Search Around Me Smart Location Allocator Application

By:

Ayesha Arooj

2019-GCUF-08041

The project is submitted for the partial fulfillment of

The requirement of the degree of

BACHELOR OF SCIENCE

IN

SOFTWARE ENGINEERING

DEPARTMENT OF SOFTWARE ENGINEERING

GOVERNMENT COLLEGE UNIVERSITY FAISLABAD

August 2023

DECLARATION

The work mentioned in this document to accomplish the project was carried by us under the supervision of "**Mr.Nauman-ul-Haq**" lecturer Department of Software Engineering, Government College University Faisalabad.

We hereby declare that the "Search Around Me Smart location Allocator Application " and all the contents of this project are the outputs of our efforts and Research. We further declare that all the work which we mentioned in this document has not been submitted for the award of any other degree or diploma. The university may take action if the information provided in this document is inaccurate at any stage.

Signature of the student**:** _____

STATEMENT OF SUBMISSION

This is to certify that Ayesha Arooj (8985) has successfully completed the final year project named as "**Search Around Me Smart Location Allocator App"** at the department of Software Engineering Government College University Faisalabad to fulfill the partial requirements of the, degree of BS (Software Engineering).

Project Coordinator

Project Examiner

Department of Software Engineering

Government College University Faisalabad.

Table of Content

Introduction....8

1.1 Scope of Project....8

1.2 Objective....8

1.3 Motivation....9

1.4 Background....9

1.5 Project description and goals....10

1.6 Definition:....10

1.7 Software Project Management Plan....11

1.8 Project Overview....11

1.9 Project Deliverables....11

1.9.1 Evolution of SPMP....13

Project Management....14

2.1 Project Organization....14

2.1.1 Process Model....14

2.2 External structure....15

2.3 Internal structure....15

2.4 Supervisor....16

2.5 Development leader....16

2.6 Managerial process....17

2.7 Cost Estimation Process....17

2.8 Software resources....17

2.9 Management Objectives and Priorities....17

2.10 Assumptions and constraints....18

2.11 Risk Management....18

2.12 Monitoring and Controlling Mechanism....19

2.13 Staffing Plan....20

2.14 Software Documentation....20

Designing & UML Diagram....22

3.1 Work Breakdown Structure For Frugal Healthcare Collaboration System....23

3.1.1 What is a work breakdown structure in project management?....24

3.1.2 What is the purpose of creating a work breakdown structure?....24

3.2 Use case diagram....24

3.3 Sequence Diagram.....25
3.3.1 Key parts of a sequence diagram:.....26
3.3.1.1 Participant:.....26
3.3.1.2 Message.....26
3.3.2 Sequence diagram for Finding Doctor.....27
3.3.3 Sequence diagram for Creating doctors Schedule.....28
3.3.4 Sequence diagram for View Schedule.....29
3.3.5 Sequence diagram for Requesting an appointment.....30
3.3.6 Sequence diagram for Appointment Canceling.....30
3.4 User Interface.....31
3.4.1 User Interphase for User Login.....32
3.4.2 User interphase for Doctors registration.....32
3.4.3 User interphase for Doctors and patients view appointment.....33
3.4.4 User interphase for Admin.....34
3.4.5 User interphase for Patients.....34
3.4.6 User interphase for patients to view Schedules.....35
3.4.7 User interphase for Patients Get appointment.....36
3.4.8 User interphase for change password.....37
3.4.9 Conformation Message for Doctor.....38
3.4.10 Conformation message for Patient.....39
Implementation.....41
4.1 Layered Architecture.....42
4.2 Creating Frugal healthcare collaboration system MVC application.....46
4.3 Code.....54
4.3.1 Java Script Code.....54
4.3.2 C# Code.....55
4.3.3 C# Code for Controller in MVC.....56
4.3.4 C# Code for Controller in MVC.....57
4.3.5 Repositories For database connection.....57
4.3.6 API Controllers.....58
4.3.7 Class's code.....59
4.3.8 DB Context.....60
4.4 Creating SQL Database.....61

4.4 Tables In database.....64
4.4.1 Appointment table.....64
4.4.2 Patient table.....65
4.4.3 Doctors table.....65
4.5 Frugal healthcare collaboration system database Views.....66
4.6 Stored Procedure for Frugal healthcare collaboration system database.....67
4.6 Class Diagram for frugal healthcare collaboration system database.....68
Software Quality Assurance & Testing.....69
5.1 SOFTWARE QUALITY ASSURANCE PLAN:.....70
5.1.1 Purpose.....70
5.1.2 SQA Program Audits.....70
5.1.3 Scheduled Audits.....70
5.1.4 Unscheduled Audits.....70
5.1.5 Audit Reports.....70
5.1.6 Formal Reviews.....71
5.1.7 Informal Reviews.....71
5.1.8 Quality Reviews.....71
5.2 Software Testing.....71
5.2.1 Hardware Testing.....72
5.2.2 Unit Testing.....72
5.2.3 Integration Testing.....72
5.2.4 System Testing.....73
5.2.5 Validation Testing.....73
5.3 Document problems.....73
5.3.1 Code problems.....73
5.3.2 Problem solving procedure.....74
5.3.3 Code control.....74
5.3.4 Hardware control.....74
5.3.5 Media control.....74
5.3.4 Supplies control.....74
5.3.5 Training.....75
5.4 Risk management.....75
Postmortem report For Frugal Healthcare Collaboration System for Doctor and Patient.....76

6 Postmortem report..... 77
6.1 Description..... 77
6.1.1 Project Name..... 77
6.1.2 Client..... 77
6.1.3 Project Manager..... 77
6.1.4 Solutions Architect..... 77
6.1.5 Start Date..... 77
6.1.6 Completion Date..... 77
6.2 Project Overview..... 77
6.2.1 Project success criterion?..... 78
6.3 Performance..... 78
6.3.1 Key Accomplishments..... 78
6.3.2 Key Problem Areas..... 79
6.3.3 Risk Management..... 79
6.3.4 Overall Project Assessment..... 80
6.3.5 Additional Comments..... 80
6.4 Key Lessons Learned..... 80
6.5 Post Project Tasks/Future Considerations..... 80
Summary of the project frugal healthcare collaboration system..... 81

Chapter 1

Introduction

1.1 Scope of Project

It is well known today that the discoveries changed the face of technology and that change of technology also changed the world. Technology plays a very important role in every field of life like in Automobiles, communication and learning etc. So by seeing this innovation why we ignore to automate our specific locations searching system for different categories and activities around us. The purpose of this mobile app project revolves around creating a user-friendly application that enables users to discover and explore various places within a specified range based on selected categories. Users will have the ability to choose a category such as "food", ”Banks", ”Entertainment” or "travel" and set a search radius between 1 to 10 kilometers. The app will then display a list of relevant places within the designated range, presenting essential details like names and addresses. The user interface will be straightforward, providing a map view for easy visualization and navigation to the selected places. The app's core functionality lies in simplifying the process of finding nearby places of interest, streamlining user experiences, and enhancing their ability to explore their surroundings conveniently. The user interface will be mobile-friendly, offering search, filtering, and communication functionalities. Security measures will ensure data protection. The goal is to provide a user-friendly, efficient tool for allocating and managing locations while ensuring a seamless mobile experience.

1.2 Objective

The core objectives of the “Location Allocator App” revolve around creating a user-friendly platform that simplifies the process of discovering and accessing local places of interest. The app aims to cater to users who seek an efficient and convenient way to explore nearby establishments. The primary objective is to streamline the exploration process by allowing users to select categories and specify a search radius between 1 to 10

kilometers. This feature enables users to find relevant places that match their interests and needs, such as restaurants, cafes, or attractions, without being overwhelmed by excessive information. Some more Specific objectives include:

- **Efficient Exploration:** Enable users to efficiently explore nearby places within a specified radius, facilitating quick decision-making about where to visit based on their preferences.
- **Customizable Range:** Allow users to set their desired search radius, giving them control over the proximity of the displayed places and tailoring the search to their current location and preferences.
- **Simplified Information:** Present users with essential information about each place, including names and addresses, to help them make informed choices without overwhelming them with excessive details.
- **Enhanced Exploration:** Facilitate users' exploration of their surroundings by providing clear directions to selected places, contributing to a more enjoyable and stress-free travel experience.

1.3 Motivation

The motivation behind this project stems from the desire to simplify and enhance the way people explore and discover local places of interest. In an increasingly fast-paced world, individuals often seek efficient and user-friendly solutions to make informed decisions about where to go. This app aims to address this need by providing a seamless platform that empowers users to easily find nearby establishments based on their preferences. By offering a customizable range, clear categorization, and essential details, the app aims to inspire users to explore their surroundings with confidence and convenience, ultimately enriching their overall experiences.

1.4 Background

The project is rooted in the changing dynamics of urban living and the increasing reliance on mobile technology. Traditional methods of exploring local areas are often time-consuming and lack personalization. This project aims to leverage modern

technology to bridge this gap by providing a user-friendly mobile app that enables efficient discovery of nearby places within specified ranges and categories. The project is fueled by the aspiration to offer users a seamless way to navigate their environment, fostering a more informed and enjoyable exploration of their surroundings.

1.5 Project description and goals

Project Description:

This mobile application is designed to simplify the process of discovering and exploring local establishments within a specified radius. Users can select categories of interest and set a search range between 1 to 10 kilometers. The app displays relevant places on a map, presenting essential information for each location.

Goals:

- Develop a user-friendly mobile app to streamline local exploration.
- Enable users to select categories (e.g., food, travel) for personalized search.
- Provide a customizable search radius between 1 to 10 kilometers.
- Display places on a map, aiding visual exploration and navigation.
- Offer clear and concise details about each place, including names, distance and addresses.
- Facilitate informed decision-making by presenting essential information.
- Foster spontaneous exploration by providing a simple and efficient tool.
- Create a platform that encourages users to confidently explore their surroundings.

1.6 Definition:

SPMP	Software project management plan
IEEE	Institute of Electrical and Electronics Engineer
SRS	Software requirement specification document
SQA	Software quality assurance

WBS Work Breakdown Structure

QA Quality assurance

1.7 Software Project Management Plan

All the activities must be delivered on time. Excellent management must be done by managing all the assumptions like

- The Developer has enough experience personally to complete the project.
- The Developer will work together with the supervisor to complete the project.
- The Developer will respond in a timely manner to all questions from the Supervisor.
- Additional resources might be available to the project.

1.8 Project Overview

The project is to create a mobile app allowing users to easily find and explore nearby places based on chosen categories and search radius. With a user-friendly interface, map visualization, and offline capabilities, the app aims to simplify local exploration and decision-making.

1.9 Project Deliverables

There are 9 project deliverables that must be delivered on time with working software.

Table 1.1: Project deliverable of Frugal Healthcare Collaboration System

Document Deliverables	**Description**	**Activities**
Software Project Management Plan (SPMP)	Description of the software approach and associated milestones.	• System requirement analysis

		• Software requirement analysis
Software Quality Assurance Plan (SQAP)	Description of plan that consists of a means of monitoring the software engineering processes and methods used to ensure quality	• Product quality • Process quality
Software Requirements Specifications (SRS)	Description of the expected software features, Constraints, interfaces and other attributes.	• Process implementation
Software Design Description (SDD)	Description of how the software will meet the requirements. Also describes the rationale for design decisions taken.	• System architectural design • Software architectural design • Software detailed design
Software Test Documentation (STD)	Description of the plan and specifications to verify and validate the software and the results.	• Software qualification testing • System qualification testing
User manual	Description of Instructions for hands-on users of the software	User manual

Working software	Working software fulfilling all functional and quality requirements	Working software
Post mortem report	Description of Individual Reflections on Degree Project	Post mortem report

1.9.1 Evolution of SPMP

Any changes made in SPMP must be forwarded to the Supervisor and it is assured that it is monitored and under the control of the supervisor.

Chapter 2

Project Management

2.1 Project Organization

The SPMP will identify the organizational entities external to the project as well as internal project structure and roles and responsibilities for the project.

2.1.1 Process Model

The account box project will follow **waterfall** model for its deliverables

- Requirements are less likely to change.
- Phases are completed and it is an ongoing process.
- The stages mentioned and used in the model are clear to understand.
- All the other process model phases are derived or underpinned from waterfall model
- Ease of management.
- Works well when requirements are clearly understood.

Table 2.1: Process Model for Smart Location Allocator App

Phase	**Activity**	**Deliverable**
Requirements Specification	Display visual component to users and get feedback on User Interface and product requirements.	Prototype and SRS

Design	Design the system to the client's specifications.	Software Design Document
Project Management Planning	Defines the management plan to deliver the product within the time constraints of the schedule.	Project Management Plan
Implementation of the project	Here we code the project. So that it work properly.	Codes
Quality Assurance	Ensure the product and quality of tasks through the course of the project.	Software Quality Assurance Plan
Testing	Ensure that there is no bug in the working software and software is defect free	Software test plan and test report
Deployment of project	Must deploy the software with all its features hence and documents with training	Working software

2.2 External structure

This app is designed to cater to individuals seeking a hassle-free way to discover and navigate local establishments within their vicinity. Whether someone is new to an area or simply looking for nearby places of interest, this app provides a streamlined solution. As part of our commitment to user satisfaction, feedback about the app's performance is highly valued, helping us continuously enhance the experience and assist users in their exploration journey.

