

INDUSTRY INTERNSHIP PROJECT

(Project Term Jun-April 2020)

(Plant Tracking System)

Submitted by

**Amit Kr. Sanu
V. Raja Reddy
Ganugula Tarun
Sudhir Kr. Sah
Ridham Pundir
Shubham Singh**

**Registration Number :11616194
Registration Number :11610085
Registration Number :11604996
Registration Number :11617718
Registration Number :11812787
Registration Number :11606138**

Project Group Number: 7

Course Code: CSE441: INDUSTRY INTERNSHIP PROJECT

Under the Guidance of

Mr. Yadwinder Singh, Assistant Professor

LOVELY INFOTECH



**L OVELY
P ROFESSIONAL
U NIVERSITY**

DECLARATION

We hereby declare that the project work entitled (“Plant Tracking System”) is an authentic record of our own work carried out as requirements of Industry Internship Project for the award in Web developer from Lovely infotech under the guidance of Mr. Yadwinder Singh during January to April 2020. All the information furnished in this industry internship project report is based on our own intensive work and is genuine.

Project Group Number: 7

Name of Student: Amit Kr. Sanu

Registration Number: 1616194

Name of Student: Shubham Singh

Registration Number: 11606138

Name of Student: Ganugula Tarun

Registration Number: 11604996

Name of Student: Sudhir Kr. Sah

Registration Number: 11617718

Name of Student: V. Raja Reddy

Registration Number: 11610085

Name of Student: Ridham Pundir

Registration Number: 11812787

(Signature of Student 1)

Date:

(Signature of Student 2)

Date:

(Signature of Student 3)

Date:

(Signature of Student 4)

Date:

(Signature of Student 5)

Date:

(Signature of Student 6)

Date:

CERTIFICATE

This is to certify that the declaration statement made by this group of students is correct to the best of my knowledge and belief. They have completed this Industry Internship Project under my guidance and supervision. The present work is the result of their original investigation, effort and study. No part of the work has ever been submitted for any other developer at any Organization. The Industry Internship Project is fit for the submission and partial fulfillment of the conditions for the award in Web developer from Lovely Infotech.

Signature and Name of the Mentor

Designation

Web developer,
Lovely Infotech.

Date:

ACKNOWLEDGEMENT

We take this opportunity to present our votes of thanks to all those guideposts who really acted as lightening pillars to enlighten my way throughout this project that has led to successful and satisfactory completion of this study.

We are grateful to Yadwinder Sir for providing us with an opportunity to undertake this project and providing us with all the facilities. I am highly thankful to Sir for his active support, valuable time and advice, whole- hearted guidance, sincere cooperation and pains- taking involvement during the study and in completing the assignment of preparing the said project within the time stipulated.

Lastly, we are thankful to all those, particularly the various friends, who have been instrumental in creating proper, healthy and conductive environment and including new and fresh innovative ideas for me during the project, without their help, it would have been extremely difficult for us to prepare the project in a time bound framework.

TABLE OF CONTENTS

Inner first page	(i)
Declaration	(ii)
Certificate.....	(iii)
Acknowledgement	(iv)
Table of Contents.....	(v)
1. Introduction.....	1
2. Profile of the Problem. Rationale/Scope of the study	3
3. Existing System	4
• Introduction	
• Existing Software	
• DFD for present system	
• What's new in the system to be developed	
4. Problem Analysis	8
• Product definition	
• Feasibility Analysis	
• Project Plan	
5. Software Requirement Analysis	12
• Introduction	
• General Description	
• Specific Requirements	
6. Design	14
• System Design	
• Design Notations	
• Detailed Design	
• Flowcharts	
• Pseudo code	
7. Testing.....	23
• The Testing Triangle	

• Unit Testing	
• Database Integration Testing	
• Manual Testing	
8. Implementation	26
• Implementation of the project	
• Post-Implementation and Software Maintenance	
9. User Manual: A complete document (Help Guide)	
of the software developed	29
10. Source Code (where ever applicable)	37
11. Bibliography	38

1. Introduction

The main objective of the “Plant Tracking System” is to develop a website that Scan QR Code of the plants and find the details of the plants. The position of plants is calculated according to the area and symptoms entered by user in the website. With the help of this Website, a user can find the plant detail according to Scanner availability. A Plant details will be available in the website including, Plant Search, advanced plant search. The Plant Tracking Service can be entered using a username and password. It is accessible either by an administrator or user. Only they can add data into the database. The data can be retrieved easily. The interface is very user-friendly. The data are well protected for personal use and makes the data processing very fast. Plant Tracking System is powerful, flexible, and easy to use and is designed and developed to deliver real conceivable benefits to Peoples. Plant Tracking System is designed for the Type of plants, to cover a Plants administration and management processes. Plant Tracking is a software product suite designed to improve the quality and management of plants.

Introduction about different technology used -

A) HTML: Hypertext Markup Language (HTML) is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript. Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

B) CSS: Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language like HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript. CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .CSS file, and reduce complexity and repetition in the structural content. A separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or

screen reader), and on Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device.

C) JavaScript: JavaScript (often abbreviated as JS), is a high-level, interpreted scripting language that conforms to the ECMAScript specification. JavaScript has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first- class functions.

Alongside HTML and CSS, JavaScript is one of the core technologies of the World Wide Web. JavaScript enables interactive web pages and is an essential part of web applications. The vast majority of websites use it, and major web browsers have a dedicated JavaScript engine to execute it.

D) SQL Server: SQL Server is a relational database management system (RDBMS) developed by Microsoft. It is primarily designed and developed to compete with MySQL and Oracle database. SQL Server supports ANSI SQL, which is the standard SQL (Structured Query Language) language.

E) BOOTSTRAP: Build responsive, mobile-first projects on the web with the world's most popular front-end component library. Bootstrap is an open source toolkit for developing with HTML, CSS, and JS. Quickly prototype your ideas or build your entire app with our Sass variables and mixing, responsive grid system, extensive prebuilt components, and powerful plugins built on jQuery.

F) ASP.Net MVC: ASP.NET supports three major development models: Web Pages, Web Forms and MVC (Model View Controller). ASP.NET MVC framework is a lightweight, highly testable presentation framework that is integrated with the existing ASP.NET features, such as master pages, authentication, etc. Within .NET, this framework is defined in the System. Web. MVC assembly. The latest version of the MVC Framework is 5.0. We use Visual Studio to create ASP.NET MVC applications which can be added as a template in Visual Studio.

2. Profile of the Problem. Rationale/Scope of the study (Problem Statement)

This project is a simple study regarding website this project can be used by everyone in the city, university. It is using Html, CSS, JavaScript, Bootstrap, Sql Server, MVC and various other technology. We do need to do the entire task from the beginning as we all do not had any kind of knowledge regarding the web development. We all belonged to different majors and minors. We all wanted to learn something new for ourselves and for the better understanding of web development. For us, creating the new website from the scratch is the biggest challenge as all of us are not having professional level skills in the web development and the languages used in the project. We all decided to start reading about the technologies which we had some basic idea that can be helpful in creating the project. We downloaded online courses, sought help from friends who had some knowledge in website development, sought help from everyone's favorite google and did everything possible to complete the project. We wanted to do a lot of things but were unable to do because of some circumstances. But what we did was our best we could do.

Problem Description:

Lack of immediate retrievals: The information is very difficult to retrieve and to find particular information like- To find out about the plant history, the user has to go through various registers. This results in in convenience and wastage of time.

Lack of immediate information storage: The information generated by various transactions takes time and efforts to be stored at right place.

Lack of prompt updating: Various changes to information like plant details.

Error prone manual calculation: Manual calculations are error prone and take a lot of time this may result in incorrect information. For example, nearby plants.

Scope of the Project:

At a finer level of detail, deploying the system consists of executing all steps necessary to educate the Consumers on the use of the new system, placing the newly developed system into production, confirming that all data required at the start of operations is available and accurate, and validating that business functions that interact with the system are functioning properly. Transitioning the system support responsibilities involves changing from a system development to a system support and maintenance mode of operation, with ownership of the new system moving from the Project Team to the Performing Organization. A key difference between system implementation and all other phases of the lifecycle is that all project activities up to this point have been performed in safe, protected, and secure environments, where project issues that arise have little or no impact on day-to-day business operations. Once the system goes live, however, this is no longer the case. It is through the careful planning, execution, and management of system Implementation activities that the Project team can minimize the likelihood of these occurrences and determine appropriate contingency plans in the event of a problem.

3.Existing System

3.1 Introduction and existing software:

Plants currently use a manual system for the management and maintenance of critical information. The current system requires numerous paper forms, with data stores spread throughout the plant management infrastructure. Often information is incomplete or does not follow management standards. Forms are often lost in transit between departments requiring a comprehensive auditing process to ensure that no vital information is lost. Multiple copies of the same information exist in the plant and may lead to inconsistencies in data in various data stores. Plant Tracking System is going to be helpful in manage all kinds of information about plant and trees in particular campus. It will be helpful to know the counts of plants, description of particular plant and also after implementing this each plant is going to have own unique ID. This System particularly going to help the student and faculties of Agriculture Department in field lab to know the basic information regarding particular species of plants. It is also going

to facilitate the curious visitors or students who wants to earn more information regarding campus plant species so it would easy through this system.

3.2 Existing Software:

The speed of innovation in the web development space is accelerating every year. Some various software, tools and methods are available to develop a website like:

- i) Ruby On Rail
- ii) Angular
- iii) ReactJS
- iv) Vue.js
- v) Meteor
- vi) PHP

So, these are the various frameworks used for web development but there are many pros and cons related to their functionality.

Here are few problems related to the frameworks:

- RubyOnRail falls short on runtime speed and performance. One of the most frequent argument against RoR is its ‘slow’ runtime speed.
- Lack of Flexibility
- High Cost of wrong decisions in development.
- Reactivity complexity
- Lack of support for larger-scale projects
- Language barrier
- Limited resources

As per above detail we discussed about the existing software for web development and now we will share some details about the existing websites based on our idea.

Existing Websites:

- i) Indiaplants.com
- ii) Plantinfo.co.za
- iii) Garden.org
- iv) Theseedsite.co.uk

3.3 DFD For Present System

Context Level DFD:

