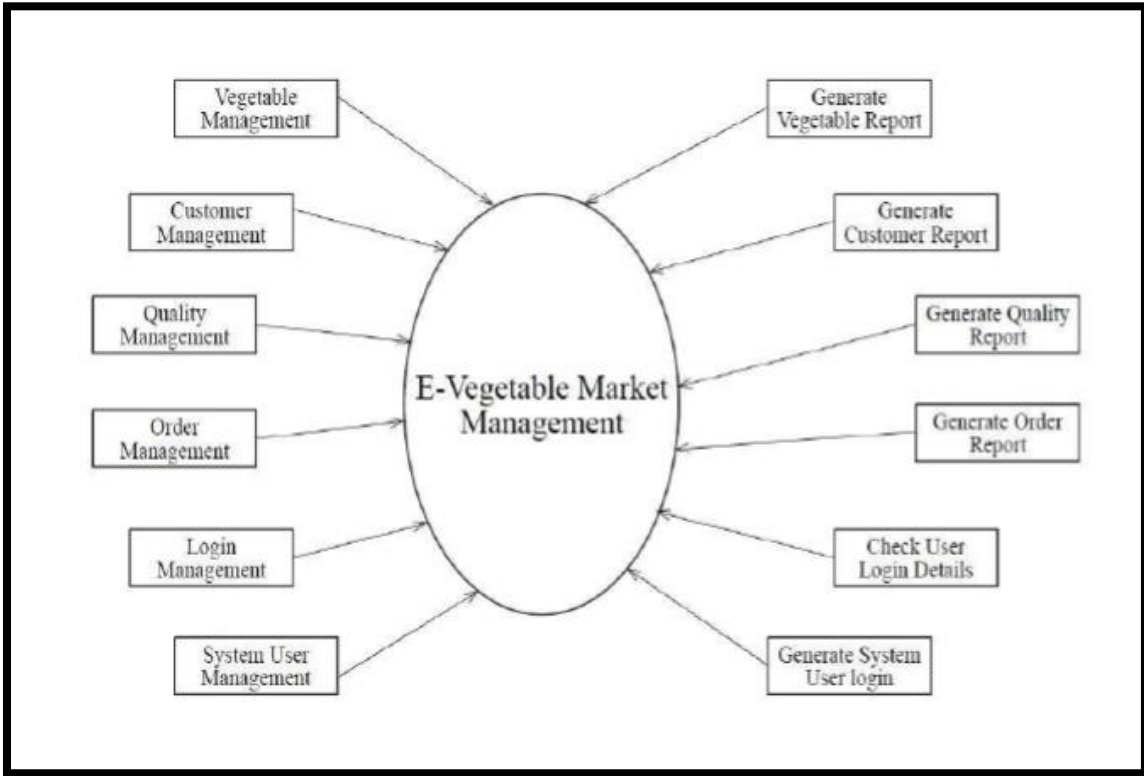
3.1 Software Development Life Cycle Model