2.3 Internal structure

Table 2.2: Responsibilities for Smart Location Allocator App

Roles	**Responsibilities**

QA/Process manager(Supervisor)	● Set up and manage communication. ● Review and publish test cases on requirements gathering sessions.
Planning manager	● Maintain optimal level of performance within required budget and ensure compliance to all standard projects.
Development manager	● All the 9 related documents must be written and deliver on time ● Must force to adopt writing standards of IEEE
Document writer	● Must deliver working system with documentation
Developer	● Remove all bugs and conduct white and black box testing ● Prepare test report

2.4 Supervisor

- Motivate the developer to perform their tasks
- Help the developer in the tasks and resolving issues
- Creates and maintains project SRS

2.5 Development leader

- Lead the project in producing the development strategy
- Lead the development of project SRS
- Lead in producing the high-level design
- Lead in producing the design specification
- Lead in implementing the product
- Lead in developing the build, integration and system test plans
- Lead in developing the test materials and running the tests

- Lead in producing the product's user documentation

2.6 Managerial process

The SPMP will specify the project management processes for the project and will include:
The project start-up plan is called project proposal, risk management plan, project development plan, project SRS plan and project test plan etc.
In order to be successful the team must deliver a software product that will satisfy the needs of the client as outlined in the SRS of the account box project.

2.7 Cost Estimation Process

For the cost estimation of the project we use the COCOMO **Model**. Which finds the cost in terms of the KLOC and also calculates the amount that we spent on hardware purchasing. After calculating both costs we add these costs to find the total cost of the project.

2.8 Software resources

Each developer must have the Visual studio, Internet, node Js (installed in the system), and a smart grip of coding on HTML/CSS/JS/React Native . Developers must make sure that Software's resources must be available before development starts.

2.9 Management Objectives and Priorities

Objectives of management are to assure proper delivery of working software along with all documents. Management is assuring for the proper submission of all the deliverables. The main objective of management is time to time delivery of all deliverables before the last or closing date of the project. Management must be monitored and appreciated and motivated by the Supervisor. Working software with all the features mentioned in SRS must be given priority. Priorities must be assigned on the basis of timely delivery of deliverables. So that it can motivate developers to close and deliver projects on time before the closing date. A good management has many objectives: it saves time and cost, never overruns and assures the possibility of success.

2.10 Assumptions and constraints

There are several assumptions and constraints that are of importance for the project and its team members

- The student has enough experience personally to complete the project.
- The student will respond in a timely manner to all questions from the staff.
- Additional human resources might be available to the project.

There are several constraints too like

- Due to the nature of the project and its dependability on already existing solutions and technologies, third party software and already available solutions will be used in the project as needed
- Additional financial resources are not available for the project.

2.11 Risk Management

The SPMP shall specify:

- Risk management plan for identifying, analyzing and prioritizing project risk factors.
- Procedures for contingency planning and the methods that will be used for tracking certain risk factors, changes in levels of the factors and responses to those changes.

Table 2.3: Risk Management of Smart Location Allocator App

Risk	Possibility	Impact	Solution
Time out	High	High	Using all the available resources including educating team members before starting project, Timely delivery of components
Unavailable resources	Low	Medium	All-important data must be stored in

			USB or on email account till resources will be available
Team member absence	Medium	Medium	Share all knowledge in every meeting.
New technology	High	High	Online research, self-learning tutorials , use natural skills, Reuseof components
Lack of communication	High	High	Staff monitor provide friendly environment
Budget over runs	High	High	Timely delivery of deliverables and component, estimation of cost before starting project, resource allocation

2.12 Monitoring and Controlling Mechanism

- Weekly project meetings with supervisor , project is taken according to MS project plan
- Shared document repository.
- Weekly goals shall be adjusted
- Documents must be updated after every meeting

2.13 Staffing Plan

- The account box project development team has a fixed staff that was set at the beginning of the project and is mandated by the BSSE program
- All Team members are responsible for the documentation of SRS,SPMP,SDD,SQA etc
- All act as developer, team leader, tester, requirement engineer and analysts.
- Staff shall monitor and assure the timely delivery of all the documents and working software.
- Staff can act as a team manager and support manager

2.14 Software Documentation

There are a number of documents that will be produced during the lifetime of the project. All documents are the responsibility of the project student.

Table2.4: Software Documentation of Smart Location Allocator App

Document Deliverables	**Description**	**Activities**
Software Project Management Plan (SPMP)	Description of the software approach and associated milestones.	• System requirement analysis • Software requirement analysis
Software Quality Assurance Plan (SQAP)	Description of plan that consists of a means of monitoring the software engineering processes and methods used to ensure quality	• Product quality • Process quality
Software Requirements Specifications	Description of the expected software features, Constraints, interfaces and other attributes.	• Process implementation

(SRS)		
Software Design Description (SDD)	Description of how the software will meet the requirements. Also describes the rationale for design decisions taken.	• System architectural design • Software architectural design • Software detailed design
Software Test Documentation (STD)	Description of the plan and specifications to verify and validate the software and the results.	• Software qualification testing • System qualification testing
User manual	Description of Instructions for hands-on users of the software	User manual
Working software	Working software fulfilling all functional and quality requirements	Working software
Post mortem report	Description of Individual Reflections on Degree Project	Post mortem report

Chapter 3

Designing & UML Diagram

3.1 Work Breakdown Structure for Smart Location Allocator App

The Work Breakdown Structure (WBS) is a guiding framework for the development of the Smart Location Allocator App. It systematically dissects the project into manageable tasks and components, offering a clear perspective of its scope and arrangement. The visual representation of the "Smart Location Allocator App”

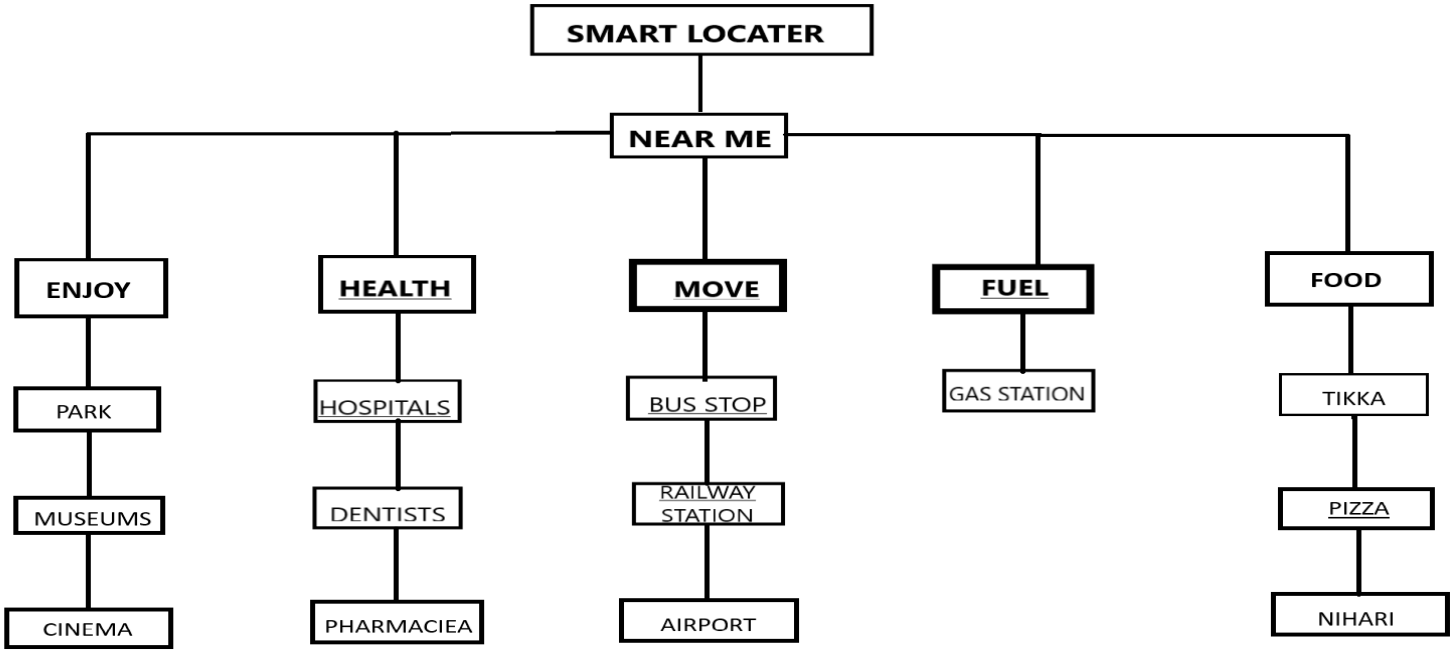


Figure 3.1: *Work breakdown structure for Smart location Allocator App*

A Work Breakdown Structure (WBS) serves as a method to deconstruct a project into smaller, more digestible segments. Comparable to disassembling a substantial undertaking into its constituent parts, this approach enhances comprehension and project management efficacy

3.1.1 What defines a work breakdown structure in project management?

A1: The Smart Location Allocator App is designed to help users easily locate and access essential services and points of interest such as health services, transportation hubs, and fuel stations. It simplifies the process of finding nearby facilities and navigating to them.

3.1.2 Why do we make a work breakdown structure**?**

Creating a Work Breakdown Structure (WBS) serves as a strategic approach to streamline the project's intricacies. By breaking down the project into smaller, distinct tasks, the overall complexity is reduced. Rather than grappling with intricate elements, we segment them into manageable, clear-cut tasks. This approach notably facilitates the project manager's oversight and management. Furthermore, each task becomes more quantifiable and executable on its own. In the context of this project, observing the WBS provided above exemplifies how all the project's activities are methodically dissected into simplified tasks, which promotes comprehensibility and ease of management..

3.2 Use Case Diagram for Location Allocator App

A Use Case Diagram provides a comprehensive overview of how various actors engage with the system and its functionalities. Within the framework of the "Smart Location Allocator App," the Use Case Diagram visually represents user interactions with the app. It encompasses actors, use cases, and their interconnections. elucidates the Use Case Diagram, offering insight into how actors like users, healthcare professionals, and administrators interact with the app's features

- Use cases epitomize discrete actions or functionalities provided by the system. In the realm of a Location Allocator App, use cases might encompass "Search for Available Locations," "Book Location," and "Cancel Booking."
- A full use-case model comprise of:

- A document describing the use case in detail.

- Actors encapsulate the entities—whether individuals or external systems—that interface with the system. In the diagram, various actors are delineated, each representing a distinct role or stakeholder within the app's ecosystem.
- The diagram employs arrows to articulate relationships between actors and specific use cases. This visual representation conveys how actors orchestrate specific actions within the system. For example, a user may trigger the "Search for Available Locations" use case.

- A use case diagram furnishes a visual manifestation of a system's functional requisites. It not only aids development teams in comprehending actor-system interactions but also illuminates the nexus among various use cases. This diagram acts as a blueprint for how actors interface with the system and provides a holistic understanding of the system's operational scope.

- By formulating a use case diagram tailored to a Location Allocator App, you enhance your grasp of the system's demands and the intricate interplay between diverse actors and functionalities. The diagram serves as a high-level guide that expedites development, design, and communication by offering a simple yet illustrative portrayal of the system's dynamics.

.

By creating a use case diagram for a Location Allocator App, you can gain a better understanding of the system's requirements and how different actors interact with it. This diagram serves as a high-level overview and can assist in the development and design process.

TThis visual representation offers an accessible depiction of the system's interactions, ensuring clarity for a wide audience, including stakeholders, developers, and users.

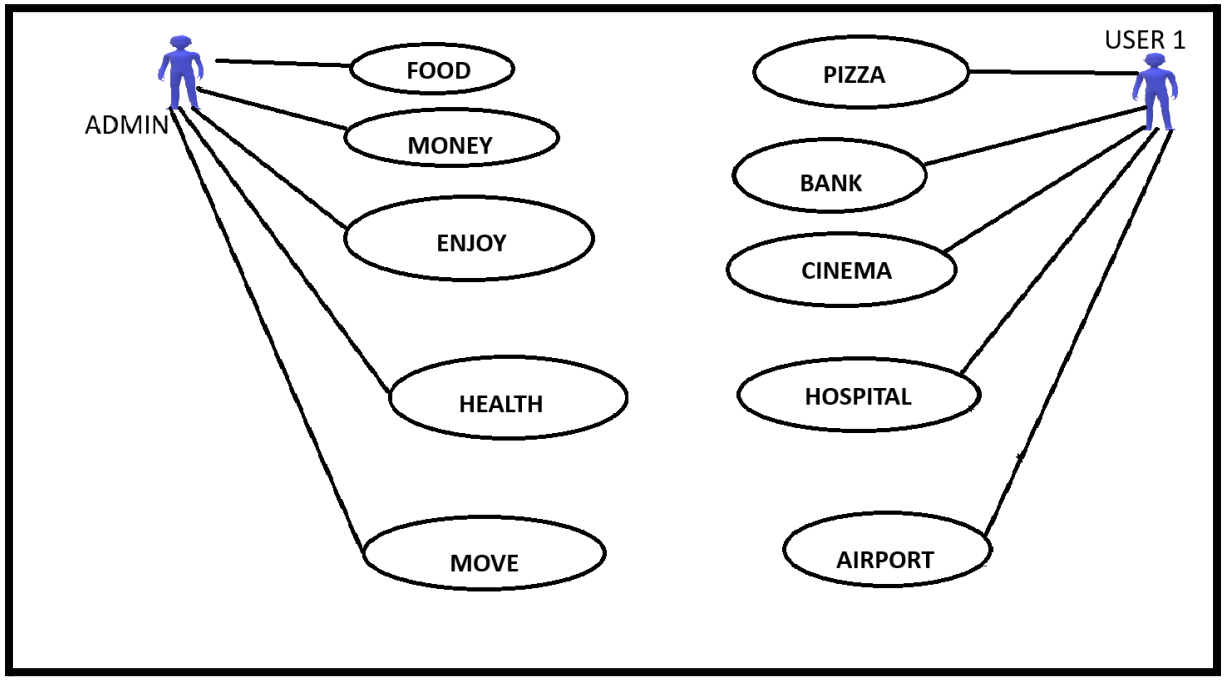


Figure 3.2: UML Diagram of Location Allocator App

Sequence Diagram

Sequence Diagrams offer a visual representation of the sequential interactions between various components within the system. These diagrams vividly depict the chronological flow of events and messages in diverse scenarios. Within the context of the "Smart Location Allocator App," we've meticulously crafted several Sequence Diagrams to illuminate essential processes. These diagrams serve to illustrate actions like finding a doctor, establishing a doctor's schedule, viewing schedules, requesting appointments, and canceling appointments. Figures 3.3 through 3.7 encapsulate instances of Sequence Diagrams that delineate these scenarios.