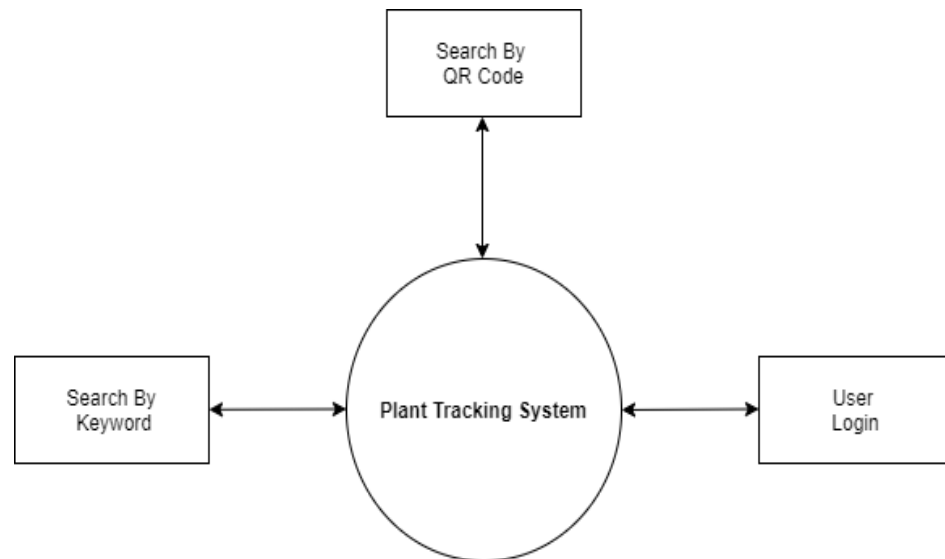


Fig.1 Zero Level DFD

Level 1 DFD

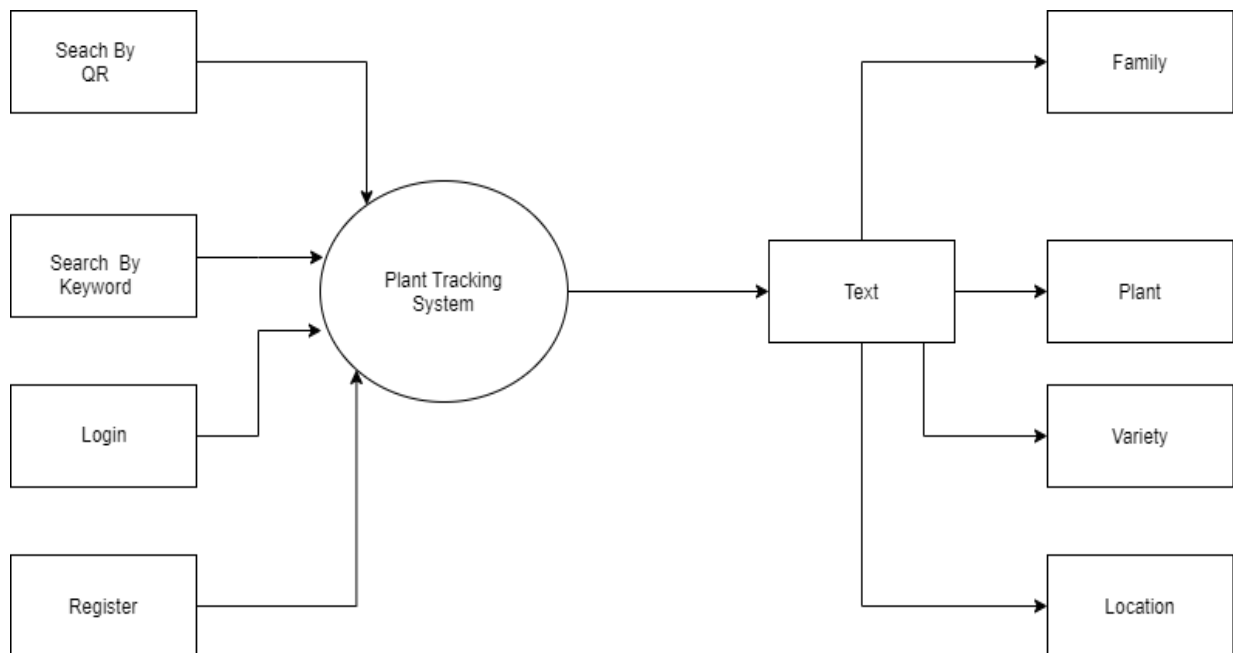


Fig.1.2 First Level DFD

Admin Level Data Flow Diagram:

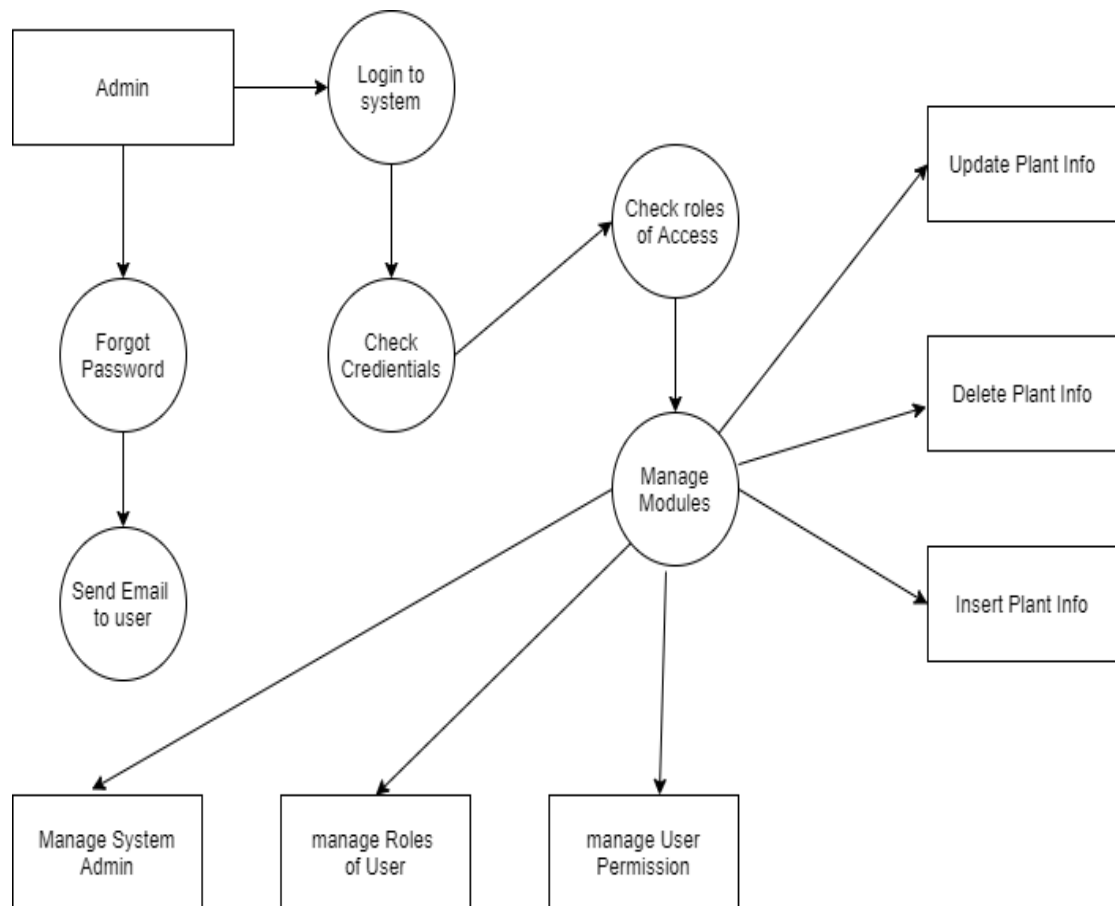


Fig. 1.3 Admin Level DFD

3.4 What's new in the system to be developed

Present system has extreme exposure with varieties of scopes so there are lot many features could be added in future. Presently we tried to cover basic facility of plant tracking system like location, unique id, biological details, age, Date of planting and etc. So, new in the system to be developed are:

- i) Humidity Tracking
- ii) Soil Health Tracking
- iii) Irrigation System
- iv) Pollination Tracking
- v) Mutation Tracking

4. Problem Analysis

4.1 Product Definition

Plant Tracking System is a web application which is developed using ASP.NET MVC framework. It is info-based plant tracking website which gives all the information regarding plant and its location, unique Id, age, date of planting and etc. This unique website facilitates basically curious person who wants earn more facts and information regarding local area plant. In this project we used unique way of searching details about plants by QR Scanner. We are assigning unique QR Code with unique id to each plant available in campus, if a person who want know about any particular plant then simply, he has to take out his phone and scan the QR code attached in plant and then link encoded in QR that will redirect him to plant detail page of our website. This website has all detailed information regarding that plant. User can search through search key also by visiting website. This project is focused on tracking the plants available in particular area. This can also be used to track, maintain the counts and data preparation of plant available in campus.

4.2 Feasibility Study

4.2.1 Technical Feasibility

Being a web application PTS will have an associated hosting cost. Since the system doesn't consist of any multimedia data transfer, bandwidth required for the operation of this application is very low.

The System will follow the freeware software standards. No cost will be changed from the potential customers. Bug fixes and maintaining tasks will having an associated negligible cost. At the initial stage the potential market space will be the local universities, higher educational institutes and plant nurseries.

Beside the associated cost, there will be many benefits for the customers. Especially the extra effort that is associated with detail feeding and feeding will be significantly reduced while the effort to create descriptive page will be eliminated, since detail page and QR code generation is fully automated.

From these it's clear that the project PTS is financially feasible.

4.2.3 Technical Feasibility

Project PTS is a complete web-based application. The main technologies and tools that are associated with PTS are

- HTML
 - CSS
 - JavaScript
 - jQuery
 - Bootstrap
 - MS-SQL
 - C#
 - ASP.NET MVC
 - Visual Studio
 - GitHub
 - Azure
 - Diagram drawing tools – Draw.IO
-
- Each of the technologies are freely available and the technical skill required are manageable. Time limitations of product development and the ease of implementing using these technologies are synchronized.

Initially the web site hosted on azure web hosting space, but for later implementations it will be hosted in other paid web hosting space with a sufficient bandwidth. Bandwidth required in this application is very low, since it doesn't incorporate any multimedia aspect.

From these it's clear that project PTS is technically feasible.

4.2.4 Resource and Time Feasibility

Resources that are required for the PTS project includes,

- Programing device (Laptop)
- Hosting Space (freely available)
- Programming tools (freely available)
- Programming individuals

So, it's clear that the project PTS has the required resource feasibility.

4.2.5 Risk Feasibility

Risk feasibility can be discussed under several contexts.

Risk associated with size

Estimated size of the in line of codes:

Being a web application with many numbers of stakeholder, PTS will contain significant amount of codes lines. As the system doesn't contain any multimedia aspect except images and QR code, the file sizes and the complete project size will not exceed 200MB.

Estimated size of product in number of programs:

Through the application supports many stakeholders, it will be constructed as a single login page rather than having the contest will be showed or hidden.

Size of database created or used by the product:

Database size will not exceed the values supported by MS-SQL. Number of relations and entities are minimized by using best practices of normalization theories.

User of the product:

- Students
- Lectures
- Teaching assistants
- Plant Nurseries
- Gardeners
- Sanctuaries
- Forest Department

Technology risks

Is the technology to be built new?

All the technologies are very well established and old enough also but not obsolete.

4.3 Project Plan

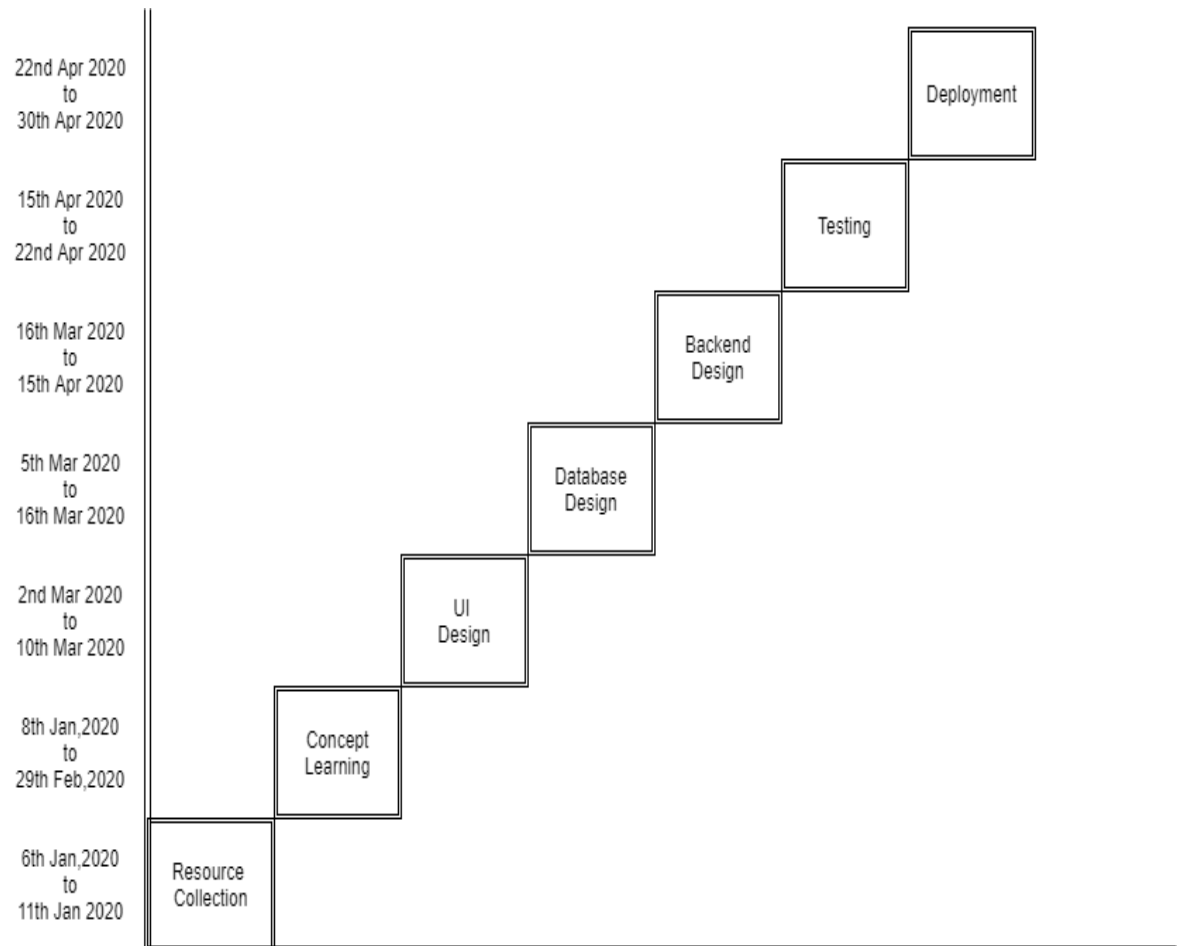


Fig.2 Phase View of Project Plan

5. Software Requirement Analysis

5.1 General Description

Web development broadly refers to the tasks associated with developing websites for hosting via intranet or internet. The web development process includes web design, web content development, client-side/server-side scripting and network security configuration, among other tasks. As a whole, ASP.NET is a great framework to use when developing web sites and web applications. It is reliable, fast, easy to use, free and widely known. ASP.NET gives you full control of your development and can be used on any project, big or small.