3.1.1 Waterfall model

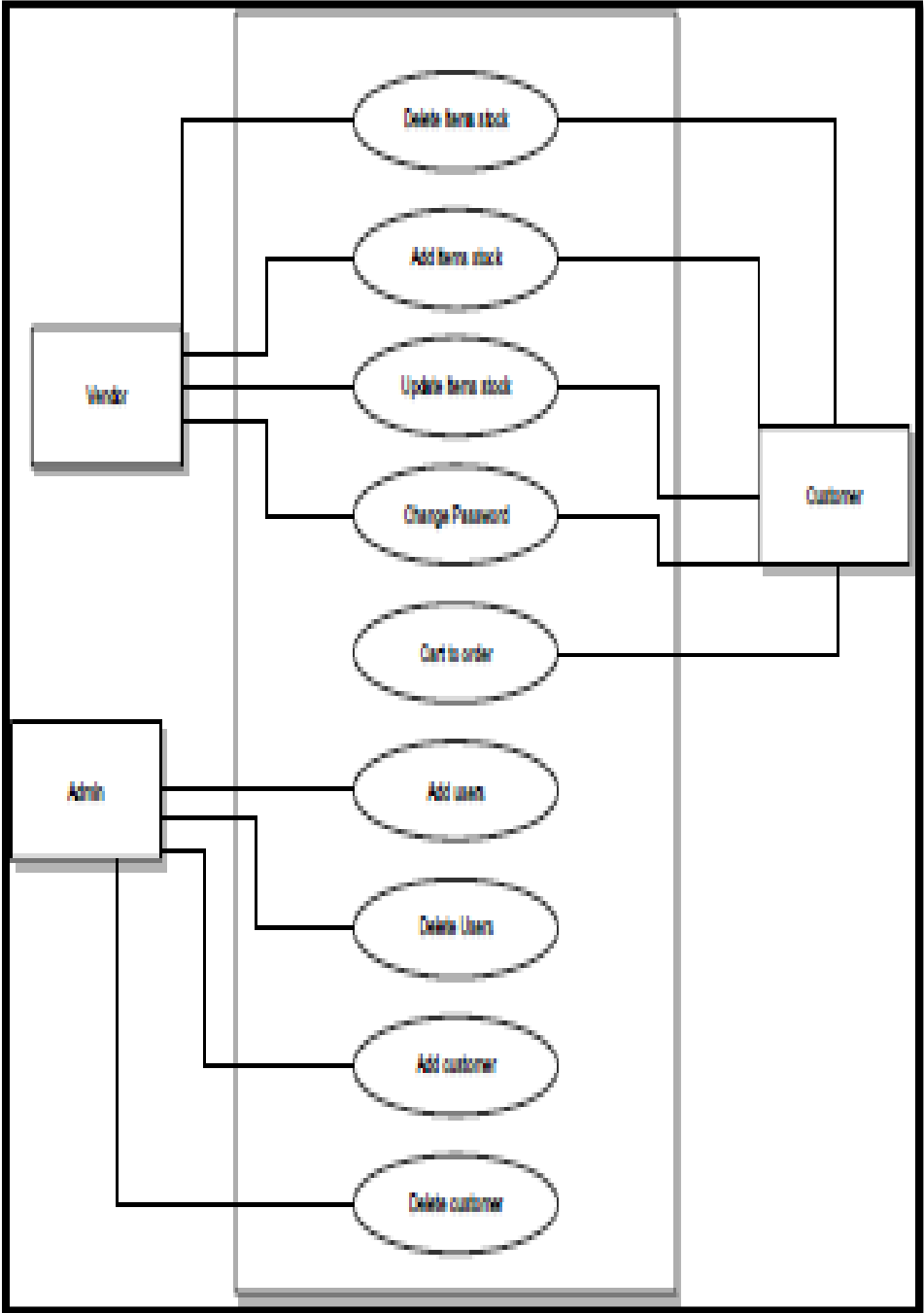
The waterfall model was selected as the SDLC model due to the following reasons: Requirements were very well documented, clear and fixed. Technology was adequately understood. Simple and easy to understand and use. There were no ambiguous requirements. Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process. Clearly defined stages. Well understood milestones easy to arrange tasks.