3.2.1 Key parts of a sequence diagram:

3.2.1.1 Participant:

- Participants denote the entities or objects that actively partake in the sequence diagram. They signify the elements interacting within the depicted scenario.

3.2.1.2 Message

- The sequence diagram's initiation is marked by an unconnected "found message" arrow, denoting the commencement of the sequence. Messages traverse between participants, delineating the interactions and communications transpiring between them.
- For instance, in the scenario of scheduling a doctor's appointment, the Sequence Diagram chronicles the flow of events. The user initiates the process by sending a "Create Schedule Request" message. The system subsequently examines the availability of the doctor and the requested time slot. In the absence of conflicts, the system proceeds to establish the schedule and updates the database correspondingly. Ultimately, the system dispatches a confirmation message back to the user, affirming the successful creation of the schedule.
- These Sequence Diagrams, encapsulating intricate sequences of interactions, bolster comprehension by offering a visual narrative of how the system's components cooperate under specific circumstances. By tracing the flow of messages and events, development teams and stakeholders gain a deeper insight into the system's operational dynamics.
-

3.2.2 Sequence diagram for Finding According Healths in Location Allocator app

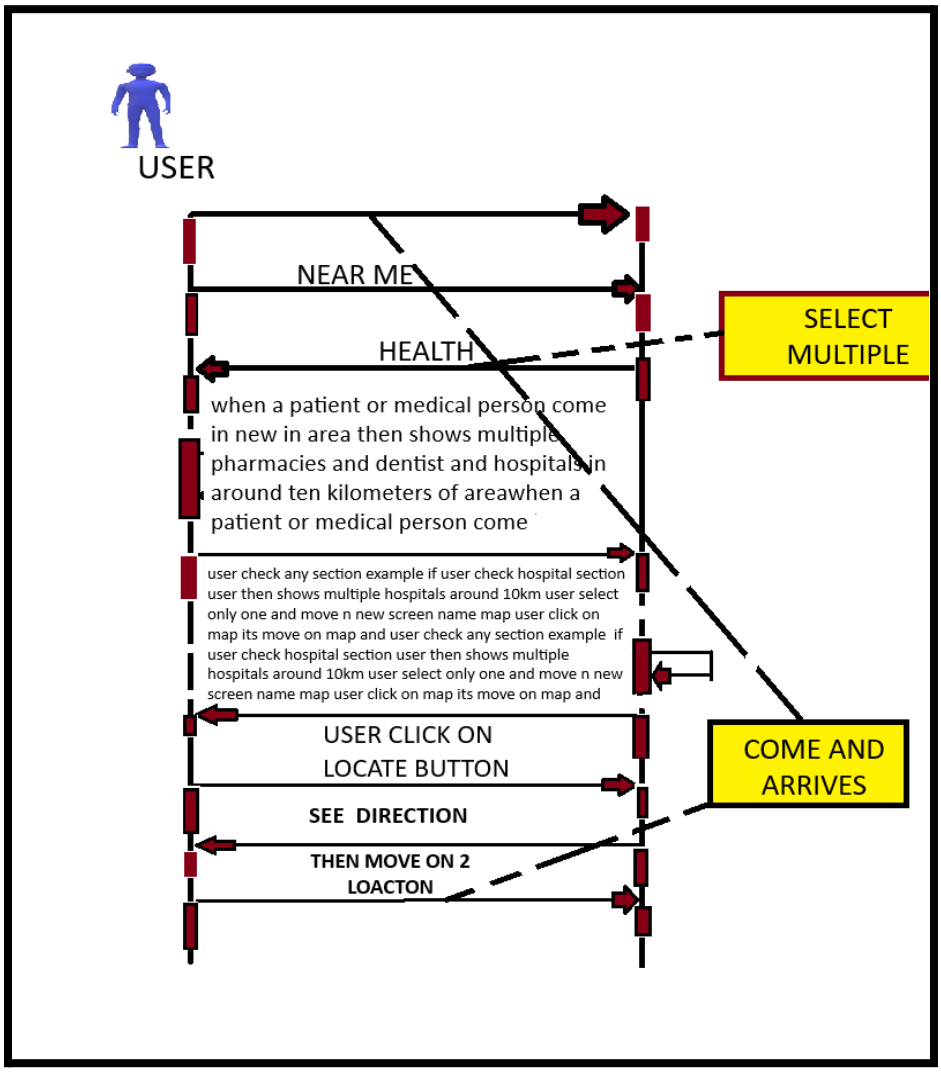


Figure 3.3: UML diagram for finding About Health in Location Allocator App

In the Health section of the "Smart Location Allocator App," the UML Sequence Diagram serves as a powerful tool to visualize the intricacies of user-system interactions. This diagram provides a lucid representation of the sequential steps involved in creating a scenario within the Location Allocator app. Through this diagram, users and stakeholders can gain a comprehensive understanding of the communication and coordination between various entities within the system.

3.2.3 Sequence diagram for Food Structure flow

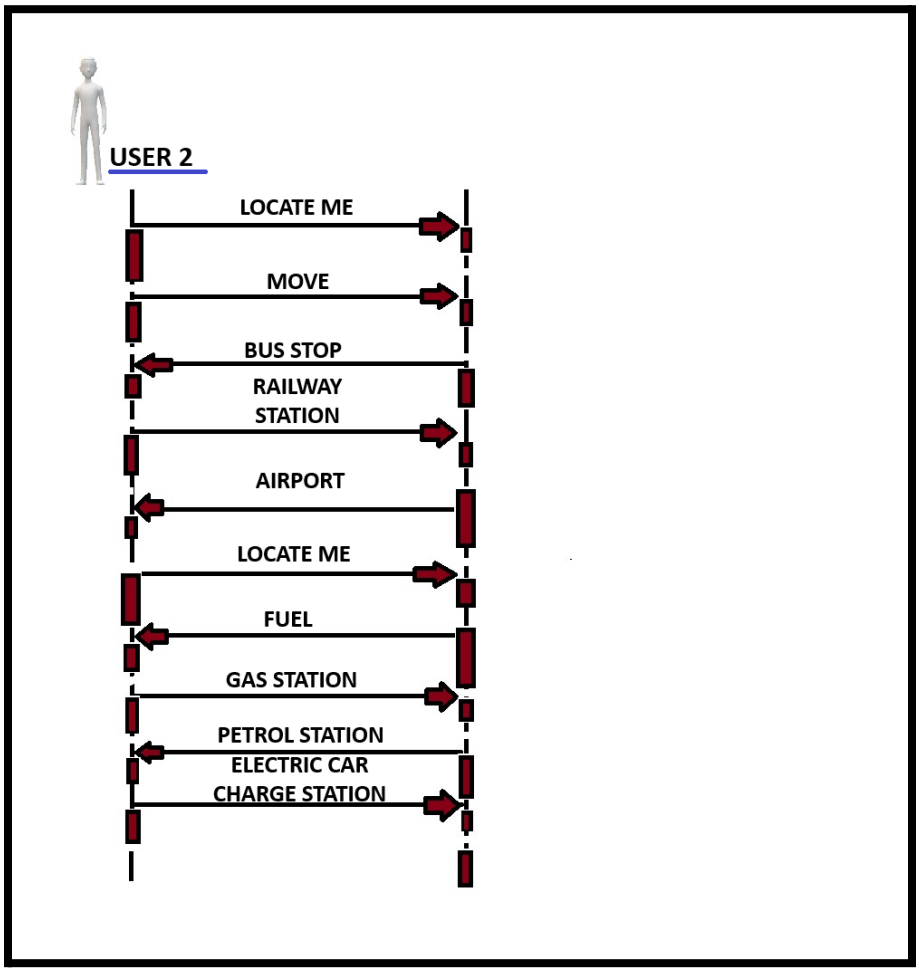


Figure 3.4: UML diagram Structure flow Smart location Allocator App

The Food Locator feature allows users to explore nearby food options. After selecting the "Food" category and clicking the "Locate" button, users can view a list of available food choices within a specified radius. The app also provides the option to view more food options beyond the initial list.

3.2.4 Sequence diagram for View Schedule

Figure 3.5 UML diagram for view schedule in Special Location Allocator

the move section within the Smart Locator app, users encounter an interface tailored to transportation. This section emphasizes seamless travel by presenting

information about bus stops, railway stations, and airports. Users can initiate the process by clicking the "Locate" button. Once activated, the app displays relevant details for each transportation category, facilitating efficient navigation within the user's vicinity.

3.2.5 Sequence diagram for Food Shops

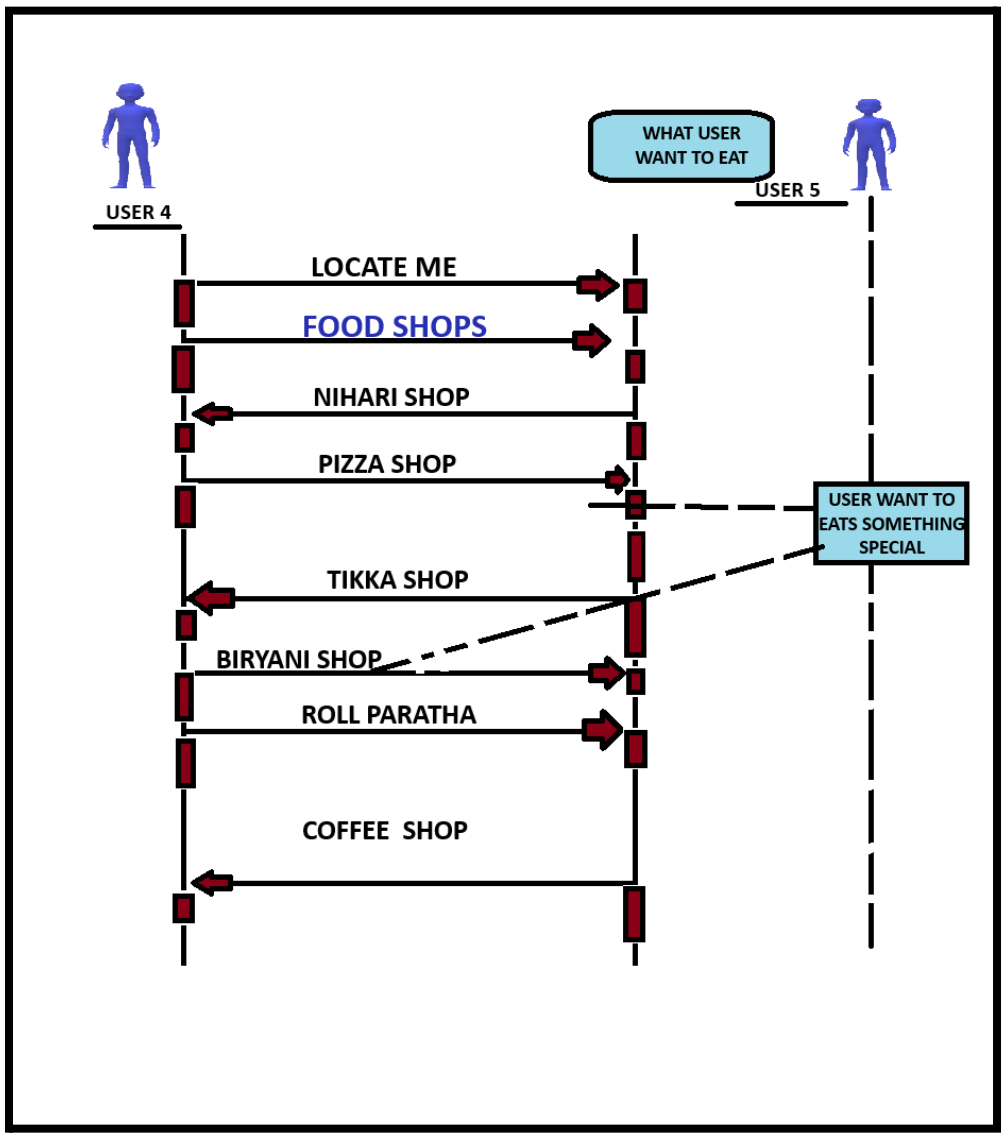


Figure 3.6 UML diagram For Food section about Location Allocator

In this diagram the patient finds a Hospital,dentist or pharmacies and finds his schedule and selects a suitable available time slot in a suitable day by selecting a time slot the accepts the term and condition that he reaches the hospital 20 minutes before and clear his dues .