5.2 Specific Requirements

Performance:

PTS requires a very low bandwidth; hence the performance will not degrade with increasing number of potential users. At the development stage, a free hosting service will be used; But when installing the system to real university environment, it will be hosted in a much more reliable server to increase the performance.

MS-SQL will provide the adequate speed for database transactions. Since no big data analysis is done. MS-SQL is the ideal database for this project.

Response time: less than 2 seconds

Processing time: Less than 2 seconds

Query and reporting times: yet to be tested

Throughput: yet to be tested

Storage: yet to be tested

Security:

Security measures are provided in many aspects in this system.

User authentication:

User will have to authenticate using the username and password.

Depending on the access level each user will gain functionality of the system. Password can be changed by the user.

Login details:

Each user's login time and logout time will be recorded in the system, to make the tractability process easy in case of a faculty action.

Usability and ease of use:

The interfaces are designed to make it easy for any potential user to get familiar with the system within a minute. No additional training is required to use the system.

Availability:

System will be available throughout the 24 hours. Mean time to failure and mean time to repair will be decided to increase the availability. With a paid hosting space, the availability can be guaranteed to be great precision.

Maintainability:

PTS is designed using the best practices of MVC and C#. Since every single segment in the system is very well structured, the system is highly maintainable. PTS database is designed on Microsoft SQL server which is considered as best option for database design. It is most reliable platform for designing and maintaining the database. So, Product can be maintained in order to correct errors, fix bugs and add additional features. Trade-offs decisions will be made on the maintainability of the Technology.

Scalability

Technology tradeoffs are made based on the technologies that are more scalable and able to handle increase loads efficiently without a break in the system efficiency.

MVC Design Pattern

The Model View Controller (MVC) design pattern specifies that an application consist of a data model, presentation information, and control information. MVC mostly relates to the user Interface/interaction layer of an application.

In the MVC pattern the user sees the View which is updated by the model which is turn manipulated by the Controller.

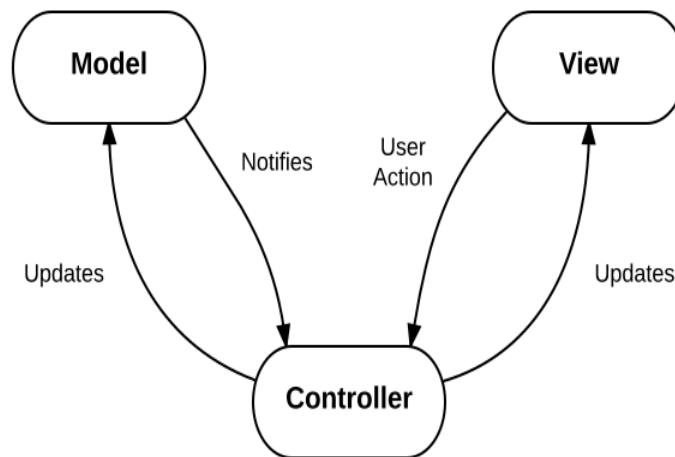


Fig. 4 MVC Architecture

- The **Model** contains only the pure application data, it contains no logic describing how to present the data to a user. They are the parts of the application that implement the logic for the application's data domain. They retrieve and store model state in a database.
- The **View** presents the model's data to the user. The view can only be used to access the model's data. They are the components that display the application's user interface (UI).
- The **Controller** exists between the view and the model. It listens to events triggered by the view and executes the appropriate commands. They are the components that handle user interaction, work with the model, and ultimately select a view to render that displays UI.

Advantages of the MVC design pattern

- Multiple developers can work simultaneously on the model, controller and views.

- MVC enables logical grouping of related actions on a controller together. The views for a specific model are also grouped together.
- Low coupling — The very nature of the MVC framework is such that there is low coupling among models, views or controllers.
- Models can have multiple views.
- Ease of modification — Because of the separation of responsibilities, future development or modification is easier

Disadvantages

- Knowledge on multiple technologies becomes the norm. Developers using MVC need to be skilled in multiple technologies.
- Must have strict rules on methods
- Cannot see design page preview like .aspx page. Every time want to run then see the design.
- Understanding flow of application is very hard one. It is little bit of complex to implement and not suitable for small level applications.
- Its deployment is little bit hard one.

Explanation

User will make request for the page which user would like to retrieve. Requested page will go to controller and on controller Route.Config will be checked. Requested page will get transfer to Model from Controller.

On Model there will be 2 steps,

1. Page initialization will get started.
2. Then result set will be generated.

After these operation result set will get unload to View through view engine.

Responsibility of View Engine

1. View Engine get the request from Controller
2. The requested page will get executed

3. The result got by the execution process will get deliver to View.

Life Cycle of MVC System

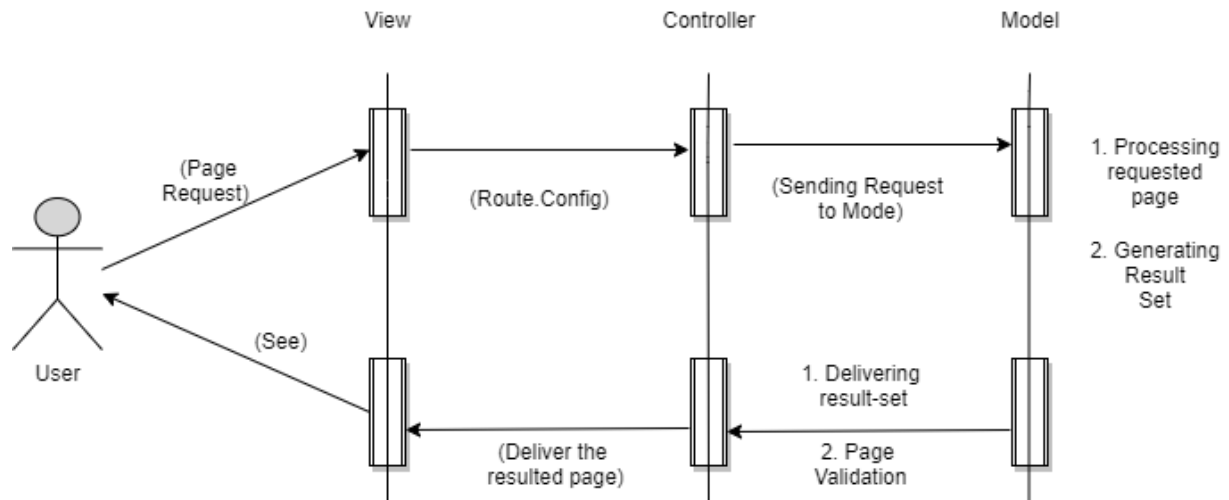


Fig. 5 Life Cycle Of MVC

Steps of MVC System

Step 1: User will make request for page

Step 2: Route.Config will get check on Controller

Step 3: After Route.Config validation request will get transfer from Controller to Model

Step 4: Page request will go to Model & on Model the following operation will perform.

- a. Page initialization will start and after initialization of page
- b. One result set will get generated

Step 5: Generated result set will get deliver to Controller from Model,

- a. On Model Page validation operation will perform.

Step 6: After page validation on controller, Page will deliver to View through View engine.

Step 7: Now user can see requested page as response page.

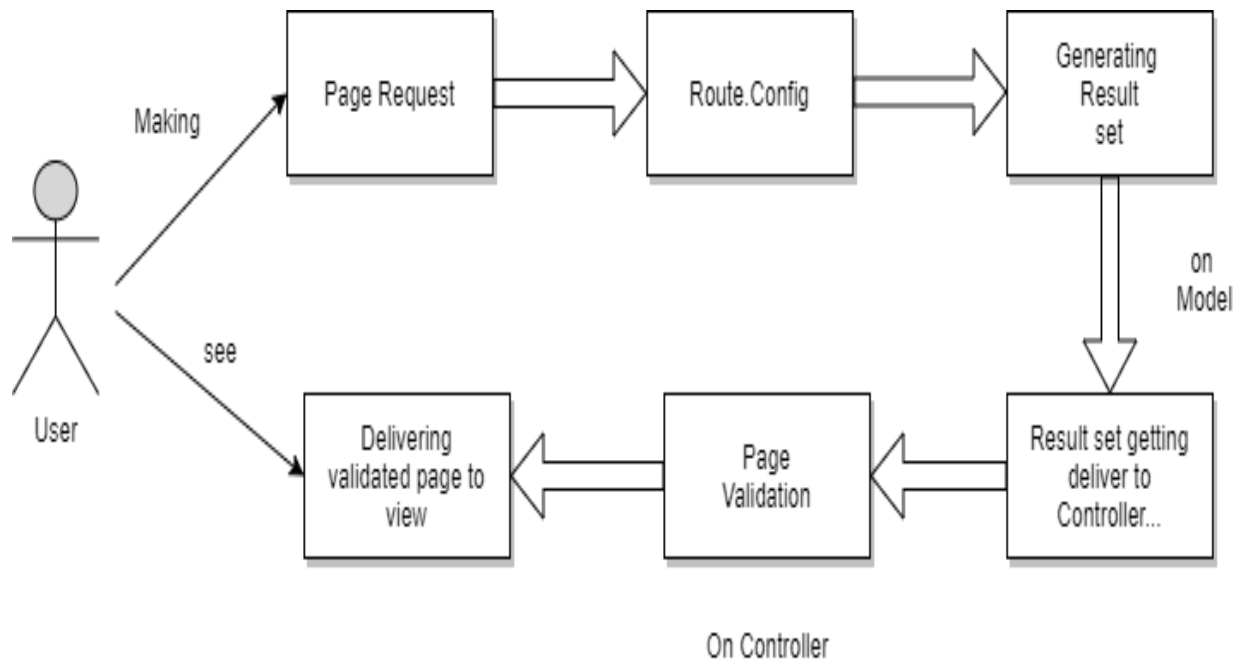


Fig.6: Box Diagram of MVC Life Cycle

6.2 Design notation

Component diagram of Plant Tracking System: The structural relationship of the system and elements. component diagrams generally simplify the interactions within more complex systems.