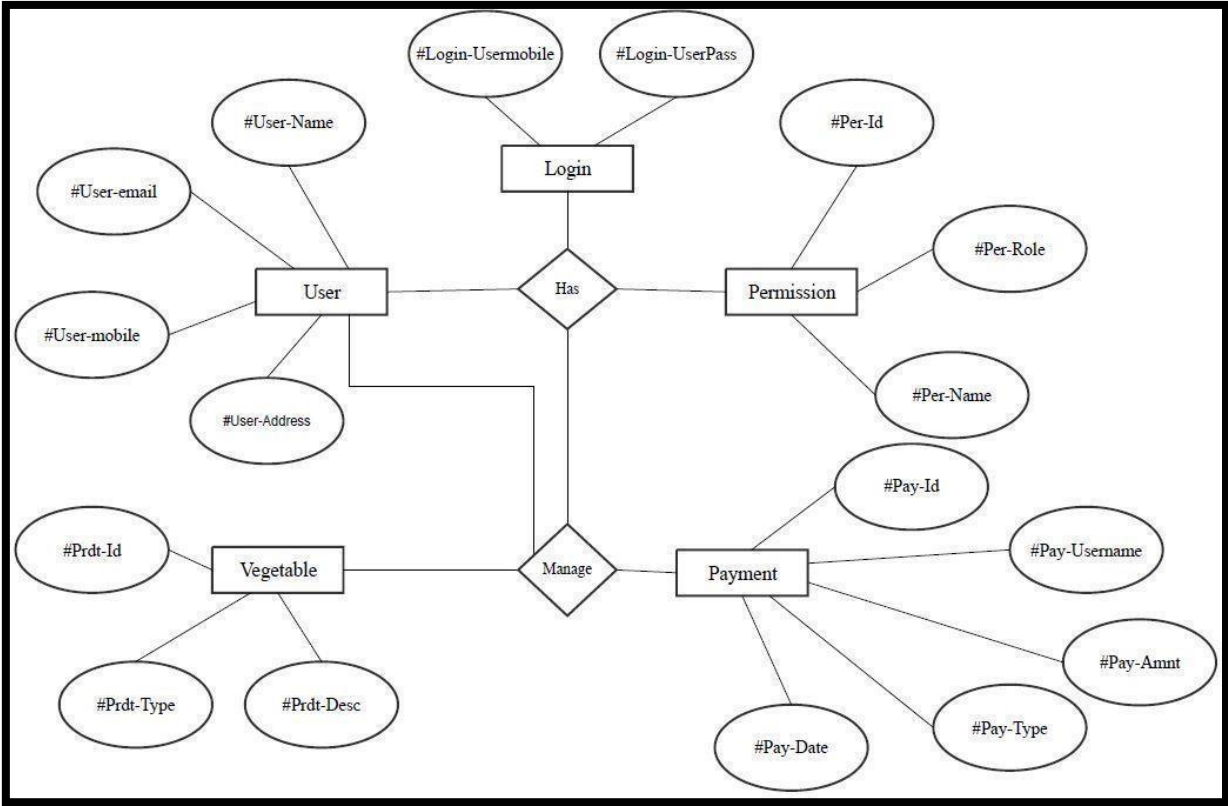
3.2 Data Flow Diagram



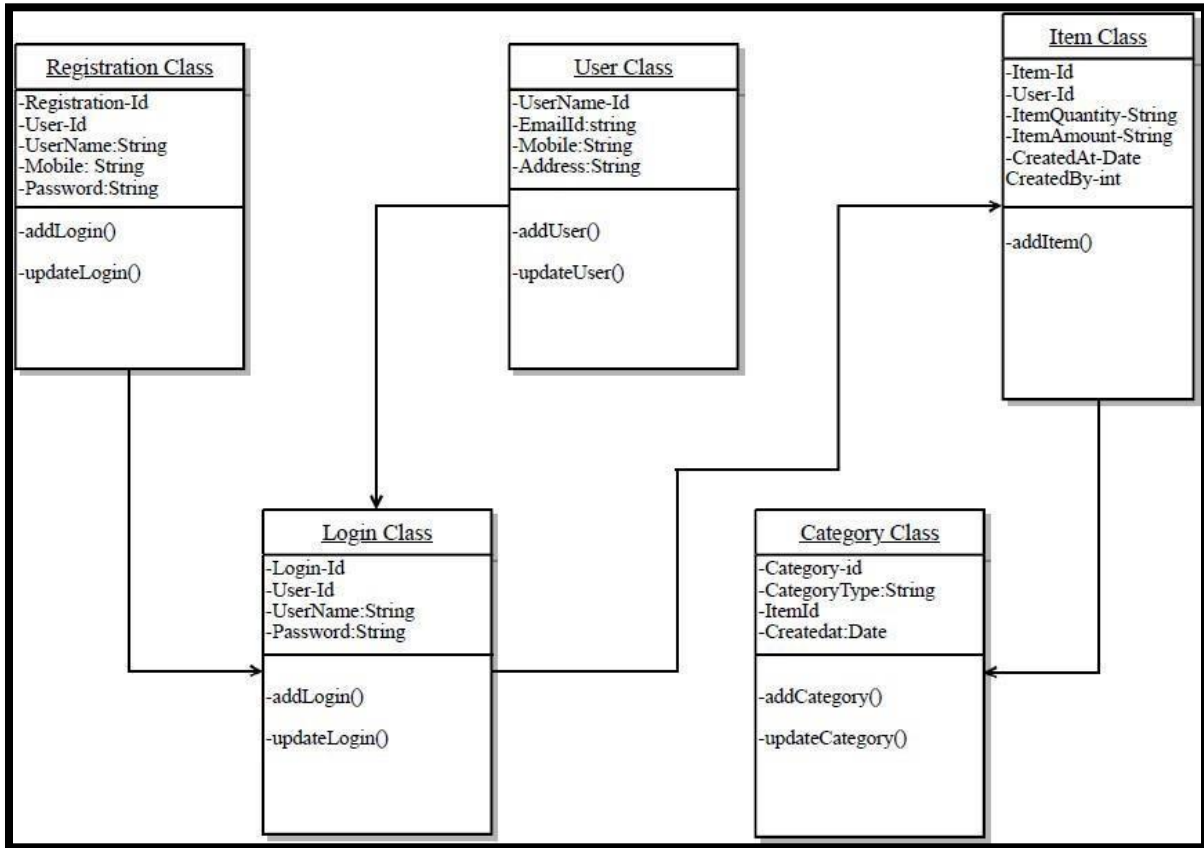
3.3 Use Case Diagram



3.4 ER-Diagram



3.5 Class Diagram



3.6 Input / Output Interface



CHOOSE AND CHECKOUT


Dummy text is text that is used in the publishing industry or by web designers to occupy the space



Let's Shopping

10:00 0.20 KB/S 54%


⊗



SIGN UP

Full Name

Mobile Number

Password 


By clicking SignUp you agree all the Terms and Conditions.

SIGN UP

Already have Account? Login here.


10:17 43.0 KB/S 56%

⊗



LOGIN

Mobile Number

Password 

Forgot password?

LOGIN

Not a Member yet? Sign Up here.


9:46 0.02 KB/S 53%

Home

₹75 paytm CASHBACK
Min order ₹1000 | Offer valid only on orders between 24 - 25 Mar