3.2.6 Sequence diagram for Appointment Canceling

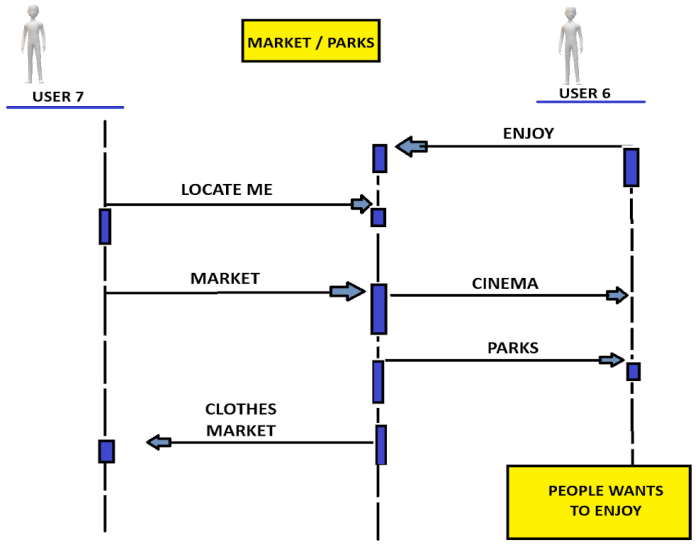


Figure 3.7: UML diagram for Canceling appointment in frugal healthcare collaboration system

3.3 User Interface

User Interface (UI) design is critical for ensuring a seamless user experience across various functionalities of the "Smart Locator" app. Below, we provide an overview of the UI interfaces catering to different user roles: users seeking health services, transportation options, fuel stations, and food; as well as administrative tasks.

3.3.1 User Interphase for User Login

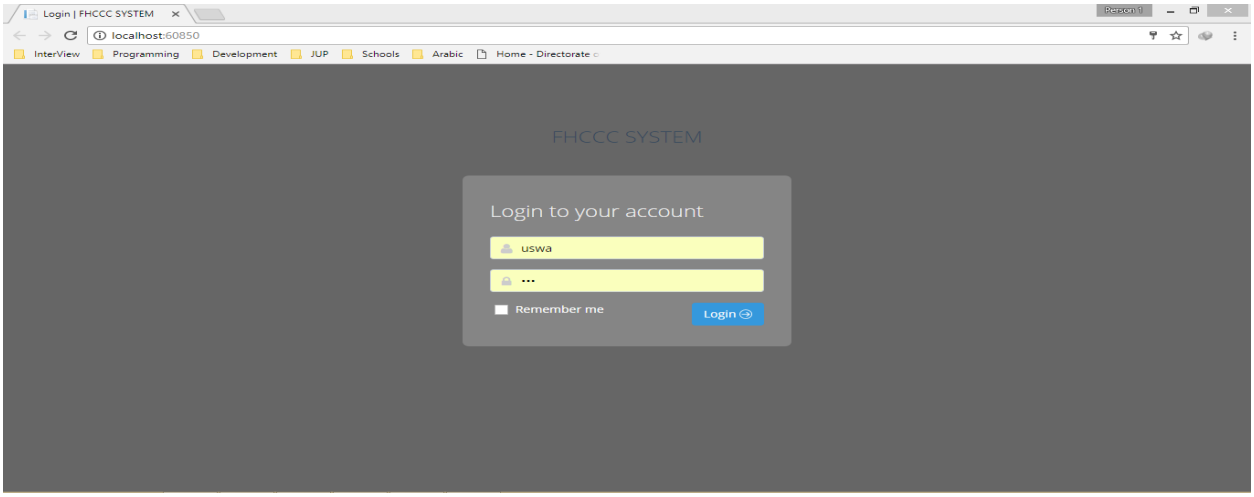


Figure 3.8 User interphase for login in to the application

The user is presented with a "Locate" button. Upon clicking, four rows appear:

Health Services: Hospital, Dentist, Pharmacies

Transportation: Bus Stop, Railway Station, Airport

Fuel Stations: Gas Stations

Other Services: Work, Money, Enjoy, Food

3.3.2 User interphase for Doctors registration

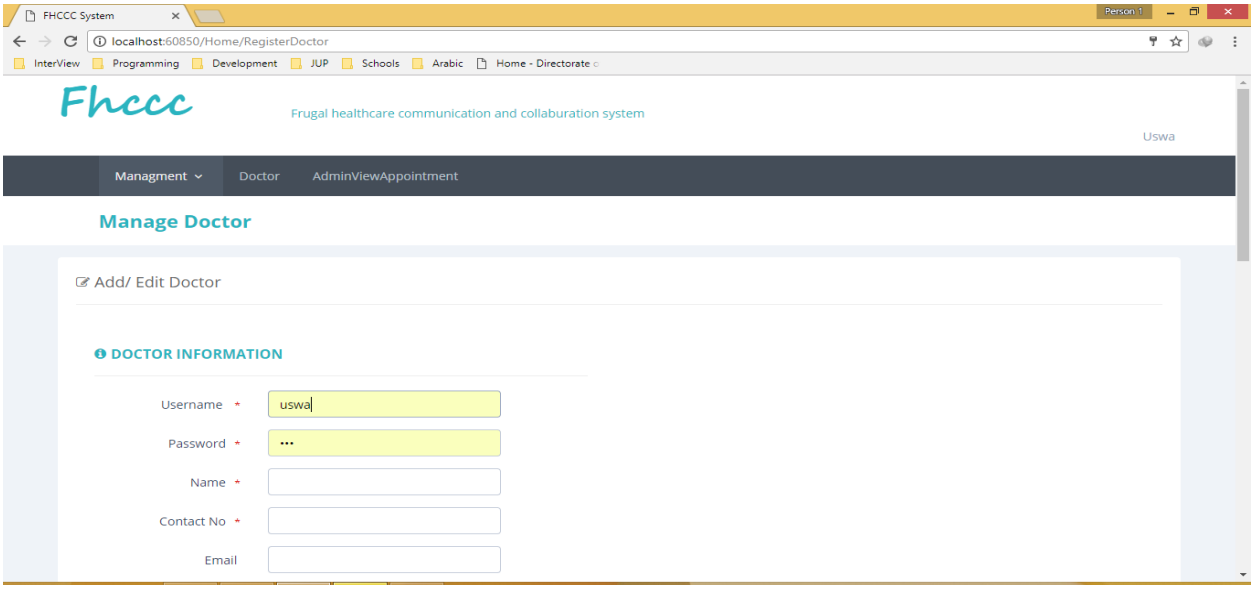


Figure 3.8 User interphase for registering Doctor

3.3.3 User interphase for Doctors and patients view appointment

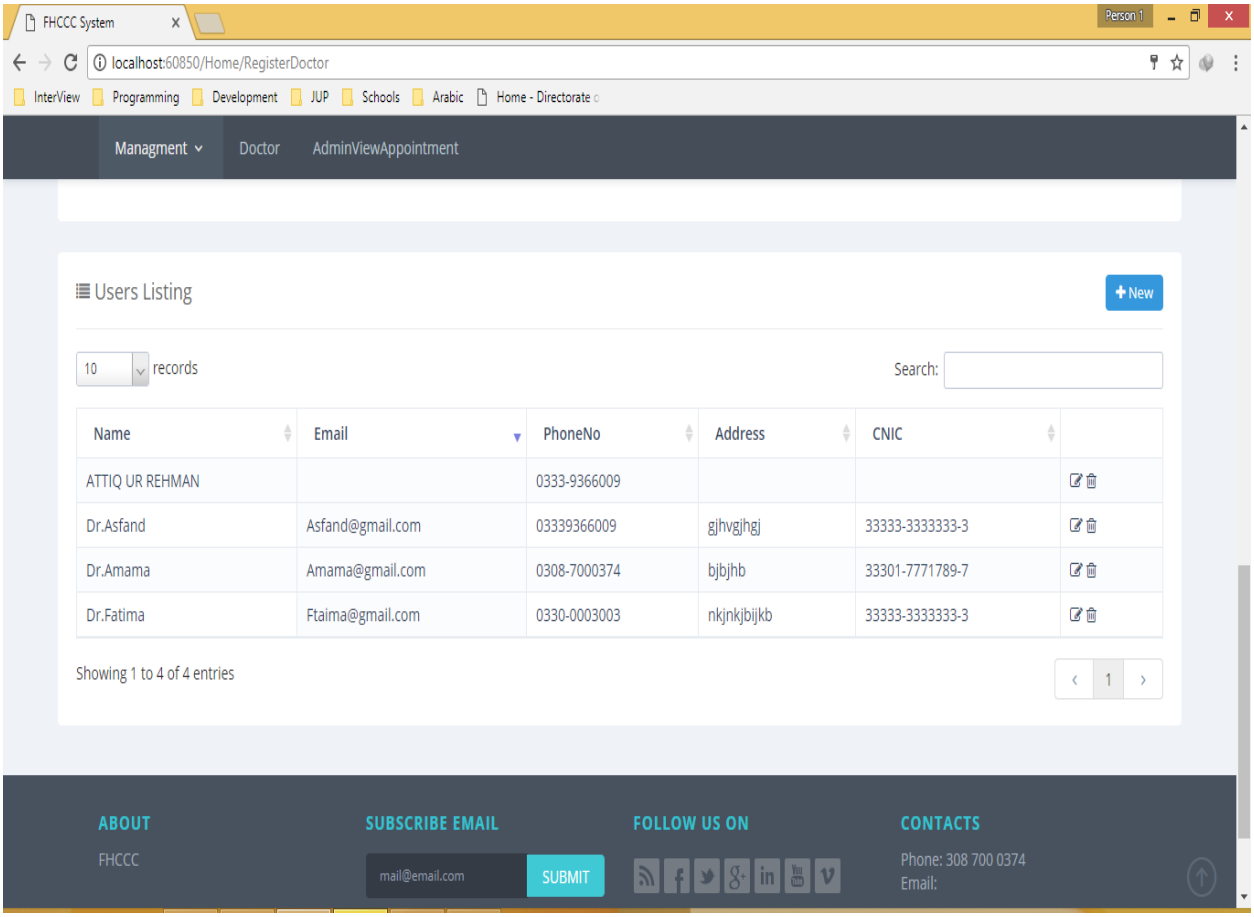


Figure 3.9 User interphase for Appointment view

This interface offers users a comprehensive overview of available food options. Users can explore a variety of food choices categorized by different types, such as cuisines, dietary preferences, and more. Each food option is accompanied by essential details, including the restaurant's name, location, and contact information. Users can also search for specific food items or restaurants within the interface. This user-friendly design facilitates efficient decision-making and allows users to plan their meals according to their preferences.

3.3.4 User interphase for Admin

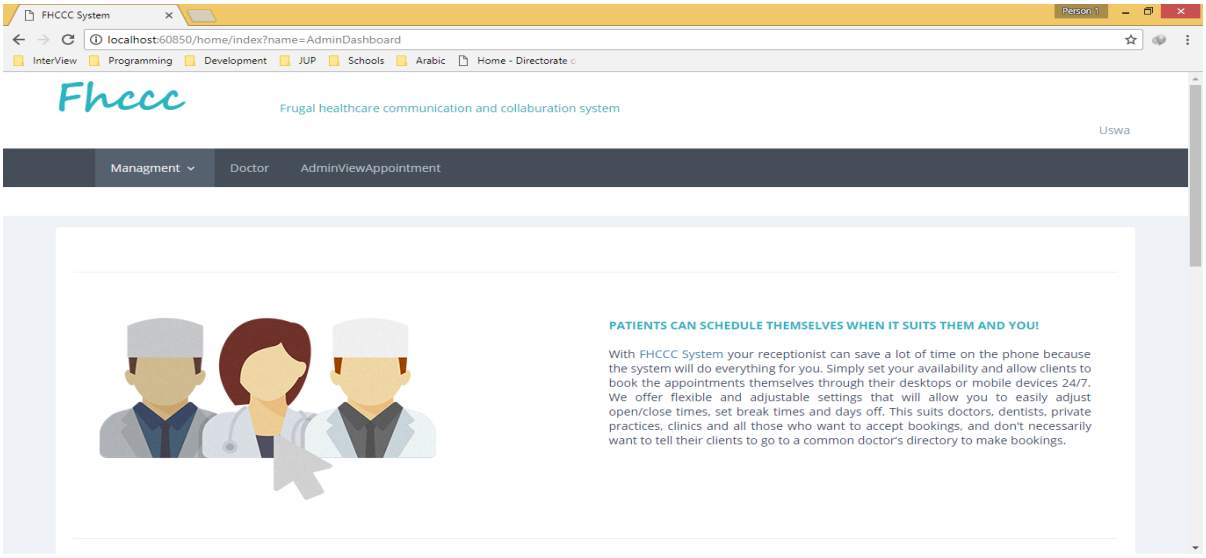


Figure 3.10 User interphase for the Admin/ admin Home page

The user interface designed for appointment scheduling serves as a streamlined hub for users to efficiently manage their engagements. Individuals can effortlessly discover available time slots for appointments, explore diverse services like health, fuel, work, and grocery options, and gain insights into the app's functionalities. This intuitive interface aims to simplify the user experience, allowing users to seamlessly interact with various features and make well-informed decisions about their appointments and activities.