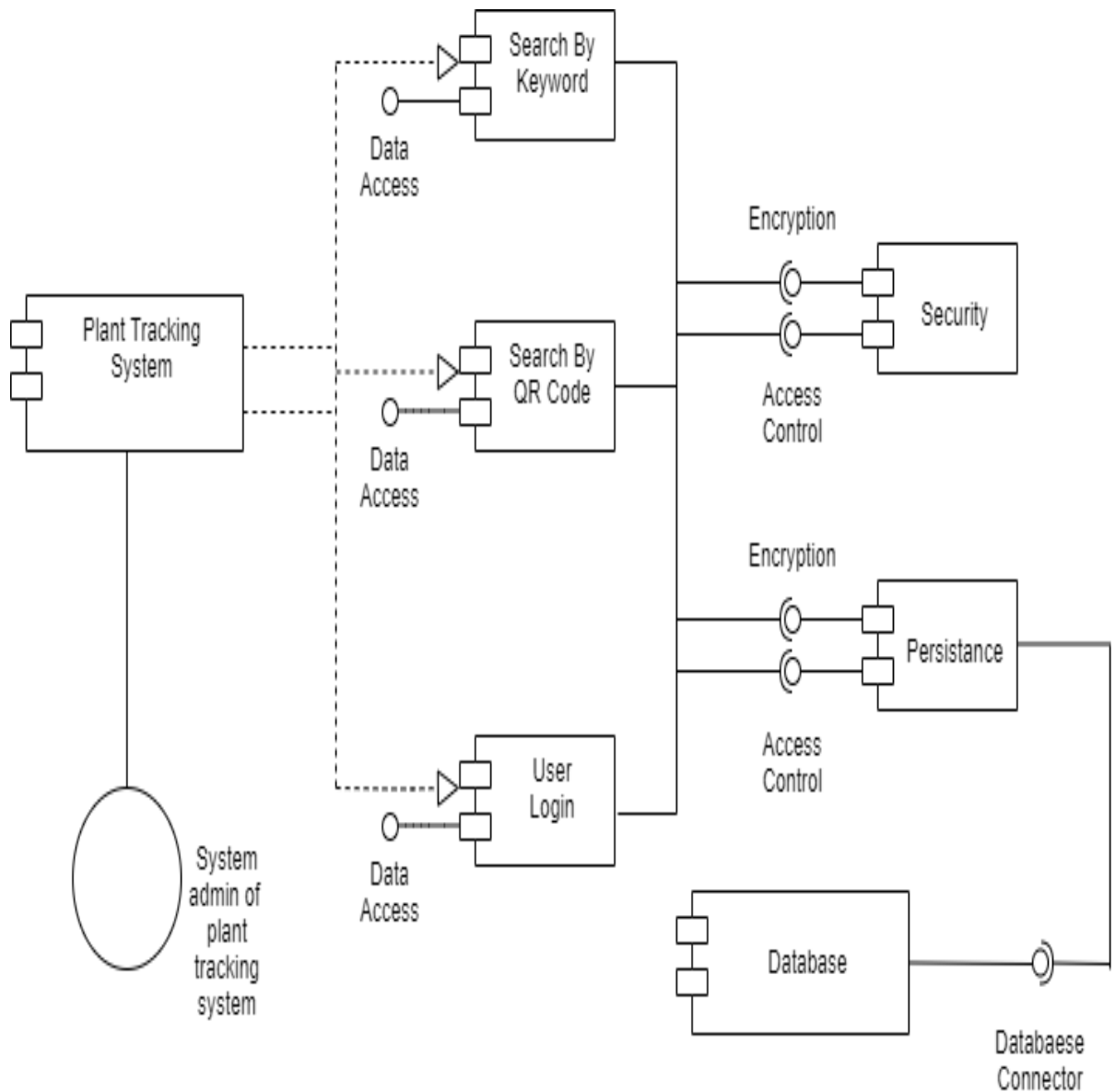


Fig. 7: Component Diagram of Plant Tracking System

6.3 Detailed Design

Now in this section we will discuss the detailed design of Plant Tracking System and define all the sequence diagram, ER diagrams and etc.

Sequence Diagram describes the each and every functionalities and feature of interface. It shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

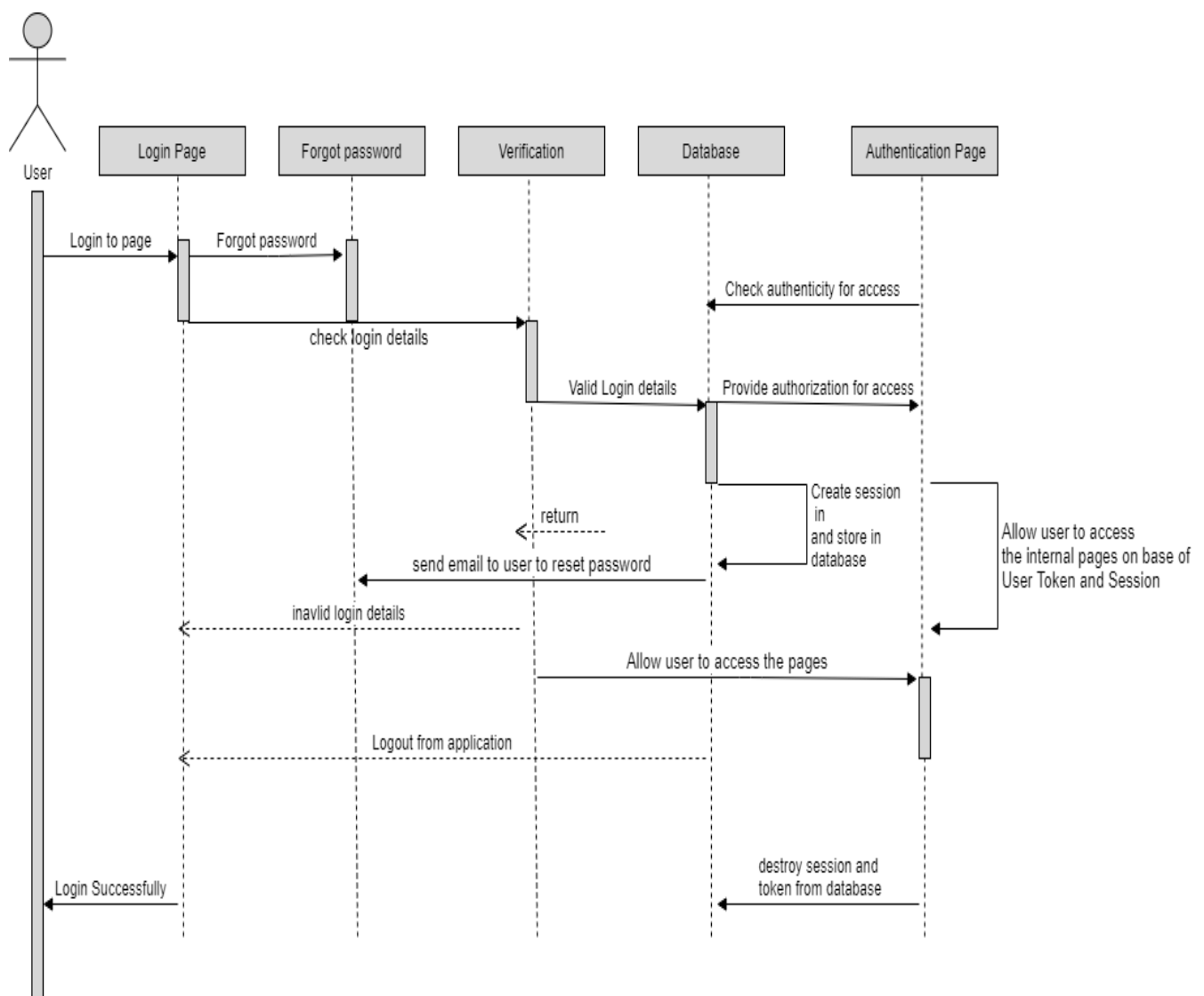


Fig 8: Login Sequence Diagram of Plant Tracking System

Next Sequence diagram depicts the Admin level interaction of object and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario

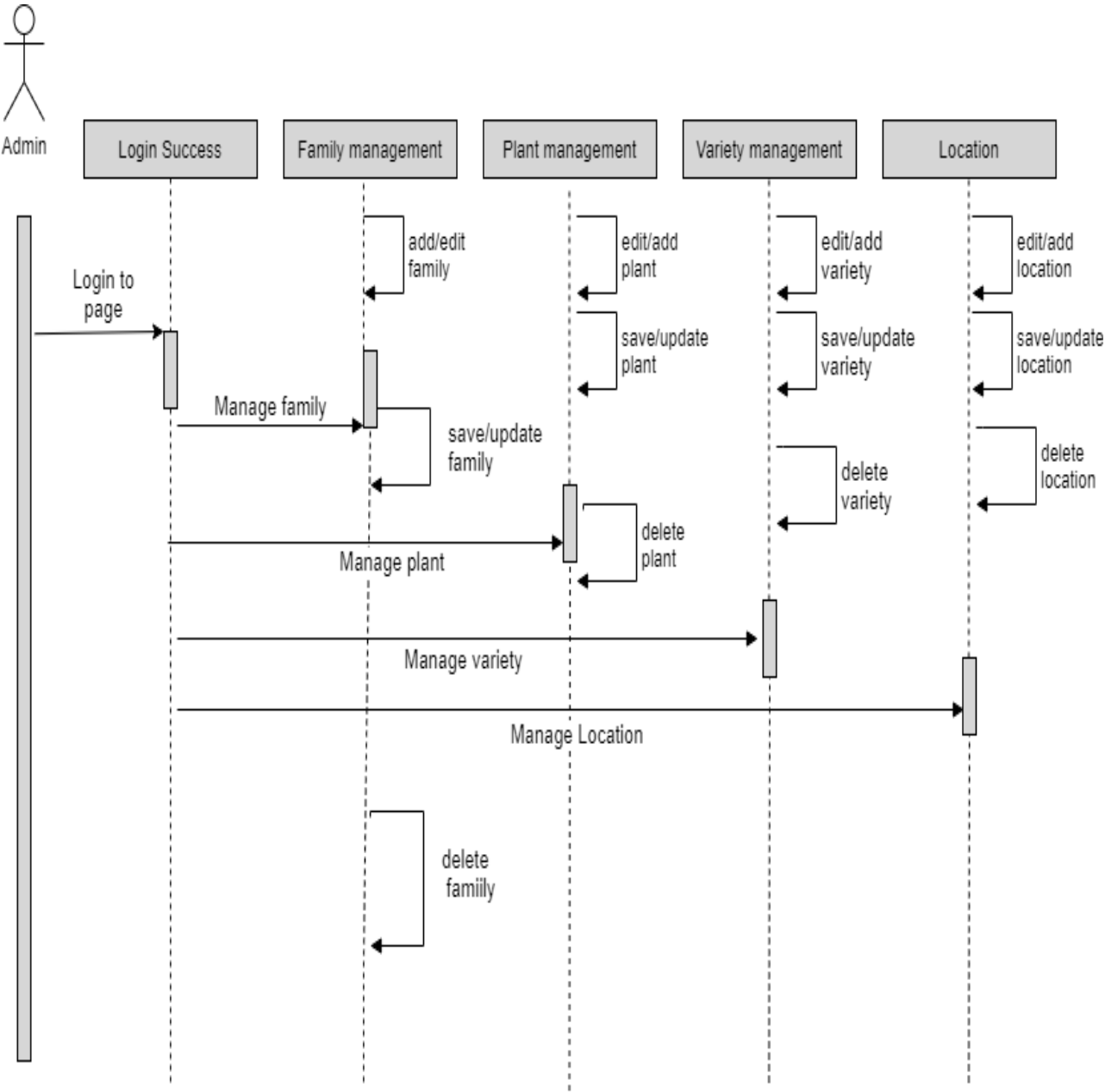


Fig 9: Admin Level Sequence Diagram

ER Diagram of Plant Tracking System

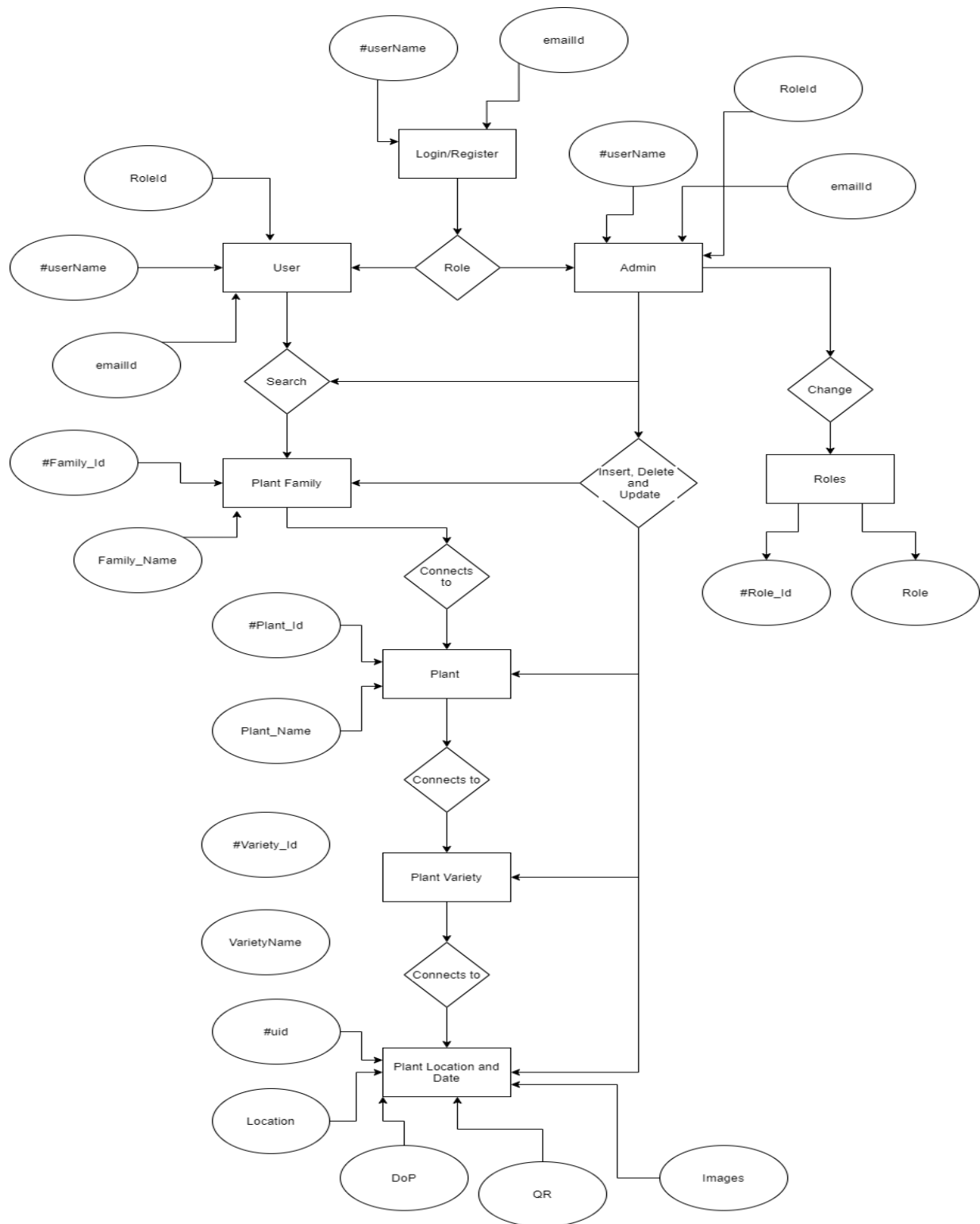


Fig.10: ER Diagram of PTS

Use case Diagram

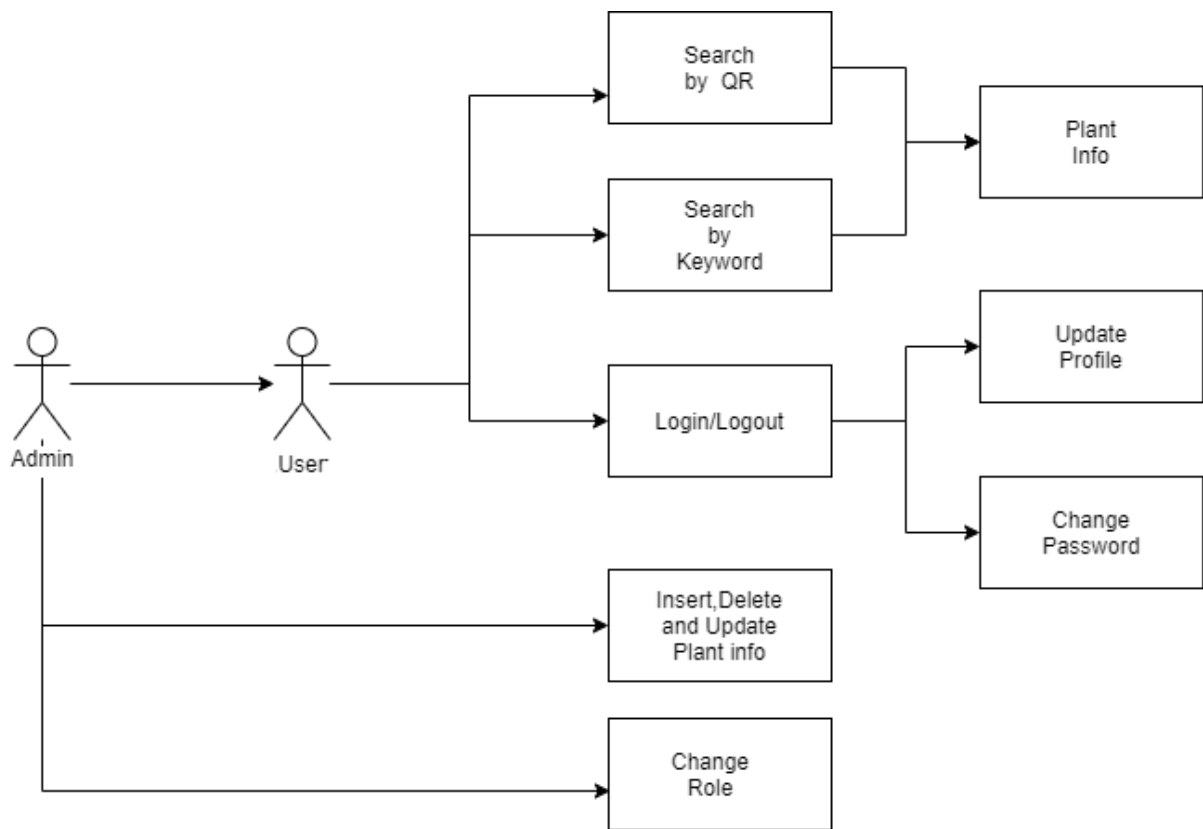


Fig.11: Use Case Diagram of PTS

7. Testing

Don't be that developer who is woken up in the middle of the night because of some problem with the web application. After all, you need your beauty sleep – some of us more than others. The best way to avoid problems with your application is to test thoroughly. Of course, there is a cost to testing, and it is easy to jump in too deeply such that every piece of code is outrageously well tested. Finding the right balance of just what to test and which ASP.NET Core testing tools to use is something that comes with experience.

Testing simple code that has a low impact too thoroughly is not as bad as failing to test complex code with a high impact, but it will still have a negative impact on the success

of your project. Complicating the problem is that we have a number of ways to test code. You may be familiar with the idea of the testing triangle or, for those who think a bit higher dimensionally, the testing pyramid.

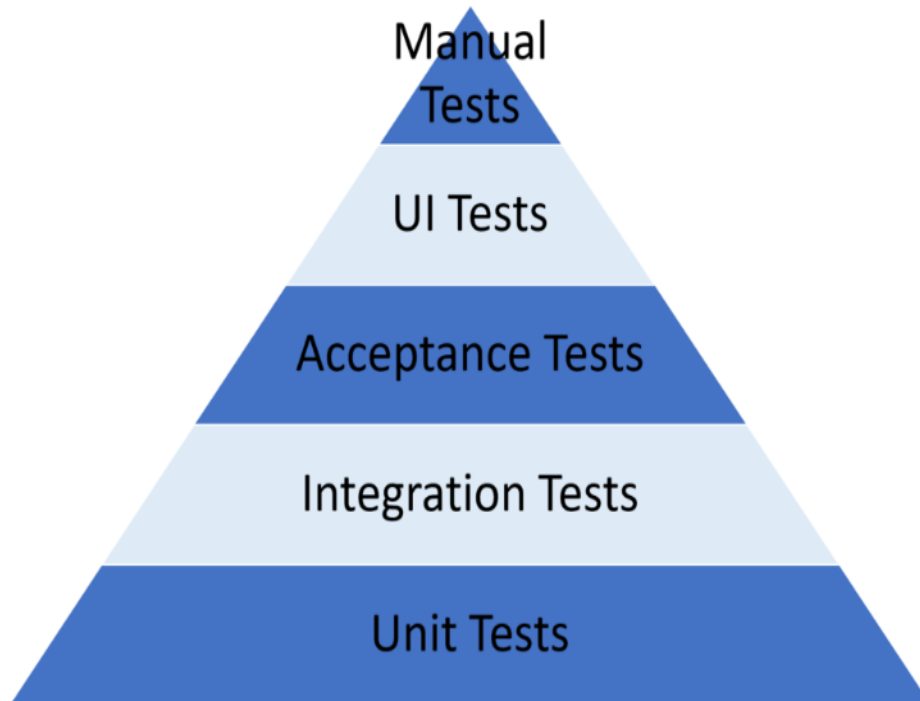


Fig.12: The Testing Triangle

7.1 The Testing Tringle

The purpose of the triangle is to demonstrate the number of tests you should have of each type. The bulk of your tests should be unit tests: these are tests which test a single public method on a class. As you move up the testing triangle the number of tests at that level decrease; conversely the scope of the tests increase. The cost of an individual test increases towards the top of the triangle.

In this article, we'll look at testing tools which can help us out on each level of the pyramid. Some of these tools will be specific to .NET, but others will be portable to almost any development platform, and we'll call out which is which as we progress. You will notice that as we increase the level of abstraction by moving up the triangle, the testing technologies become broader and applicable to more web development technologies.

The testing triangle is a good guide for your ASP.NET core testing strategies.

7.2 Unit Testing

Unit tests are the smallest sort of tests that you'll write. Ideally, they exercise a single method and should be trivial to write. In fact, many people suggest that if you find unit tests difficult to write then it is an indication that the code being tested is doing too much and should be split up. I use unit tests as an indicator that methods should be split up into multiple methods or even split into separate classes. These are the sorts of tests you should create during test-driven development.

Unit testing tools are not new in .NET or in any modern language. The migration to .NET has brought with it most of the first-class unit testing tools. Unit testing tools are divided into a number of categories: test frameworks, test runners and assertion libraries. The frameworks are a set of attributes that allow you to decorate your code such that they can be found and run. Typically, the testing frameworks also include a runner which will run the tests without starting up your entire application. Finally, there are assertion libraries that are used to check the conditions inside a test – most people stick with the assertions provided by their testing frameworks. All of these tools can run against any .NET project, not just ASP.NET.