Fruites Cereal Vegetables


New Products



Spinach

- Rs. 30

Shop Now





Cabbage

1kg - Rs. 40

shop Now


Popular Products





9:49 3.00 KB/S 53%

Cabbage



Cabbage Rs.40

Free Shipping Est. delivery 5 days Max

Weight 1kg

Item description >


Returns Not accepted

Returns Money Back Guarantee >

Add to cart Share

9:49 1.00 KB/s 53%

← Cabbage 🔍 🛒



Cabbage Rs.40

Free Shipping Est. delivery 5 days Max

Weight 1kg

Item discription >

Returns Not accepted


Returns Money Back Guarantee >

- 4 +

9:49 0.31 KB/s 53%

← Cart 🗑️

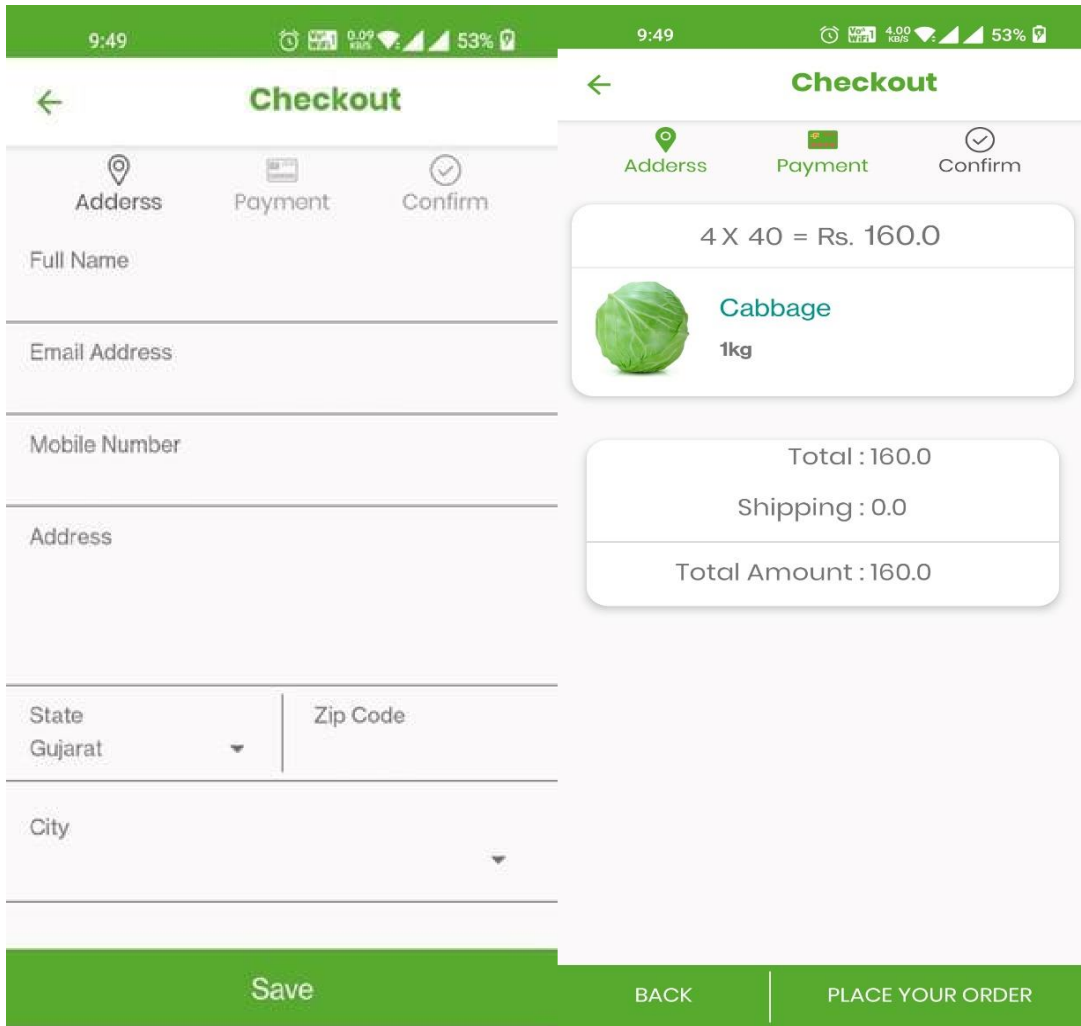
— 4 X 40 = Rs. 160.0 +

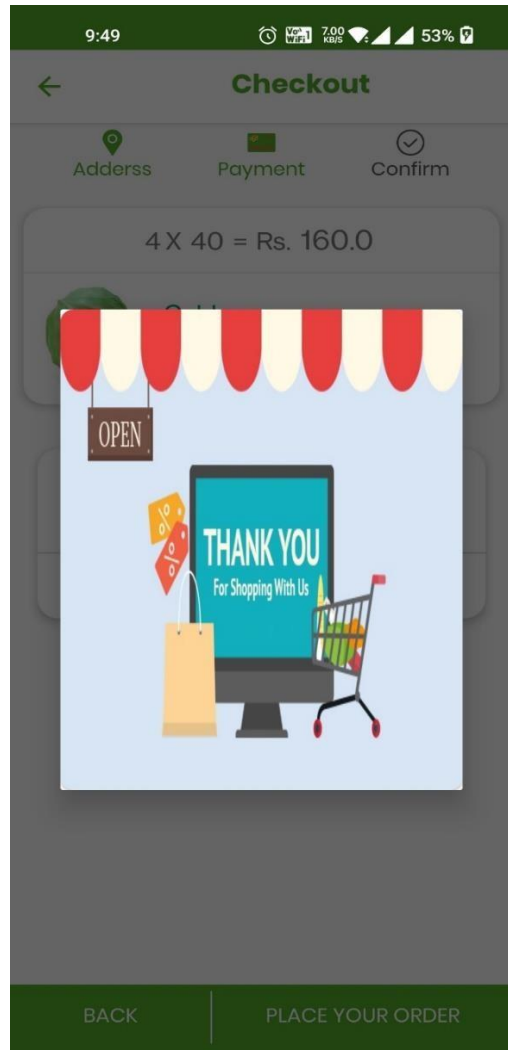
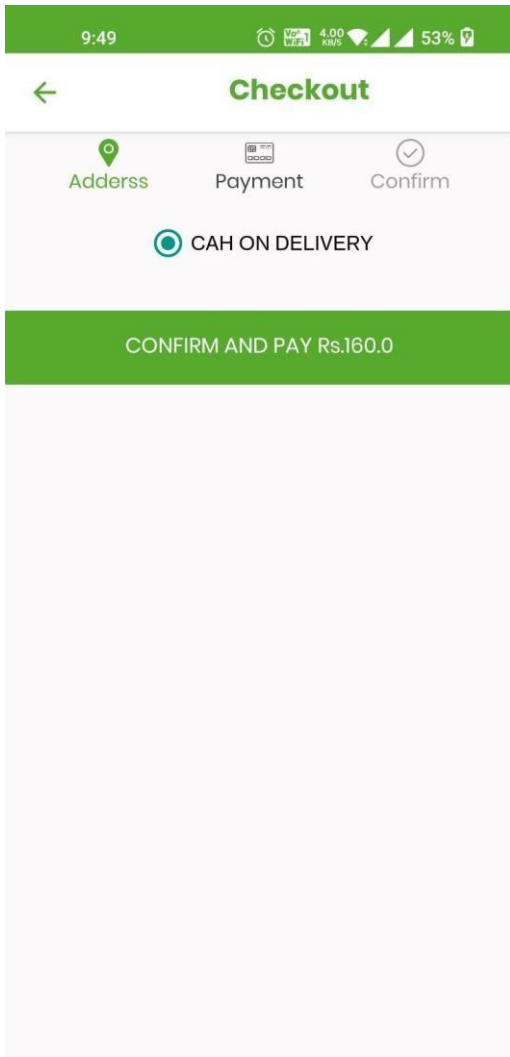


Cabbage 1kg ✕

Rs. 160.0

Checkout





CHAPTER 4

IMPLEMENTATION DETALIS

4.1 Back end

4.1.1 JAVA

Java is high level, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let programmers write once, run anywhere. Java applications are typically compiled to bytecode that can run on any java virtual machine (JVM) regardless of the underlying computer architecture. The java runtime provide dynamic capabilities that are not typically available in traditional compiled language.

4.1.2 MYSQL

MySQL is an open source relational database management system (RDBMS) based on Structured Query Language (SQL). It is one part of the very popular LAMP platform consisting of Linux, Apache, My SQL, and PHP. Currently My SQL is owned by Oracle. My SQL database is available on most important OS platforms. It runs on BSD Unix, Linux, Windows, or Mac OS. Wikipedia and YouTube use My SQL. These sites manage millions of queries each day. My SQL comes in two versions: My SQL server system and My SQL embedded system.

CHAPTER 5

TESTING AND IMPLEMENTATION