3.3.5 User interphase for Patients

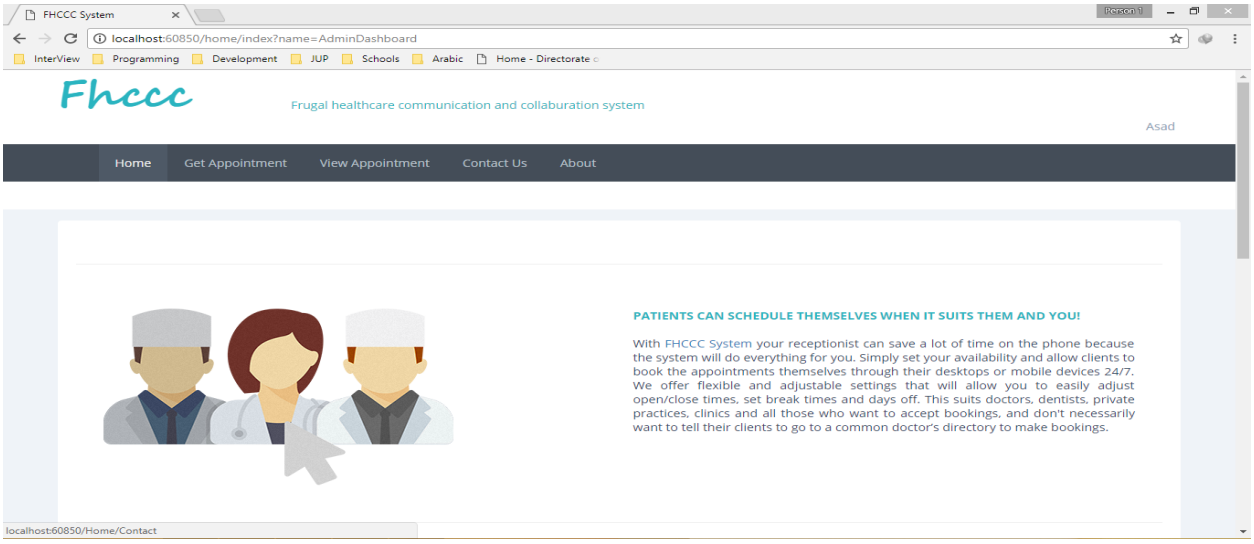


Figure 3.11 User interphase for the patient

Users can select a hospital and a doctor to explore available time slots, ensuring a streamlined and efficient appointment scheduling process.

3.3.6 User interphase for patients to view Schedules

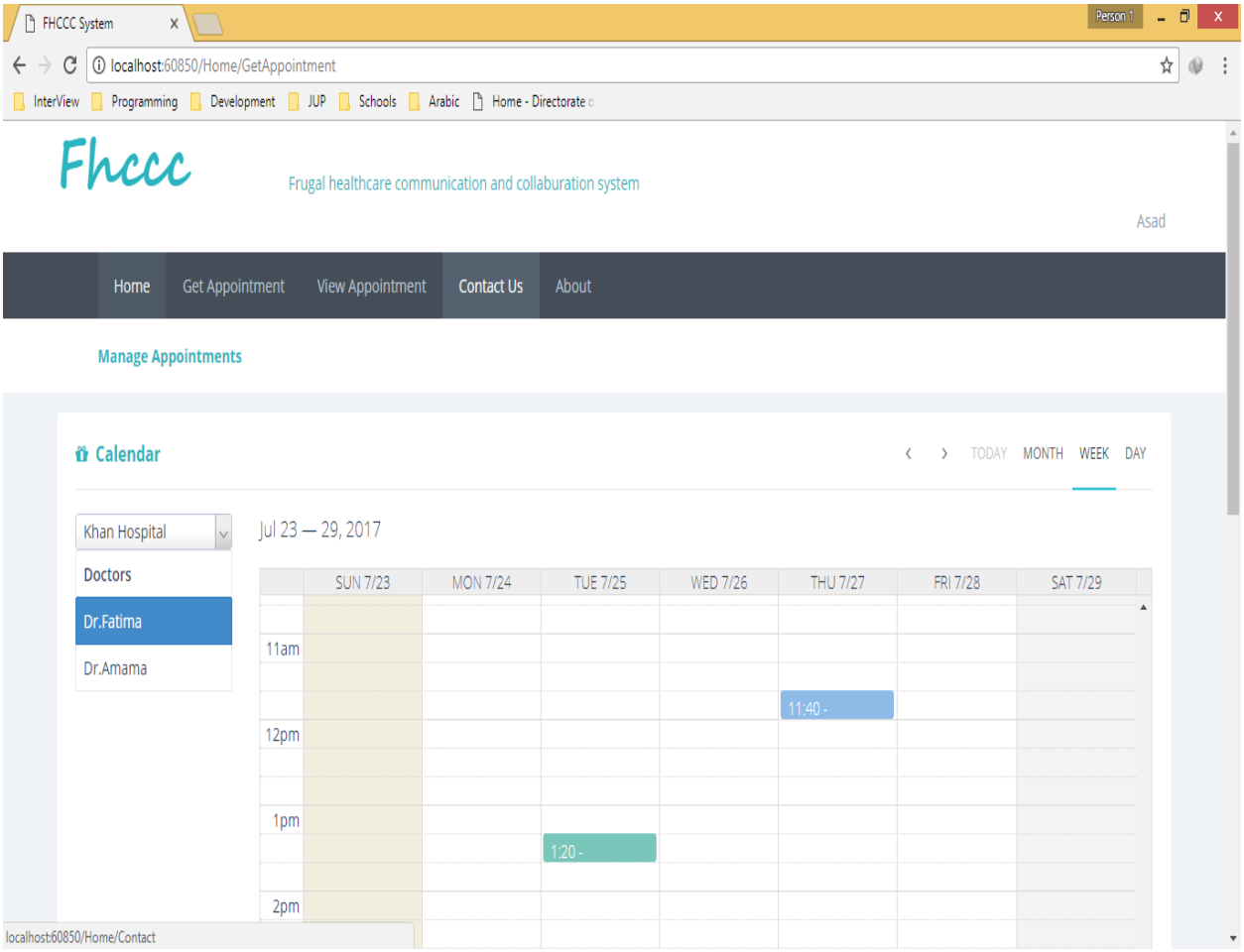


Figure 3.12 User interphase for view doctors schedule

The user interface displays essential information as users engage with the Smart Locator app. When users initiate the app by tapping the "Locate" button, they encounter a screen divided into distinct sections. The first row showcases health-related services, encompassing hospitals, dentists, and pharmacies. The second row offers options for commuting, encompassing bus stops, railway stations, and airports. The third row caters to fuel-related needs, presenting nearby gas stations.

For instance, within the health services section, users seeking appointments can select a preferred time slot. As users navigate this process, the system provides step-by-step guidance, aiding them in finalizing the appointment. Upon confirmation, the system generates a unique token number for reference and offers a clear outline of the terms and conditions associated with the successful appointment scheduling.

This approach extends across various categories, ensuring a consistent user experience as individuals interact with the app's features.

3.3.7 User interphase for Patients Get appointment

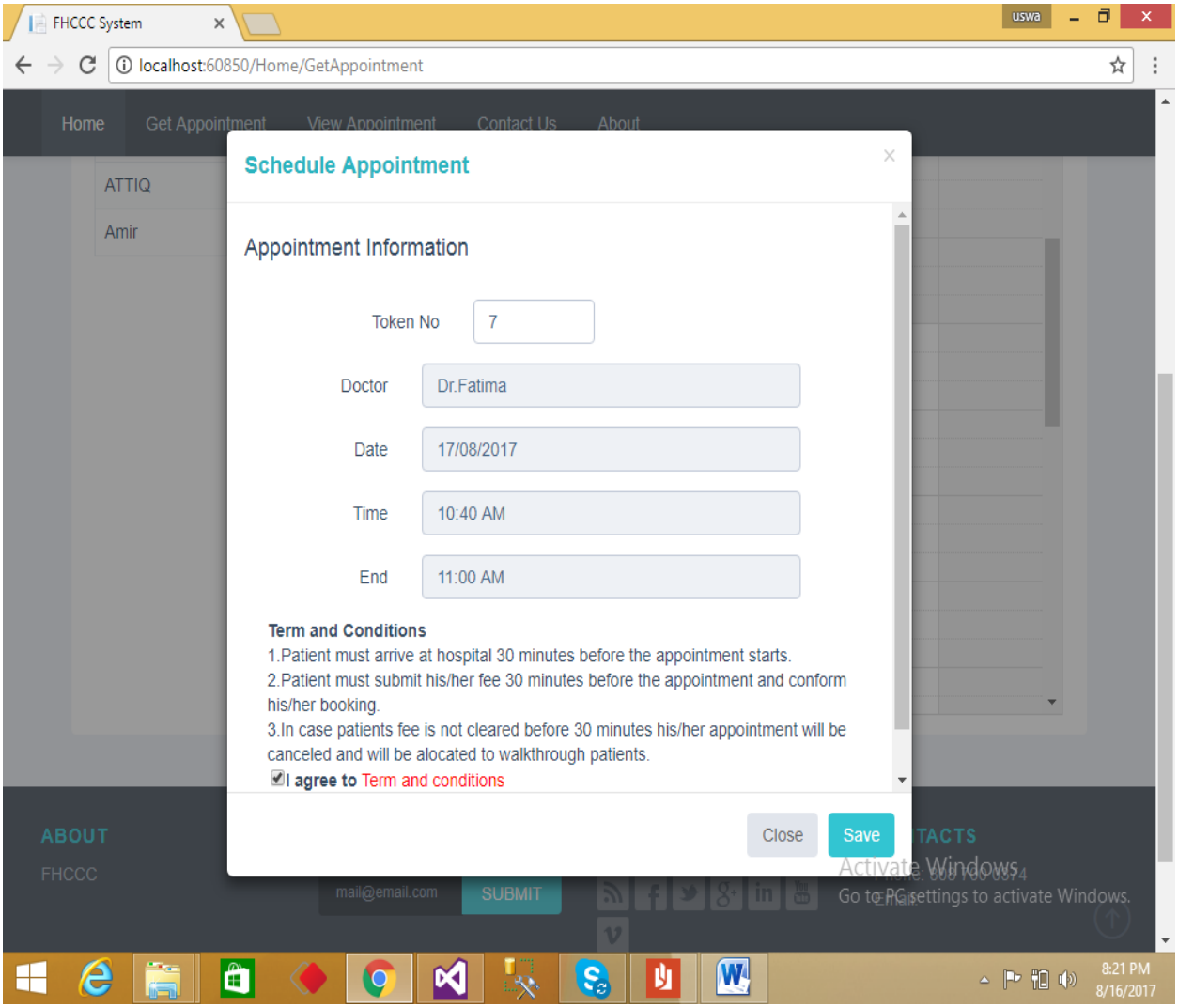


Figure 3.13 User interface for the Appointment

This interphase define the information that is required for appointment this interphase popup when patient click on an available time slot it define the token number that is auto generated when patient request appointment there are some terms and condition that the patient have to agree with if he want to conform his appointment these term and condition are that the patient must pay his dues 30 minutes before the appointment otherwise his appointment will be automatically canceled.

3.3.8 User interphase for change password

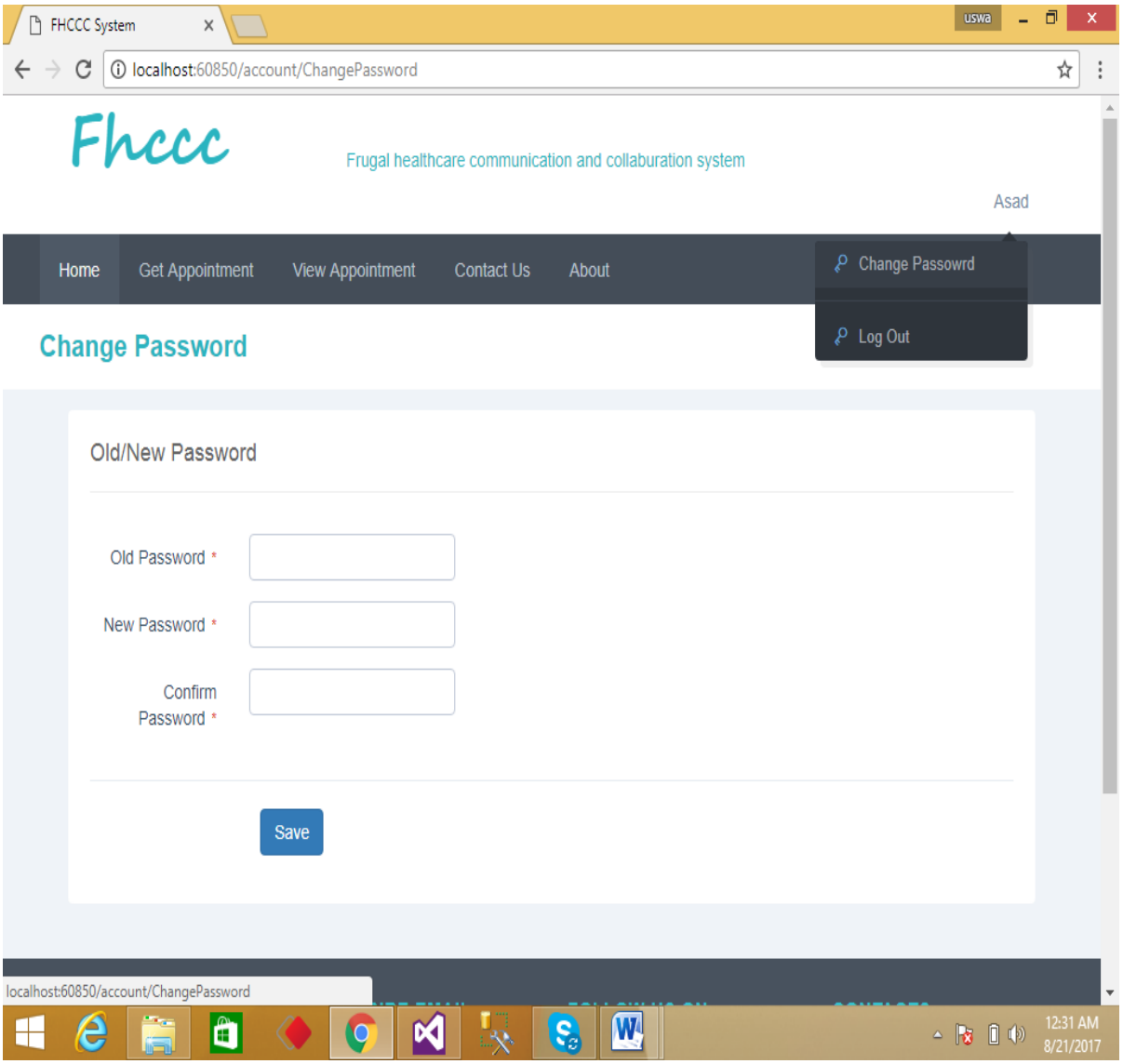


Figure 3.14 User interface for change password

the password change interface, empowering users to enhance security. By following a straightforward process, users can modify their passwords, ensuring control and safeguarding their accounts.