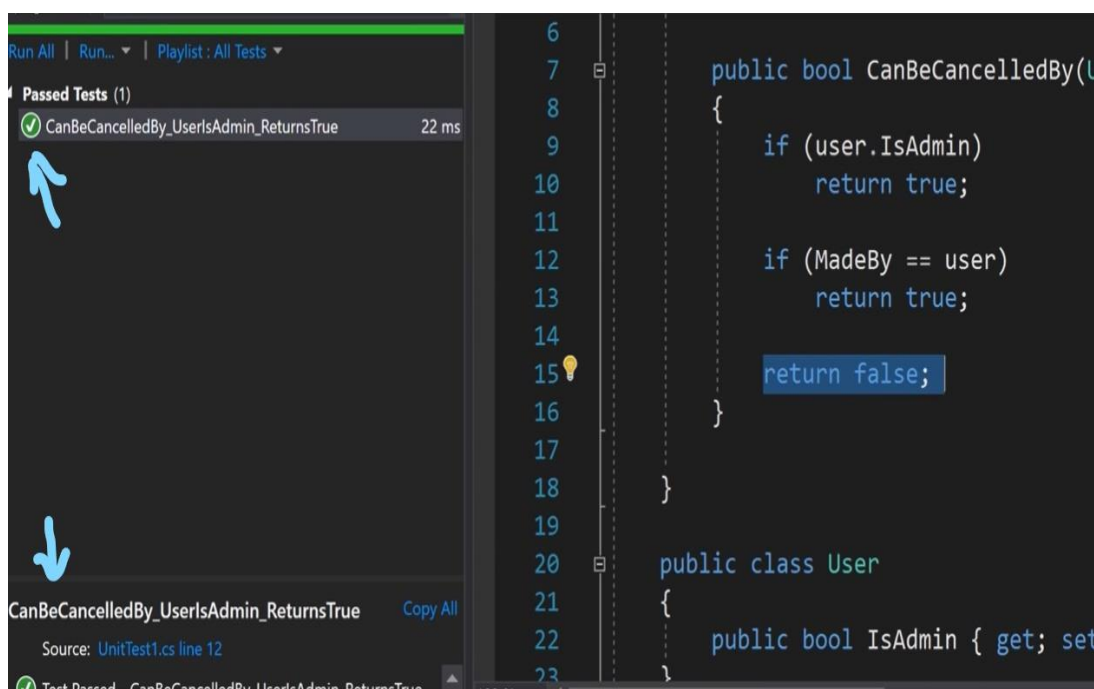


Fig.13: Screenshot of Unit testing

7.3 Database Integration Testing

In the past, I've written some pretty impressive pieces of code that stand up and tear down databases against which integration tests can be run. Although it was a pretty snazzy piece of

code at the time, the speed at which the tests could run was quite limited. Fortunately, we've moved a little bit out of the dark ages with Entity Framework Core. Just like the TestServer, EF Core provides an in-memory implementation of a database. This database can be used to test LINQ based queries with great rapidity. The one shortcoming of this approach is that queries written in SQL for performance or clarity reasons cannot be tested with the in-memory implementation. In those cases, I recommend reading Dave Paquette's article on integration testing with EF Core and full SQL Server.

7.4 Manual Testing

Manual testing is often seen as terrible drudgery. A lot of companies try to avoid manual testing in favor of cheaper automated tests. Automated tests cannot tell you what the user experience of your website is like. It is important to get a real variety of typical users to look at your site. The problems that manual testers are able to find are different from the ones you will find using unit, integration, or any other test.

8. Implementation

8.1 Implementation of the Project

To understand the implementation of PTS project, we will discuss in phase wise development of project taking the references of planning section but here we will talk about the execution and implementation part. Talking about technical execution after paper planning getting over, we moved for coding part. As we know we are going to develop a website in asp.net so we followed the steps accordingly. Let discuss whole implementation step wise:

Step 1: First we designed UI of our website which is developed using HTML, CSS, JavaScript and Bootstrap

Step 2: In next step after designing the front-end part we will moved to design our database which is designed on Microsoft SQL Server Management Studio (MSSMS). Before it we prepared all tables and relationships on paper then we moved for designing on studio.

Step 3: Now we created the tables of database according to the parameters and also established the relations and primary keys of tables.

Step 4: After this we also created the stored procedures to eliminate the repetition of queries for update, delete, insert.

Step 5: In the next after finishing database design, we moved to MVC coding means backend.

Here we connected the database from frontend and vice versa.

Let's Understand some concepts in an ASP.NET MVC application which we created

Controller: In MVC incoming requests are handled by the controllers. A Controller is a class inheriting from the built-in Controller class which implements the Controller interface.

Action: Each public method in the controller is an action that can be invoked using a URL.

View: Accepts the data in the ViewData objects passed to it by the Controller and renders them.

Model: Objects that are passed between controller and view.

Routing System: maps the URL's to controllers and actions (To decouple the URLs from the web pages and also makes the URL search engine friendly).

Step 7: While creating an ASP.NET MVC application in Visual Studio we have two choices of templates to choose from.

ASP.NET MVC 2 Web Application template: This creates a sample application preconfigured with things like authentication, styles and navigation.

ASP.NET MVC 2 Empty Web Application template: Create only the files and folders required by every ASP.NET MVC application.

Visual Studio provides us various templates for creating web applications.

Adding the Controller

Now we replace the code in the HomeController in the Controllers folder with the following code:

```
[HandleError]
public class HomeController: Controller
{
    public string Index()
    {
        return "PlantTackingSystem";
    }
}
```

Now when we run the application we will see "MVC is great!" rendered in the browser

window. We didn't create a view or action but still the correct output is displayed because under the default routing system configuration the application root (that is, the root of the virtual directory) is mapped to the Index action on the HomeController. So when we launch the application the Index() action is automatically invoked.

Adding the View

The above code correctly displays the intended output but mostly we will use views to render the HTML. We need to return a ViewResult type from the action method to use the view to render the HTML.

```
public ViewResult Index()
{
    return View();
}
```

Conventionally, views use the same name as the actions. Now we need to generate the default view for the above action method. To do so right-click on the Action and select "Add View" in the context menu.

Doing so generates an Index.aspx view which we can find in the views folder. Now we will replace the content in the Index.aspx view with our content "PTS is great!".

Running the application renders the same message as the previous one but this time makes use of the view.

The responsibility of the Controller is to generate some data which is rendered by the view so we need some mechanism to pass that data from the controller to the view. To do so we use the data structure called ViewData.

```
public ViewResult Index()
{
    ViewData["MVC"] = "PlantTrackingSystem";
    return View();
}
```

And replace the content we previously added in the Index.aspx view with the following:

```
<%: ViewData["MVC"]%>
```

So now instead of hardcoding the content in the view we are passing it from the controller.

This is how we implemented the MVC Concepts in developing our websites.

8.2 Post-Implementation and Software Maintenance

Step 1: Understand the system how it works

Step 2: Feed all the details of plant where to install this system

Step 3: Print all the generated QR code and install the printed QR into Plant, tree and pots.

Step 4: Now it's ready to use but when it comes to maintenance the you have keep updating the details, if new plant species in area then you add all detail about that in system. This system requires maintenance in only updates.

9. User Manual: A complete document (Help Guide) of the Plant Tracking System

User manual or Help Guide for the developed the project provides the key steps and measures as well as some tips to the user of the project who want to manage and control the web application as administrations and maintain its working.

Here there are different mode of using PTS website so we will explain all the scenarios of user. Let's understand the scenarios and learn how to use it:

Scenario 1: Direct Scan

Step 1: Each plant has its own unique QR Code so take out your phone and scan the QR Code and it will redirect to our website and display you detail information about that particular plant.

Step 2: If you don't have scanner app in your phone then visit our website and Click on the QR Scan Button that will open your camera to scan QR and then get the details.

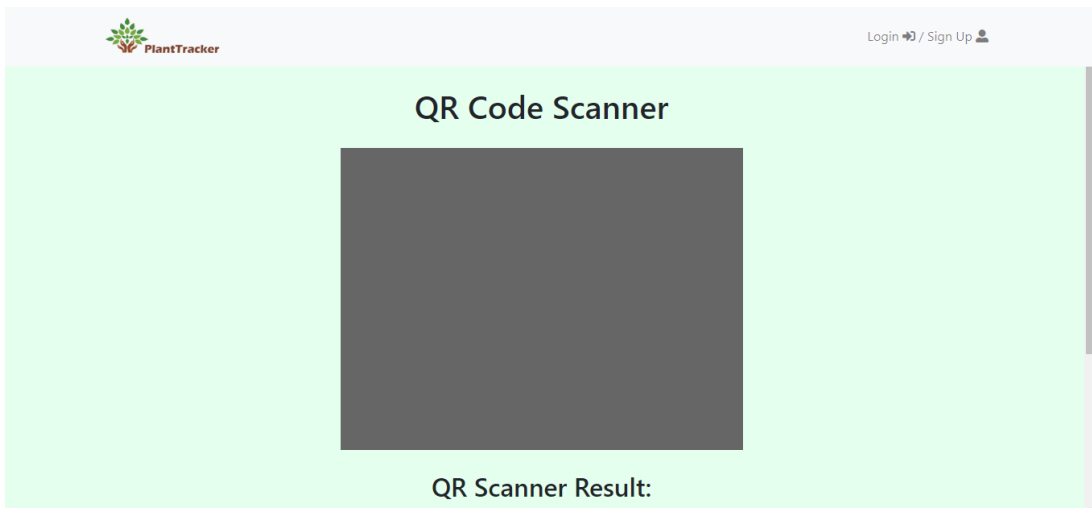


Fig.14: Search by QR part from website

Scenario 2: On website visit

Step 1: Run the project on Visual Studio.

Step 2: At home page you will get search box, QR Scan button and Login/Sign up Options.

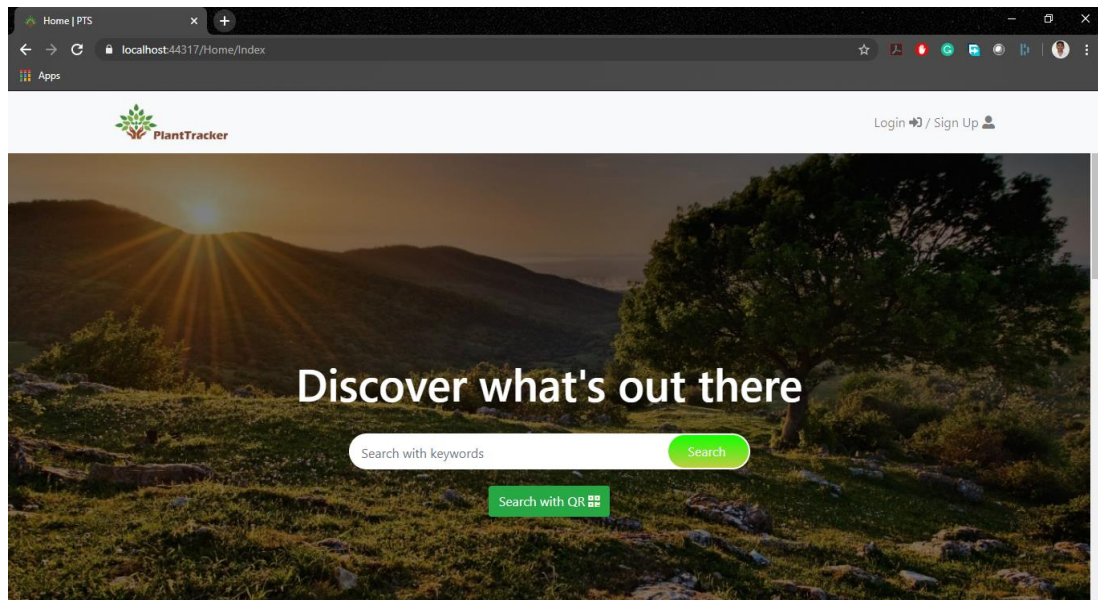



Fig.15: Home Page of website


Step 3: If you want to search about any plant then type in search box – get the details.