The term implementation has different meanings ranging from the conversion of a basic application to a complete replacement of a computer system. The procedures however, are virtually the same. Implementation includes all those activities that take place to convert from old system to new. The new system may be totally new replacing an existing manual or automated system or it may be major modification to an existing system. The method of implementation and time scale to be adopted is found out initially. Proper implementation is essential to provide a reliable system to meet organization requirement.

5.1 Unit testing

5.1.1 Introduction

In computer programming, unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming, a unit could be an entire module, but it is more commonly an individual function or procedure. In object-oriented programming, a unit is often an entire interface, such as a class, but could be an individual method.

5.1.2 Benefits

The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict, written contract that the piece of code must satisfy. As a result, it affords several benefits.

1) Find problems early : Unit testing finds problems early in the development cycle. In test-driven development (TDD), which is frequently used in both extreme programming and scrum, unit tests are created before the code itself is written. When the tests pass, that code is considered complete. The same unit tests are run against that function frequently as the larger code base is developed either as the code is changed or via an automated process with the build.

2) Facilitates Change : Unit testing allows the programmer to refactor code or upgrade system libraries at a later date, and make sure the module still works correctly (e.g., in regression testing). The procedure is to write test cases for all functions and methods so that whenever a change causes a fault, it can be quickly identified.