3.3.9 Conformation Message for Doctor

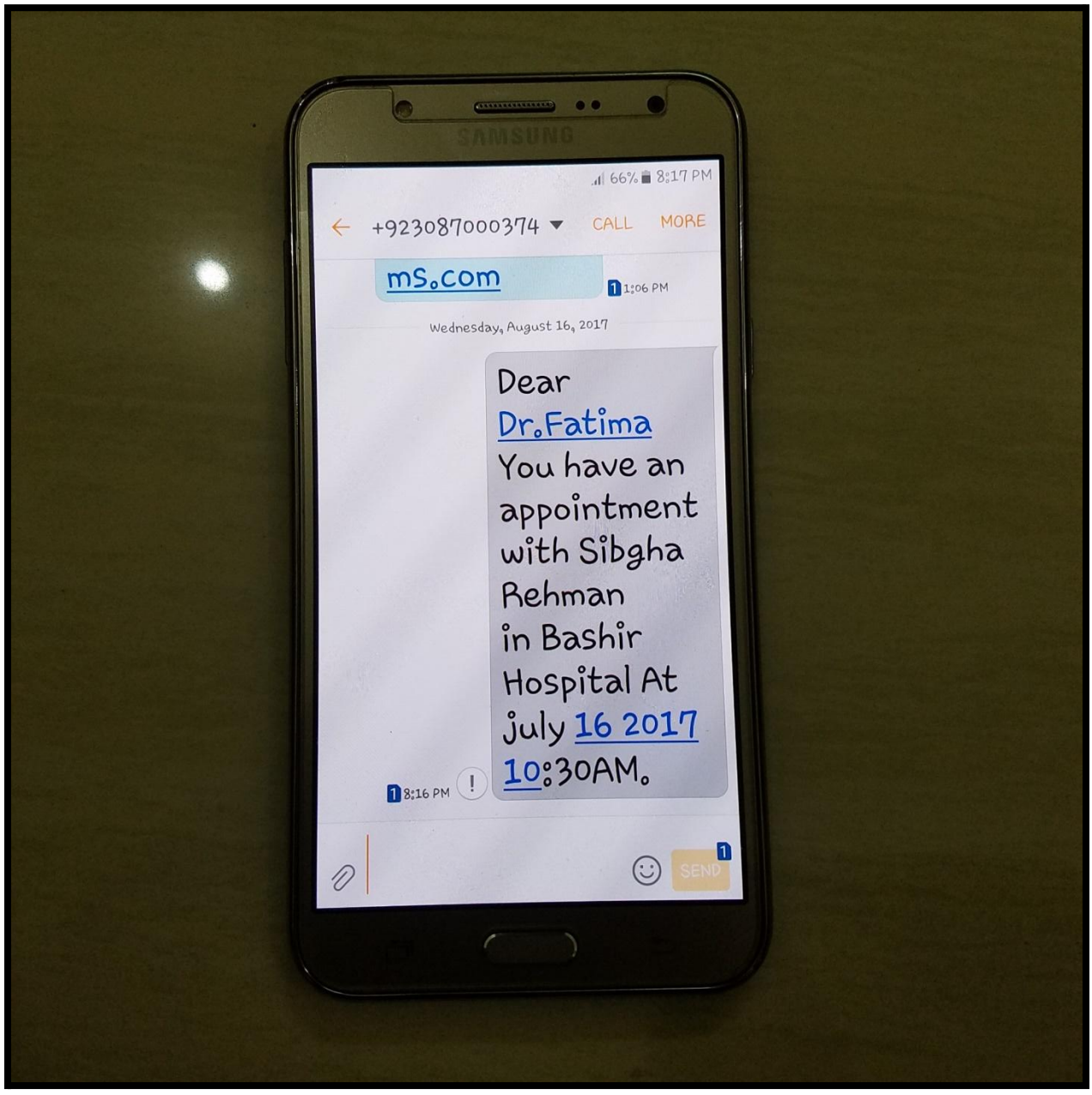


Figure 3.15 Appointment Conformation message for doctor

the confirmation messages that doctors and patients receive upon successful appointment requests. These messages communicate essential appointment details and terms, ensuring both parties are informed and prepared

3.3.10 Conformation message for Patient

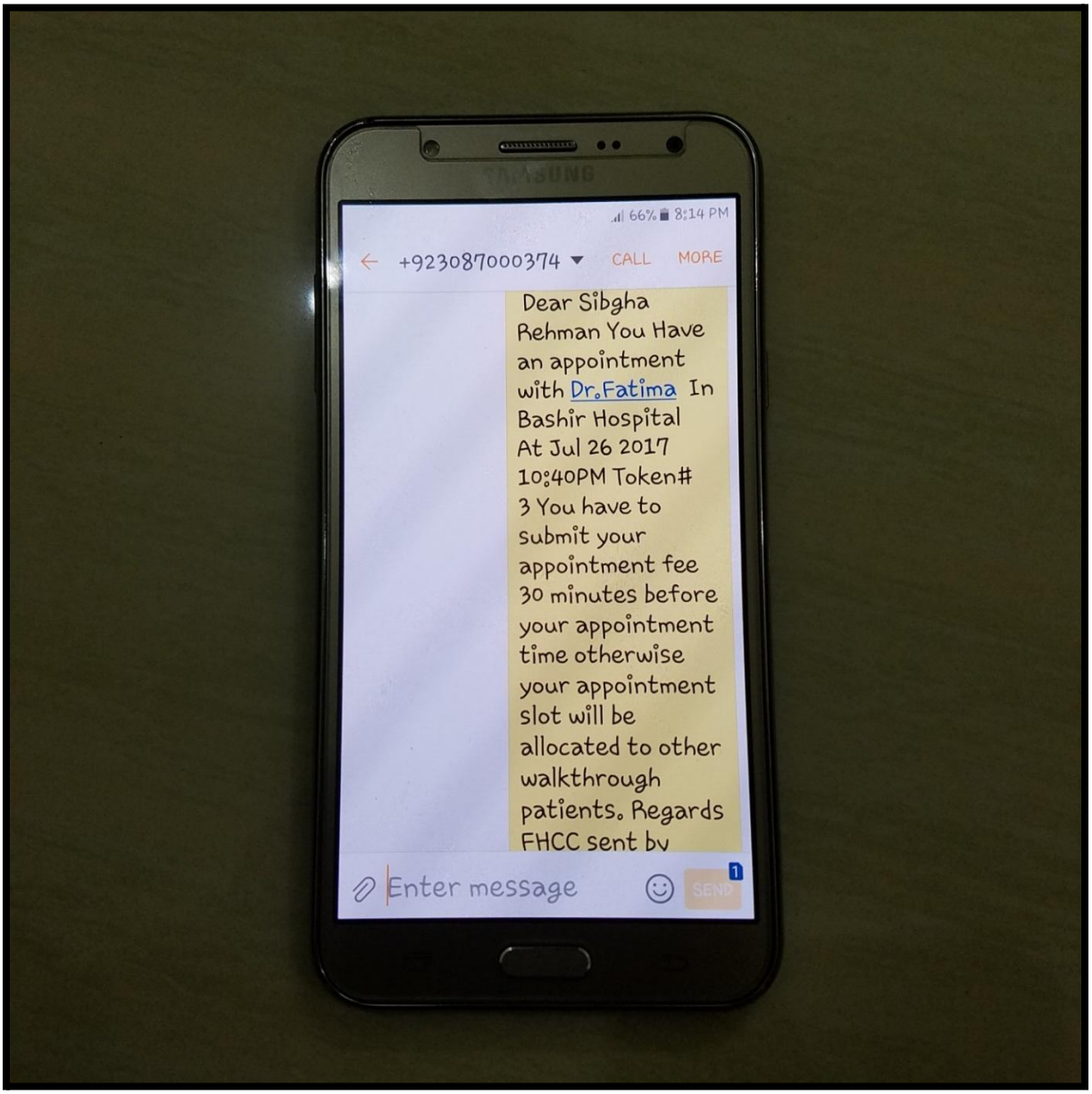


Figure 3.16 Appointment conformation message for the patient

the confirmation messages that doctors and patients receive upon successful appointment requests. These messages communicate essential appointment details and terms, ensuring both parties are informed and prepared.

Chapter 4

Implementation

4.1 Layered Architecture for the Location Allocator App

In the context of the Location Allocator App project, a layered architecture is a pivotal design choice that empowers the system to effectively handle the complexities of location-based services and adapt to changing user needs. This architectural approach segments the system into different logical tiers, each responsible for specific functionalities, thereby promoting modularity, flexibility, and maintainability.

A solution to address these challenges lies in adopting a layered architecture. This architectural style segments the system into distinct logical tiers, each responsible for specific functionalities. These tiers can evolve, update, and adapt relatively independently, minimizing the cascading effects of change and ensuring that alterations in one area do not disrupt the entire system. This approach enhances the system's resilience and responsiveness, critical factors in accommodating change.

. For example:

- A solution to address these challenges lies in adopting a layered architecture. This architectural style segments the system into distinct logical tiers, each responsible for specific functionalities. These tiers can evolve, update, and adapt relatively independently, minimizing the cascading effects of change and ensuring that alterations in one area do not disrupt the entire system. This approach enhances the system's resilience and responsiveness, critical factors in accommodating change.
- Application Layer: At the base of the architecture lies the Application Layer. This layer houses the actual applications responsible for providing services such as locating health facilities, transportation options, food outlets, financial institutions, and recreational spots. While these applications are

crucial to the system's functioning, they tend to change at a slower pace due to their complexity.

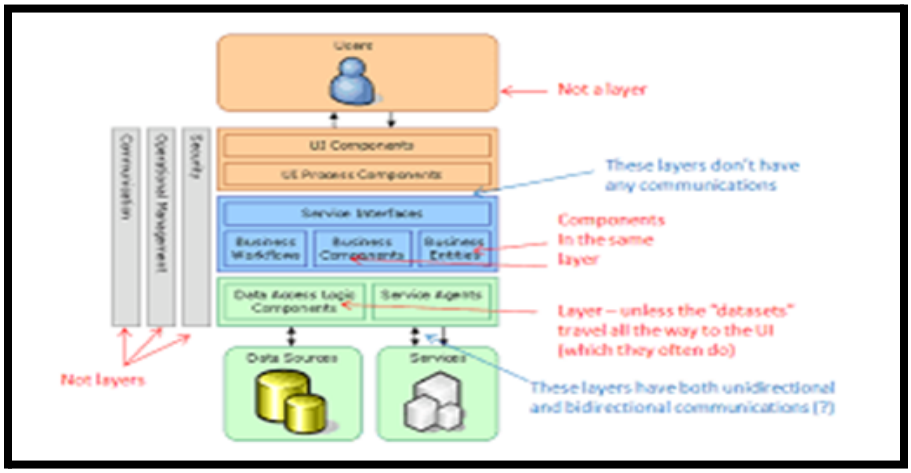


Figure 4.1 interaction of Layered architecture in Special location Allocator

Individual proclivities and environments.

- Services and Events Layer: Positioned above the Application Layer, the Services and Events Layer serves as an interface between the applications and the business domain. This layer externalizes application capabilities through services (such as APIs) and events. It aligns more closely with business needs and can evolve more swiftly. Changes to services and events have a more controlled impact on the overall system.

Processes, Rules, and Insights Layer: This layer is closely tied to the business domain and encapsulates processes, rules, and analytical insights. It utilizes the services and events provided by the lower layer to support the business value chain. Tools like business process engines, rules engines, and event processing engines are used to implement functionalities at this layer. This

separation ensures that business logic can evolve independently from the application layer, promoting agility.

The accompanying class diagram (Figure 4.3) visually represents the layered architecture within the context of the Location Allocator App. The diagram demonstrates the segregation of components into distinct layers, highlighting the relationships and interactions between them. This separation not only promotes stability but also empowers the system to adapt seamlessly to evolving requirements.

In conclusion, the adoption of a layered architecture in the Location Allocator App project is a strategic choice that empowers the system to thrive amidst change. By compartmentalizing functionalities into distinct layers, the architecture fosters adaptability, minimizes the impact of alterations, and enables the system to respond effectively to evolving user needs and industry trends.