<div>  Login / Sign Up </div>		
About 6 results were found.		
Botanical Name: F1b1 Common Name: F1cn1 Family: Family1 Variety: F1b1v1 Location: infotech Date Planted:2020-04-03	Botanical Name: F1b1 Common Name: F1cn1 Family: Family1 Variety: F1b1v1 Location: 34 block Date Planted:2020-04-01	Botanical Name: F1b1 Common Name: F1cn1 Family: Family1 Variety: F1b1v1 Location: qr testing 1 Date Planted:2020-04-01
Botanical Name: F1b1 Common Name: F1cn1 Family: Family1 Variety: F1b1v1 Location: bh8 Date Planted:2020-04-17	Botanical Name: F1b1 Common Name: F1cn1 Family: Family1 Variety: F1b1v1 Location: Bh1 Date Planted:1990-12-11	Botanical Name: B1 Common Name: C1 Family: F1 Variety: V1 Location: BH-01 Date Planted:2020-12-01

Fig.16: Search details display page


Login / Sign Up

Plant Details



Family Name	Family1
Family Common name	Fcm
Subsist	
Common Name	F1cn1
Botanical Name	F1b1
Chromosome Number	50
Genus	
Species	
Uses	
Medical Benefit	
Health Hazard	
Variety Name	F1b1v1
Nature	N1
Time of Sowing	
Time of Flowering	
Rotation Period	
Propagation Method	
Tree Height	0
Trunk colour	
Tree form	
Leaf Shape	
Fragrance	
Wood Character	
Fruit Type	
Woods of value	

Fig.17: Details Search display page

Step 4: Or click on Scan button and scan the QR Code – get the details.

Scenario 3: Create PTS account

Step 1: Visit on landing page and click on sign-up button

Step 2: Fill the details

Step 3: Click on Register Button

Step 4: Verify your Email by clicking on verification link sent to your email

Step 5: After verification your account will be created successfully

Step 6: Go and login.

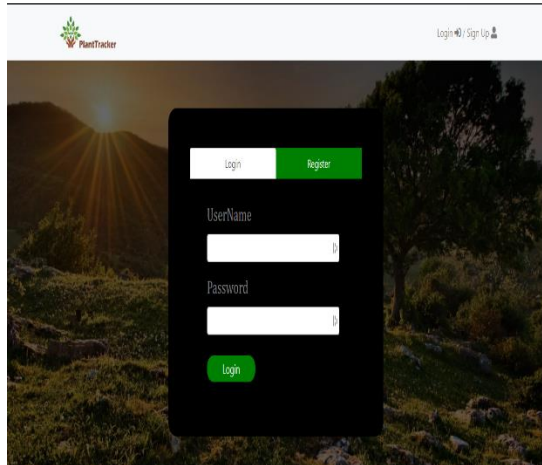


Fig.18: Login Page

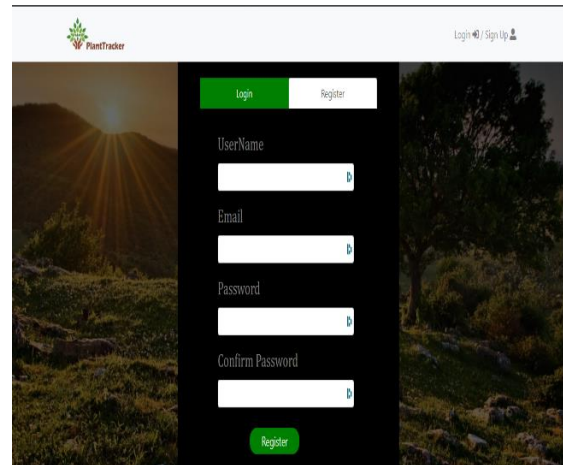


Fig.19: SignUp page

UML

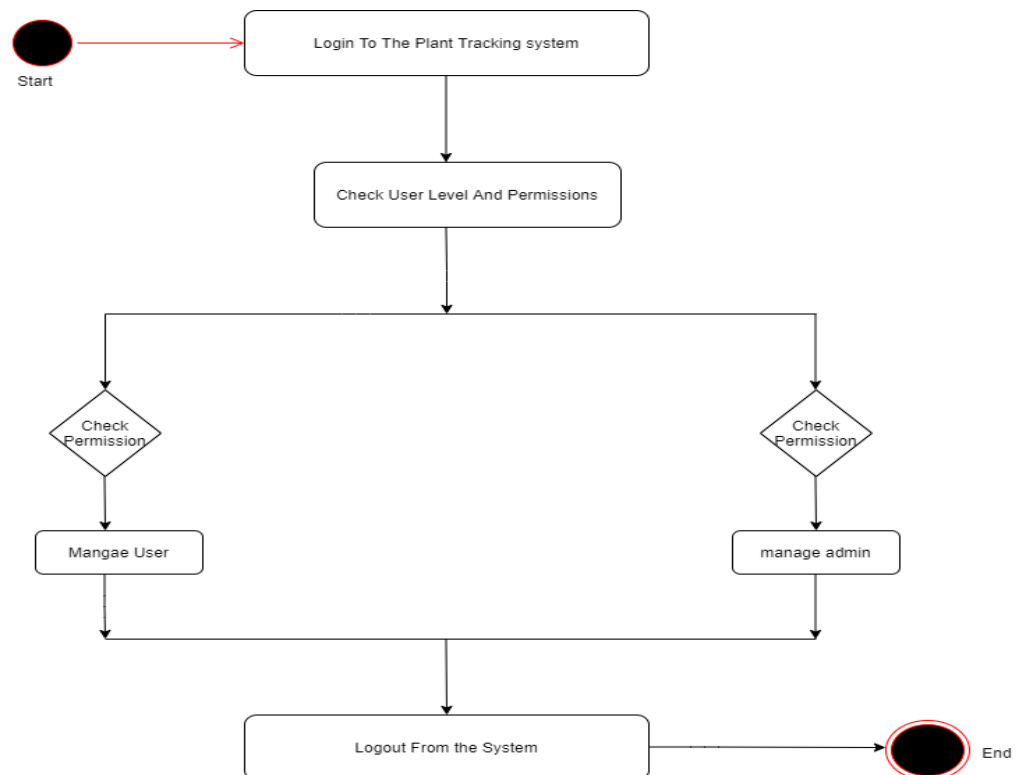


Fig.20: User login UML Diagram

Scenario 4: Login as admin then insert, update, delete options for plant details and change role options will display.

Step 1: Login with Admin Id. These Navigation bar will be displayed.

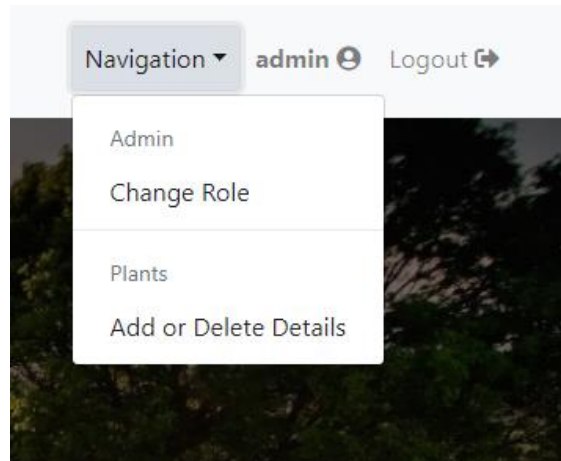


Fig.21: Admin options

Step 2: By selecting Change Role option Fig.22 will display and by entering email and selecting role admin can change role.

Grant/Revoke Admin Rights

Email	<input type="text"/>
Role	<input type="text" value="admin"/>
<input type="button" value="Submit"/>	

Fig.22

Step 3: By selecting Add or Delete Details option Fig.23 will be displayed and from drop down the family can be selected. If plant details exist admin have to delete from plant details first.

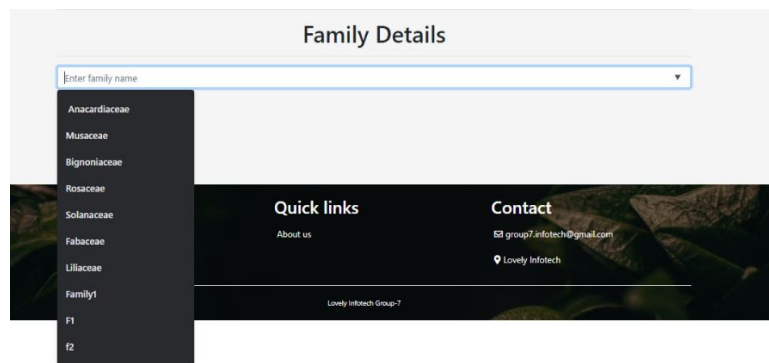


Fig.23

Step 4: Then it will redirect to plant details page. From that admin can add new plant or select existing plant. If variety details exist admin have to delete from variety table first.

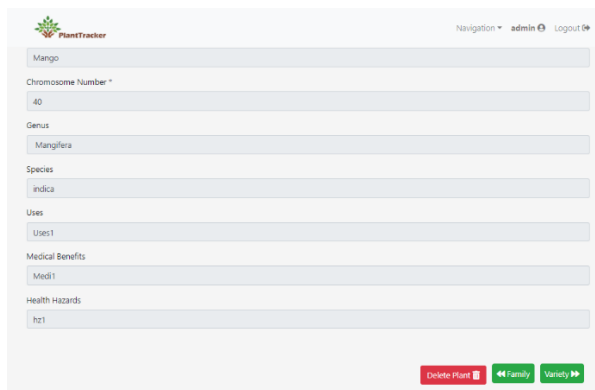


Fig.24: Admin Plant Details Page

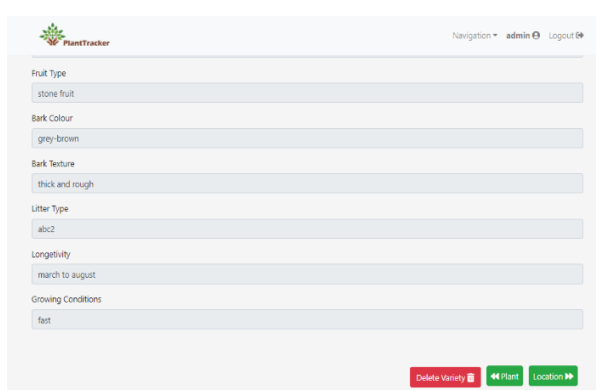


Fig.25: Admin Variety Details Page

Step 5: Finally, it redirects to Location details page Fig.26. After adding all Family, plant, variety and location details at the end we will click on “Submit & Generate QR” button then Fig.25 will be displayed.

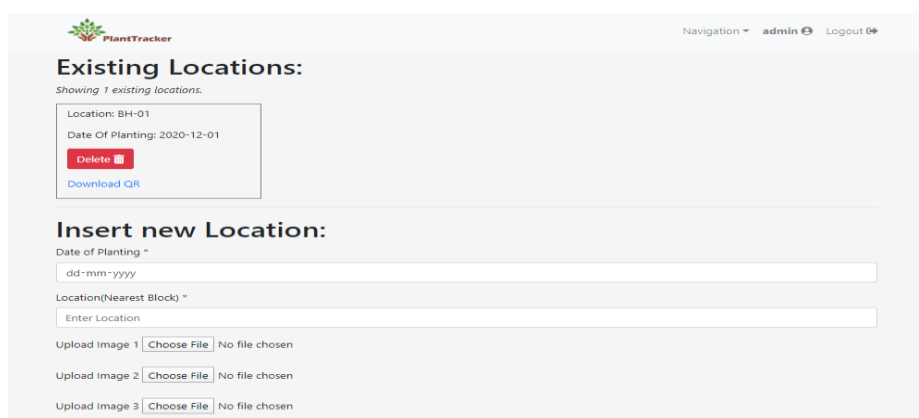


Fig. 26

Step 6: Download the QR as shown figure.

Step 7: Print the QR and attach to plant.

Step 8: If admin did any mistake and need to update details then admin can search and edit the details.



The screenshot shows the 'Admin Edit Details Page' in the PlantTracker application. The page has a header with the PlantTracker logo, a navigation menu, and user information (tarun). The main content area contains several input fields for plant details: Wood Character, Fruit Type, Bark Colour, Bark Texture, Litter Type, Longevity, Growing Conditions, Date of Planting (2020-04-03), and Location of Plant (infotech). A blue 'Edit Plant Details' button is located at the bottom left.