3) Simplifies Integration : Unit testing may reduce uncertainty in the units themselves and can be used in a bottom-up testing style approach. By testing the parts of a program first and then testing the sum of its parts, integration testing becomes much easier.

4) Documentation : Unit testing provides a sort of living documentation of the system. Developers looking to learn what functionality is provided by a unit, and how to use it, can look at the unit tests to gain a basic understanding of the unit's interface (API). Unit test cases embody characteristics that are critical to the success of the unit.

5.2 INTEGRATION TESTING

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

5.2.1 Purpose

The purpose of integration testing is to verify functional, performance, and reliability requirements placed on major design items. These "design items", i.e., assemblages (or groups of units), are exercised through their interfaces using black-box testing, success and error cases being simulated via appropriate parameter and data inputs. Simulated usage of shared data areas and inter-process communication is tested and individual subsystems are exercised through their input interface. Test cases are constructed to test whether all the components within assemblages interact correctly, for example across procedure calls or process activations, and this is done after testing individual modules, i.e., unit testing. The overall idea is a "building block" approach, in which verified assemblages are added to a verified base which is then used to support the integration testing of further assemblages. Software integration testing is performed according to the software development life cycle (SDLC) after module and functional tests.

5.2.1.1 Top-down And Bottom-up

Bottom-up testing is an approach to integrated testing where the lowest level components are tested first, then used to facilitate the testing of higher level components. The process is repeated until the component at the top of the hierarchy is tested. All the bottom or low-level modules, procedures or functions are integrated and then tested. After the integration testing of lower level integrated modules, the next level of modules will be formed and can be used for integration testing. This approach is helpful only when all or most of the modules of the same development level are ready. This method also helps to determine the levels of software developed and makes it easier to report testing progress in the form of a percentage.

5.3 SOFTWARE VERIFICATION AND VALIDATION

5.3.1 Introduction

In software project management, software testing, and software engineering, verification and validation (V&V) is the process of checking that a software system meets specifications and that it fulfills its intended purpose. It may also be referred to as software quality control. It is normally the responsibility of software testers as part of the software development lifecycle. Validation checks that the product design satisfies or fits the intended use (high-level checking), i.e., the software meets the user requirements. This is done through dynamic testing and other forms of review. Verification and validation are not the same thing, although they are often confused. Boehm succinctly expressed the difference between

5.3.2 Classification of Methods

In mission-critical software systems, where flawless performance is absolutely necessary, formal methods may be used to ensure the correct operation of a system. However, often for non-mission-critical software systems, formal methods prove to be very costly and an alternative method of software V&V must be sought out. In such cases, syntactic methods are often used.

5.3.3 Test Cases

A test case is a tool used in the process. Test cases may be prepared for software verification and software validation to determine if the product was built according to the requirements of the user. Other methods, such as reviews, may be used early in the life cycle to provide for software validation.

5.4 System Testing

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black-box testing, and as such, should require no knowledge of the inner design of the code or logic. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.

System testing is performed on the entire system in the context of a Functional Requirement Specification(s) (FRS) and/or a System Requirement Specification (SRS). System testing tests not only the design, but also the behavior and even the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software/hardware requirements specification(s).

CHAPTER 6

LIMITATIONS

- Online payment not available
- User can buy the vegetable which are available in stock
- User have to login with only mobile number

CHAPTER 7

CONCLUSION

With regard to the requirements specified, we completed the project. This system satisfies the users and it is a user– friendly application which is easy to operate. The hypothesis was that E-veg would last the longest in all of the devices tested. My results do support my hypothesis. The proposed system in which we took the idea that will make every farmer reach the homes in there nearby locality or cities by the medium of this web application. In this we have used some simple database. Finally, we achieve the farmer profit to directly connected to the end user. There are some trends that indicate the transformation of agricultural information systems in India is occurring. This application provides availability of rates in various vegs help to give good rates to farmers. Transportation losses reduced after e-agriculture marketing. This is important for the transformation of agriculture in India

CHAPTER 8

REFERENCE

- <https://www.w3schools.com/>
- <https://www.geekforgeeks.org>
- <https://javatpoint.com>
- <https://developer.android.com>