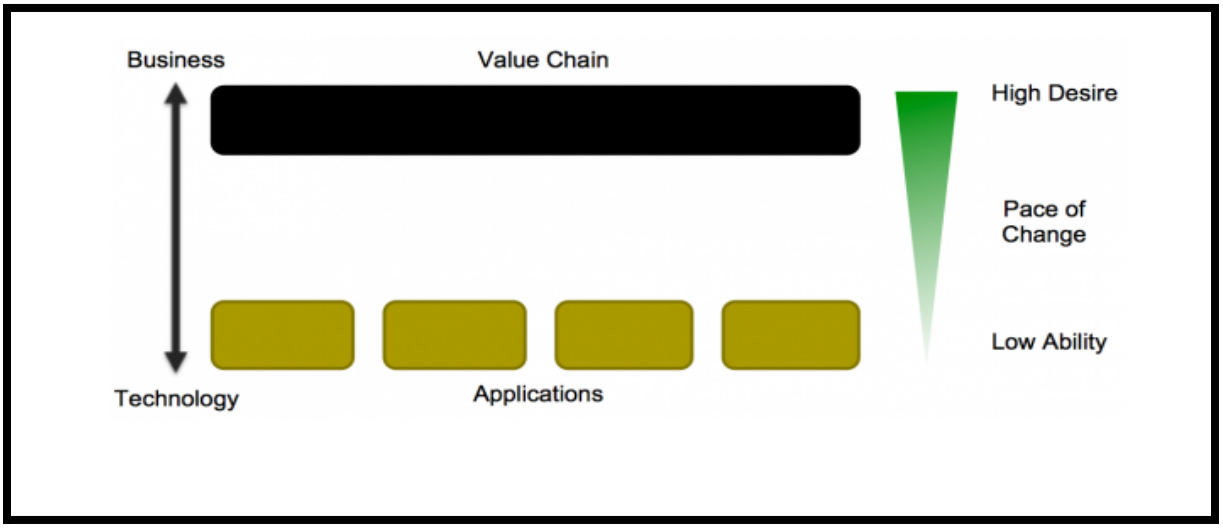


Figure 4.2 Pace of change in Special location Allocator

In the construction of the Location Allocator App's architecture, a layered paradigm emerges as a strategic framework for effectively managing the intricate landscape of location-based services. This architectural model segments the system into distinct tiers, each designated with specific responsibilities, fostering modularity, flexibility, and ease of maintenance.

Domain Logic Layer: The Domain Logic Layer forms a tightly-knit connection with the business domain, encapsulating pivotal processes, rules, and analytical insights. It leverages the services and events provided by the layer beneath to bolster the entire business value chain. Equipped with tools like business process engines, rules engines, and event processing engines, this layer executes intricate logic necessary to deliver accurate and pertinent location-based information. The segregation of the Domain Logic Layer from the Application Layer ensures that modifications in business logic can proceed independently, heightening the system's agility.

Application Layer: At the core of the architecture resides the Application Layer. This layer serves as the foundation for housing the applications that provide diverse services to users. Specifically tailored to cater to users seeking health facilities, transportation options, dining choices, financial institutions, and recreational destinations, these applications offer crucial functionality. Due to the intricate nature of their services, changes in this layer might occur at a slower pace compared to other layers. The Application Layer acts as the entry point for users, offering an intuitive and user-friendly interface to access various location-related services

Services and Integration Layer: Positioned above the Application Layer is the Services and Integration Layer. This layer acts as an intermediary between the application functionalities and the business domain. By externalizing application capabilities through services (APIs) and events, this layer ensures a streamlined interaction between the applications and the underlying system. Services provided in this layer enable seamless communication between

different parts of the application. Changes to services and integration mechanisms can be implemented more rapidly, helping the system to adapt to new requirements and technologies.

Processes and Business Logic Layer: The Processes and Business Logic Layer plays a pivotal role closely tied to the business domain. It encompasses crucial processes, rules, and analytical insights that support the business value chain. Leveraging the services and events offered by the layer below, this tier executes intricate business logic. Tools like business process engines, rules engines, and event processing engines facilitate the implementation of functionalities within this layer. This segregation ensures that business logic can evolve independently of the application layer, promoting agility.

Illustrated in the accompanying class diagram (Figure X.X), the layered architecture for the Location Allocator App project visually represents the separation of components into distinct layers, showcasing their relationships and interactions. This design not only ensures stability but also empowers the system to smoothly adapt to evolving requirements.

In conclusion, the adoption of a layered architecture in the Location Allocator App project is a strategic choice that empowers the system to effectively address the challenges posed by location-based services. By compartmentalizing functionalities into separate layers, the architecture fosters adaptability, mitigates the impact of changes, and enables the system to respond efficiently to evolving user needs and industry trends. This layered structure ensures the application's resilience and readiness to navigate through changes seamlessly.

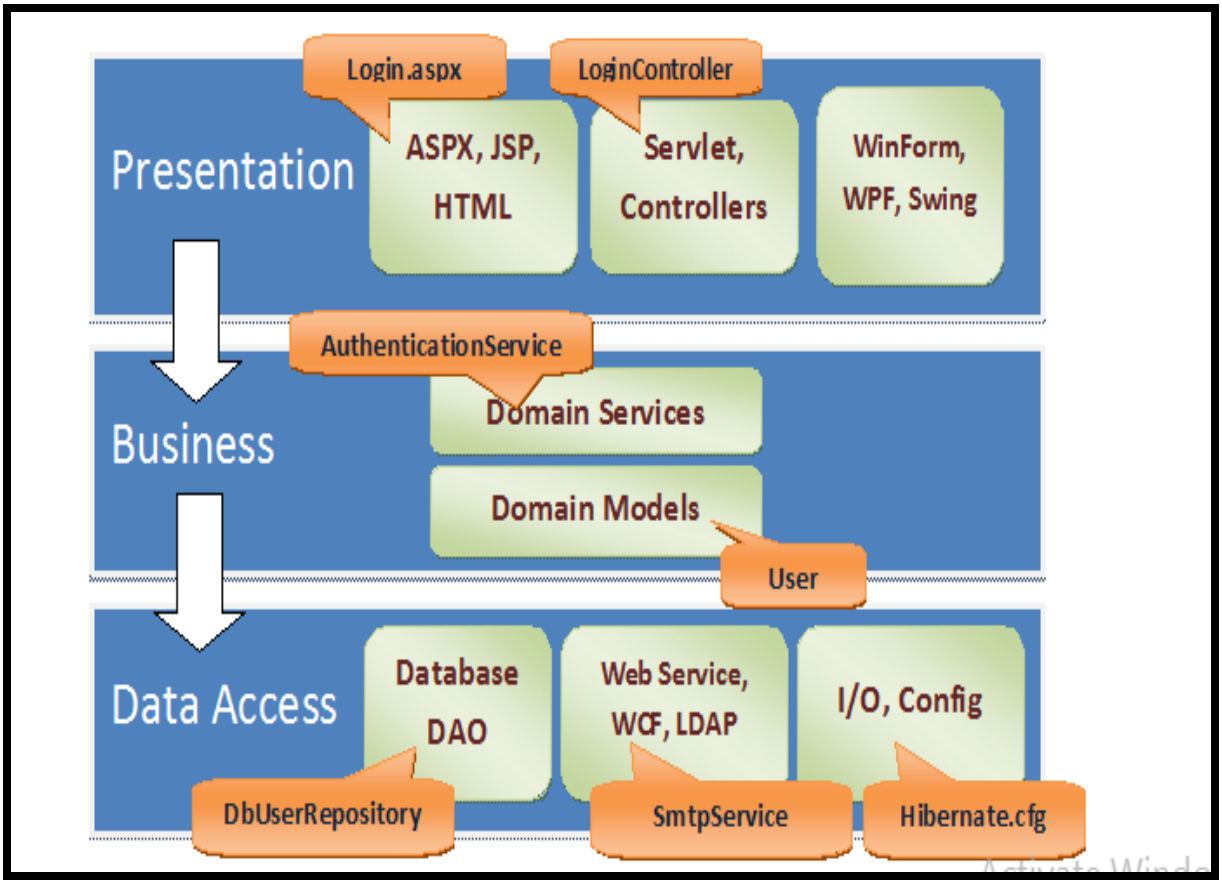


Figure 4.3 Layered architecture in LOcation Allocator App

Creating Frugal healthcare collaboration system MVC application

In the context of your "Location Allocator App," you're looking to create an MVC application that allows users to find various types of locations such as healthcare facilities, transportation hubs, food places, banks, entertainment venues, and more. The main idea of using the MVC architecture is to separate your application into distinct components, each with its own responsibilities:

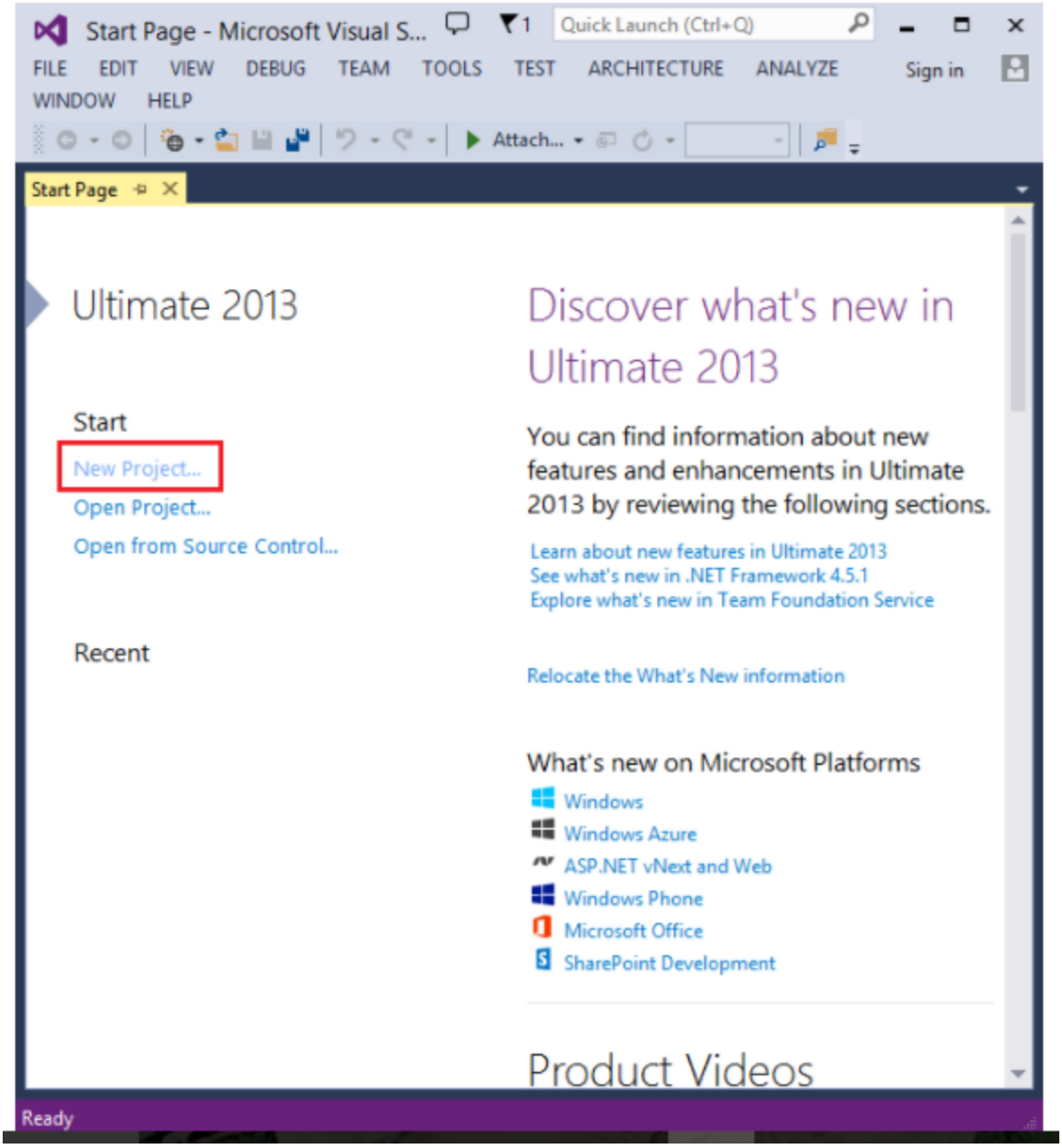


Figure 4.4 Creating MVC application For Frugal healthcare collaboration system

Model:

- Create model classes to represent different types of locations, such as HealthcareLocation, TransportationLocation, FoodLocation, etc.

- Define attributes for each model class, such as name, address, category, and any other relevant information.