Fig.27: Admin Edit Details Page

SQL Server:

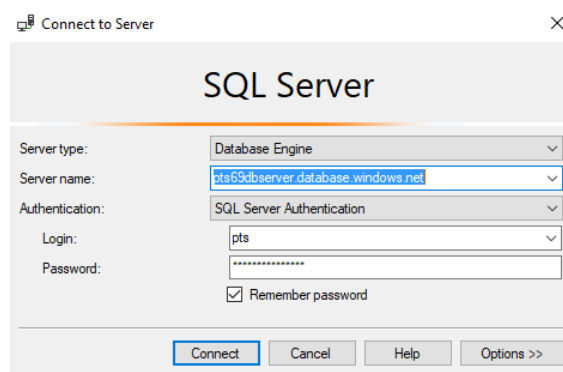
Step 1: We can access the database by entering following details.

Server Name: pts69dbserver.database.windows.net

Authentication: SQL Server Authentication

Login: pts

Password: group7@infotech



The screenshot shows the 'Connect to Server' dialog box in SQL Server. The dialog box has a title bar with a close button. The main content area is titled 'SQL Server'. It contains the following fields: Server type (Database Engine), Server name (pts69dbserver.database.windows.net), Authentication (SQL Server Authentication), Login (pts), Password (masked with asterisks), and a checkbox for 'Remember password'. At the bottom, there are buttons for 'Connect', 'Cancel', 'Help', and 'Options >>'.

Fig.28

Step 2: These are the designs of tables used for projects.

pts69dbserver.pts - dbo.FamilyMaster

Column Name	Data Type	Allow Nulls
Family_Id	int	<input type="checkbox"/>
FamilyName	varchar(50)	<input type="checkbox"/>
FamilyCommonName	varchar(100)	<input checked="" type="checkbox"/>
Habitat	varchar(100)	<input checked="" type="checkbox"/>
ModifyBy	varchar(100)	<input checked="" type="checkbox"/>
EntryBy	varchar(100)	<input checked="" type="checkbox"/>
ModifiedDate	datetime	<input checked="" type="checkbox"/>
EntryDate	datetime	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

Fig.29: FamilyMaster Table

pts69dbserver.pts - dbo.PlantMaster

Column Name	Data Type	Allow Nulls
Family_Id	int	<input type="checkbox"/>
Plant_Id	int	<input type="checkbox"/>
Common_Name	varchar(100)	<input type="checkbox"/>
Botanical_Name	varchar(100)	<input type="checkbox"/>
Chromosome_No	int	<input type="checkbox"/>
Genus	varchar(100)	<input checked="" type="checkbox"/>
Species	varchar(100)	<input checked="" type="checkbox"/>
Uses	varchar(MAX)	<input checked="" type="checkbox"/>
Medical_Benefit	varchar(MAX)	<input checked="" type="checkbox"/>
Health_Hazard	varchar(MAX)	<input checked="" type="checkbox"/>
PlantModifyBy	varchar(100)	<input checked="" type="checkbox"/>
PlantEntryBy	varchar(100)	<input checked="" type="checkbox"/>
PlantModifiedDate	datetime	<input checked="" type="checkbox"/>
PlantEntryDate	datetime	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

Fig.30: PlantMaster Table

pts69dbserver.pts - dbo.VarietyMaster

Column Name	Data Type	Allow Nulls
Variety_Id	int	<input type="checkbox"/>
Plant_Id	int	<input type="checkbox"/>
Variety_Name	varchar(100)	<input type="checkbox"/>
Nature	varchar(100)	<input type="checkbox"/>
TimeOfSetting	varchar(100)	<input checked="" type="checkbox"/>
TimeOfFlowering	varchar(100)	<input checked="" type="checkbox"/>
Rotation_Period	varchar(100)	<input checked="" type="checkbox"/>
Propagation_Method	varchar(100)	<input checked="" type="checkbox"/>
image	varchar(100)	<input checked="" type="checkbox"/>
Tree_height	real	<input checked="" type="checkbox"/>
Trunk_Color	varchar(100)	<input checked="" type="checkbox"/>
Tree_Form	varchar(100)	<input checked="" type="checkbox"/>
Leaf_Shape	varchar(100)	<input checked="" type="checkbox"/>
Fragrance	varchar(100)	<input checked="" type="checkbox"/>
Wood_Character	varchar(100)	<input checked="" type="checkbox"/>
Fruit_Type	varchar(100)	<input checked="" type="checkbox"/>
Bark_Color	varchar(100)	<input checked="" type="checkbox"/>
Bark_texture	varchar(100)	<input checked="" type="checkbox"/>
Litter_Type	varchar(100)	<input checked="" type="checkbox"/>
Longevity	varchar(100)	<input checked="" type="checkbox"/>
Growing_Condition	varchar(100)	<input checked="" type="checkbox"/>
VarietyModifyBy	varchar(100)	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

Fig.31: VarietyMaster Table

pts69dbserver.pts - dbo.LocationMaster

Column Name	Data Type	Allow Nulls
UniquePlant_Id	int	<input type="checkbox"/>
Family_Id	int	<input type="checkbox"/>
Plant_Id	int	<input type="checkbox"/>
Variety_Id	int	<input type="checkbox"/>
DateOfPlanting	varchar(100)	<input type="checkbox"/>
OtherDetails	varchar(MAX)	<input checked="" type="checkbox"/>
Location	varchar(50)	<input type="checkbox"/>
QR	varchar(MAX)	<input checked="" type="checkbox"/>
LocModifyBy	varchar(100)	<input checked="" type="checkbox"/>
LocEntryBy	varchar(100)	<input checked="" type="checkbox"/>
LocModifiedDate	datetime	<input checked="" type="checkbox"/>
LocEntryDate	datetime	<input checked="" type="checkbox"/>
img1	varchar(255)	<input checked="" type="checkbox"/>
img2	varchar(255)	<input checked="" type="checkbox"/>
img3	varchar(255)	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

Fig.32: LocationMaster Table

pts69dbserver.pts - dbo.userDetails

Column Name	Data Type	Allow Nulls
userName	varchar(50)	<input type="checkbox"/>
Role_Id	int	<input type="checkbox"/>
emailId	varchar(50)	<input type="checkbox"/>
password	varchar(50)	<input type="checkbox"/>
isVerified	bit	<input type="checkbox"/>
dateCreated	datetime	<input type="checkbox"/>
		<input type="checkbox"/>

Fig.33: userDetails Table

pts69dbserver.pts - dbo.RoleMaster

Column Name	Data Type	Allow Nulls
Role_Id	int	<input type="checkbox"/>
Role	varchar(50)	<input checked="" type="checkbox"/>
IsActive	bit	<input type="checkbox"/>
		<input type="checkbox"/>

Fig.34: RoleMaster Table

Step 3: These are stored procedure used for project.

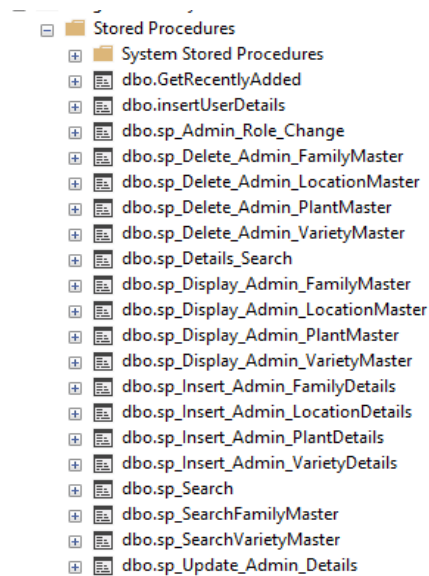


Fig.35: Stored Procedures

10. Source Code and System Snapshot

10.1 Table Entity Diagram of Database

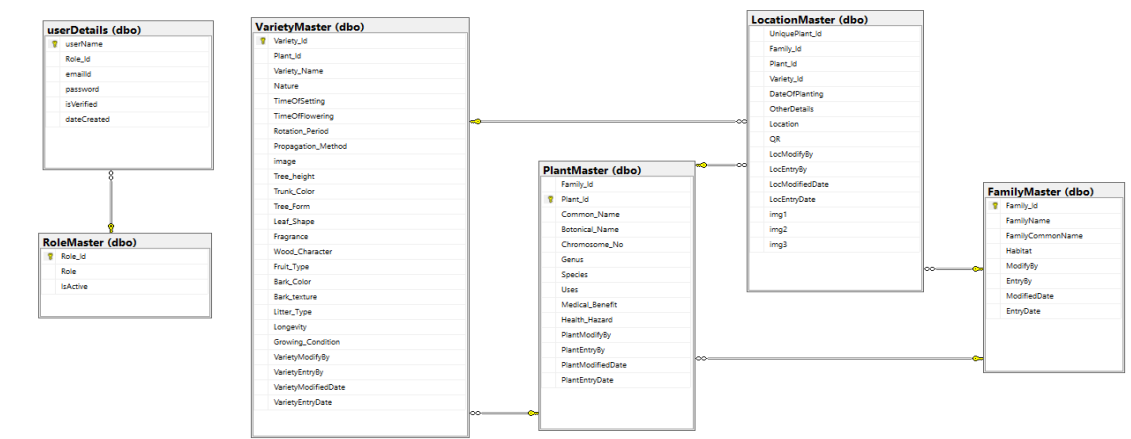


Fig.36

10.2 Solution Explorer

Here mainly we used HomeController manages home page, AccountController manages Login/SignUp and Role change, PlantController manages crud operations and FamilyApiController, PlantApiController, VarietyApiController, LocationDateApiController manages suggestion part that display while admin inserting or deleting data.

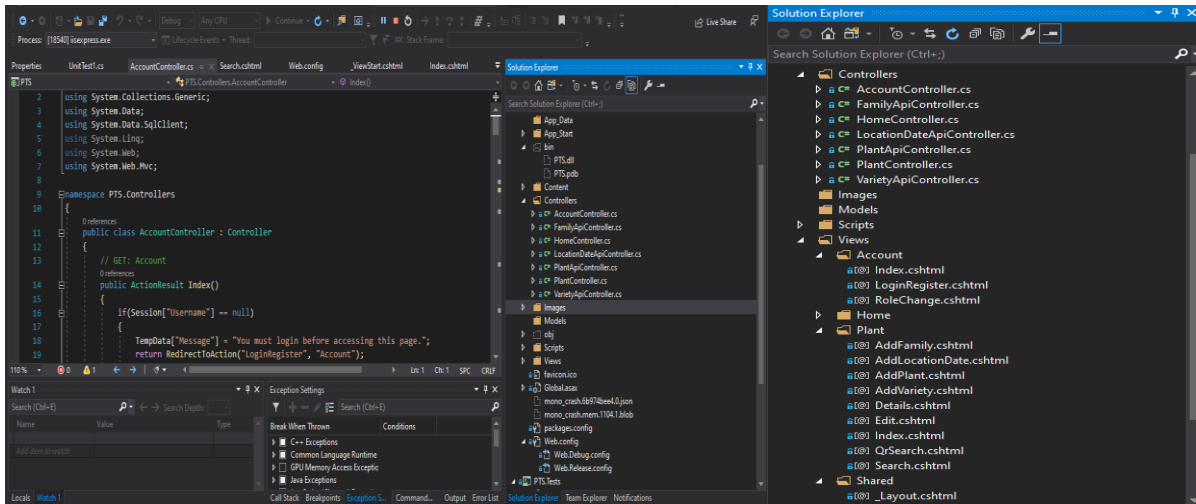


Fig.37

Fig.38

10.3 Source code

The code used for the project is available in this link. [Click here](https://github.com/rajavundela/PTS)

<https://github.com/rajavundela/PTS>

11. Bibliography

Websites:

1. <https://www.indiaplants.com/customer-login.php>
2. <http://plantinfo.co.za/>
3. <https://www.chicagobotanic.org/plantinfo>
4. <https://plantinfo.umn.edu/>

Concept help:

1. <https://www.c-sharpcorner.com/>
2. <https://stackoverflow.com/>
3. <https://www.tutorialsteacher.com/mvc/asp.net-mvc-tutorials>