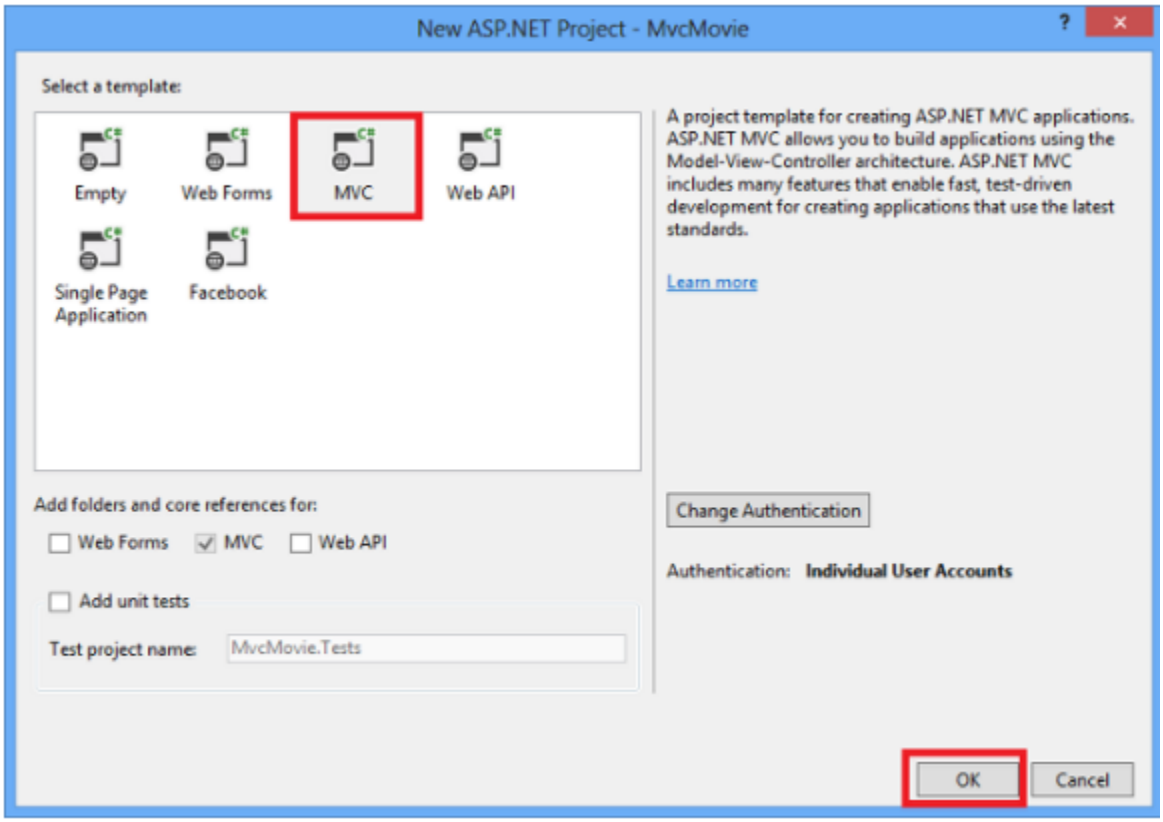


Figure 4.5 Selecting MVC Application for Frugal healthcare collaboration system

View:

- Create views to display the different categories of locations and their details. For example, you could have separate views for healthcare, transportation, food, entertainment, etc.
- Use HTML, CSS, and JavaScript (React) to structure and style your views.

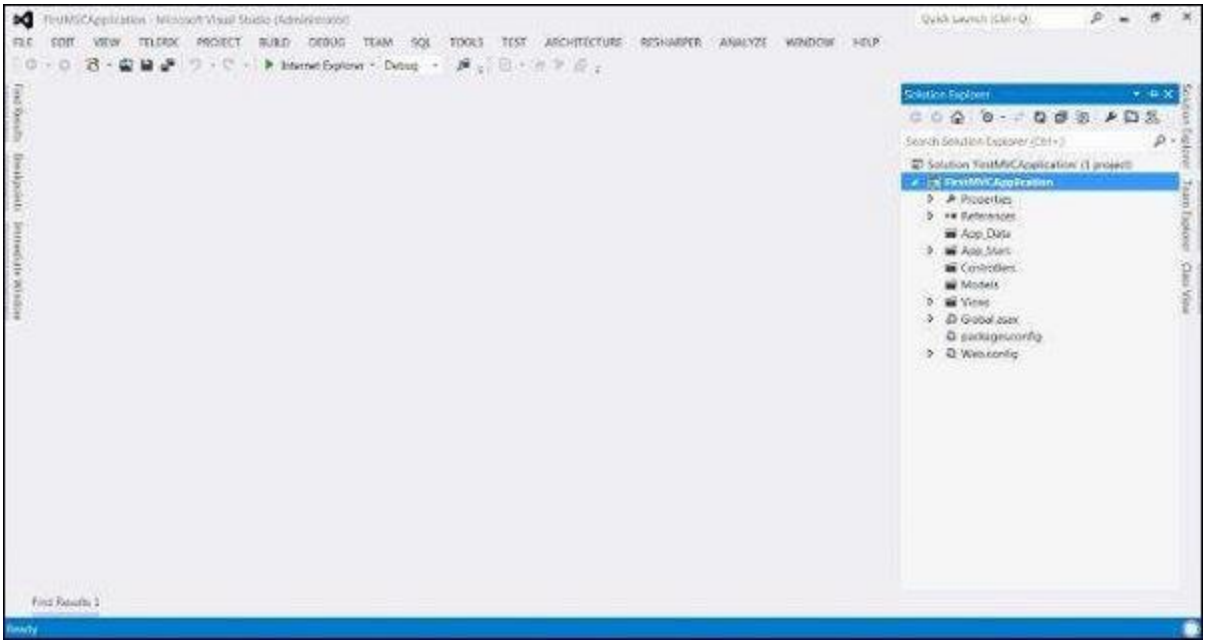


Figure 4.6 Application Created in Visual studio's

Controller:

- Implement controllers to handle user interactions and route requests to the appropriate views.
- In your controllers, you can define methods to retrieve data from the models and pass it to the views

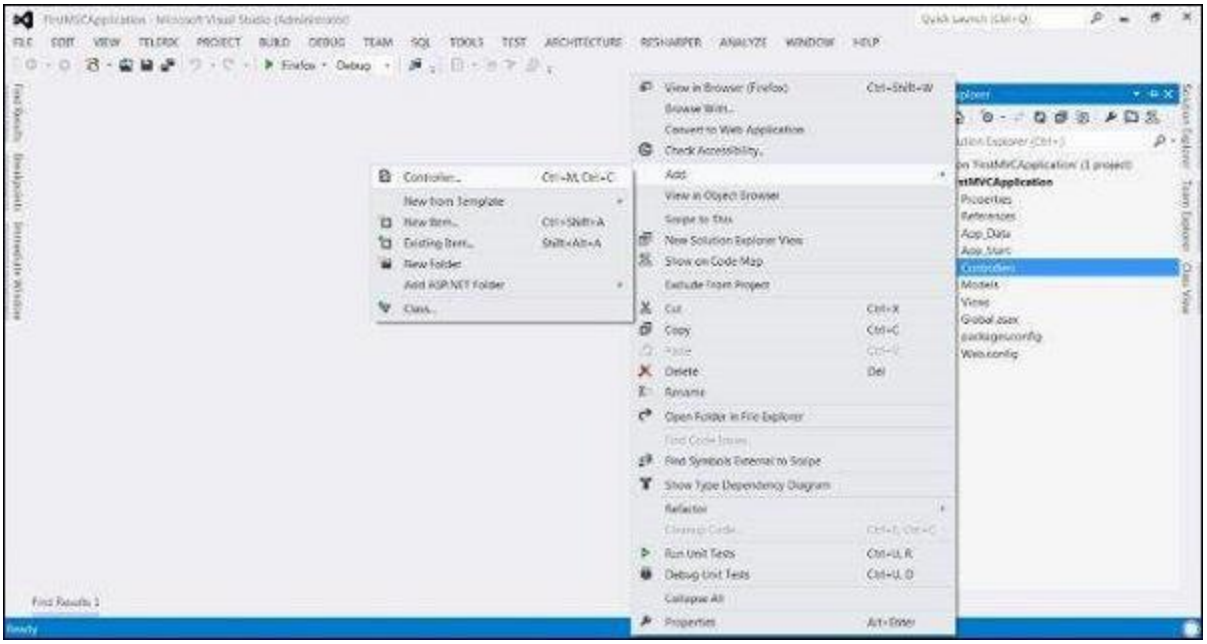


Figure 4.6 Creating Controller In application

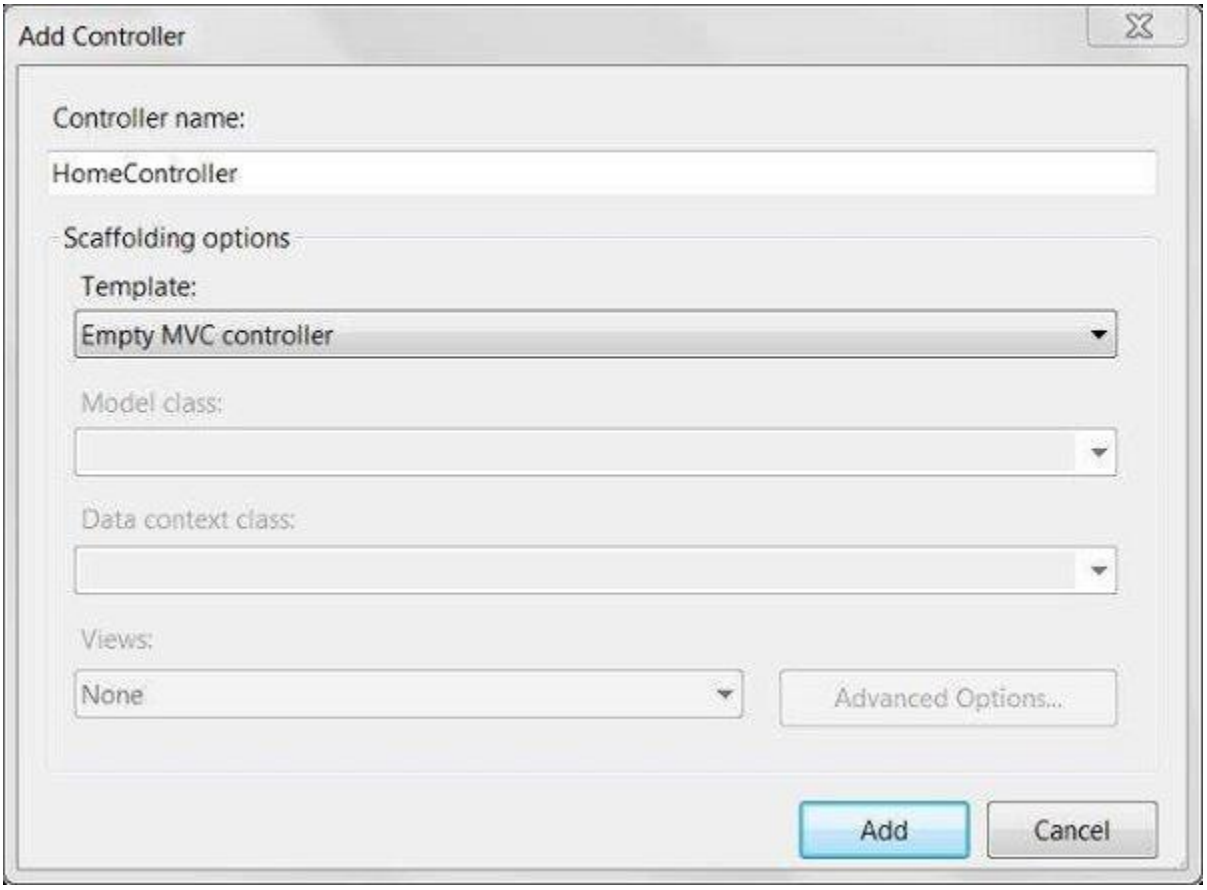


Figure 4.7 Adding Empty Controller of MVC

Controller: Handles user input, interacts with the models, and updates the views accordingly. In your case, the controllers would manage the logic for locating different types of places and passing the relevant data to the views.

```
using System;
using System.Web.Mvc;


namespace FirstMVCApplication.Controllers {


  public class HomeController : Controller {


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

It's important to note that the provided code snippets and steps seem to be taken from a different tutorial related to creating an MVC application for a different purpose (Health). While the fundamental concepts of MVC apply, you'll need to adapt the code and structure to fit the requirements of your "Location Allocator App.

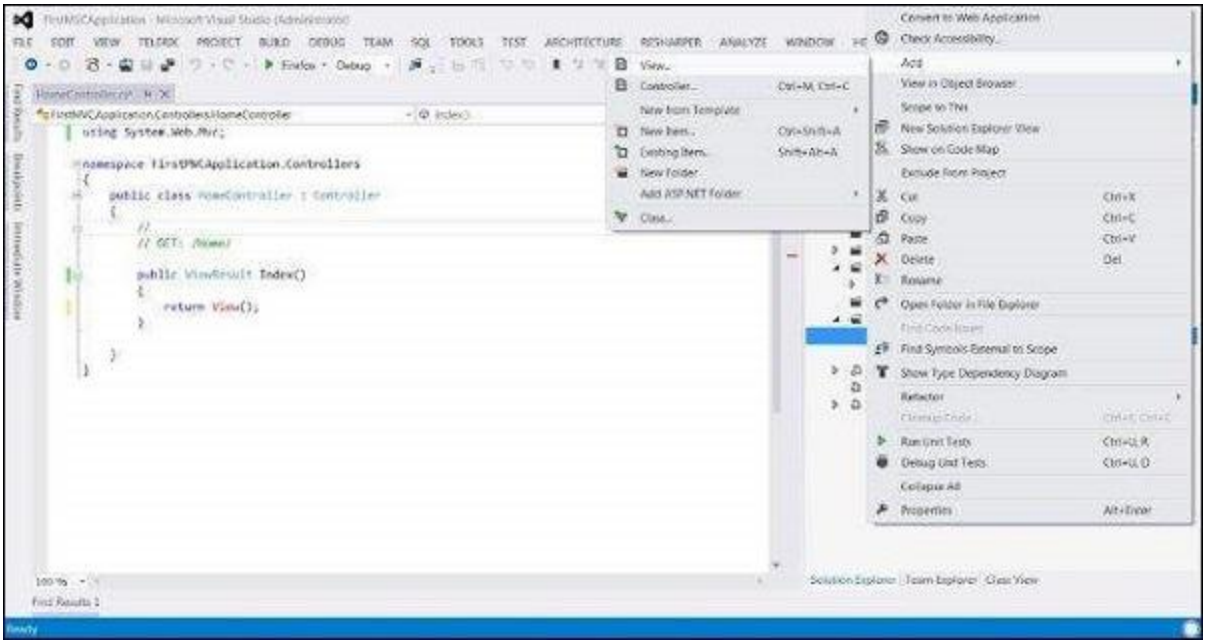


Figure 4.8 Adding View To the controller in MVC

Name the new View as Index and View Engine as Razor (SCHTML). Click Add.

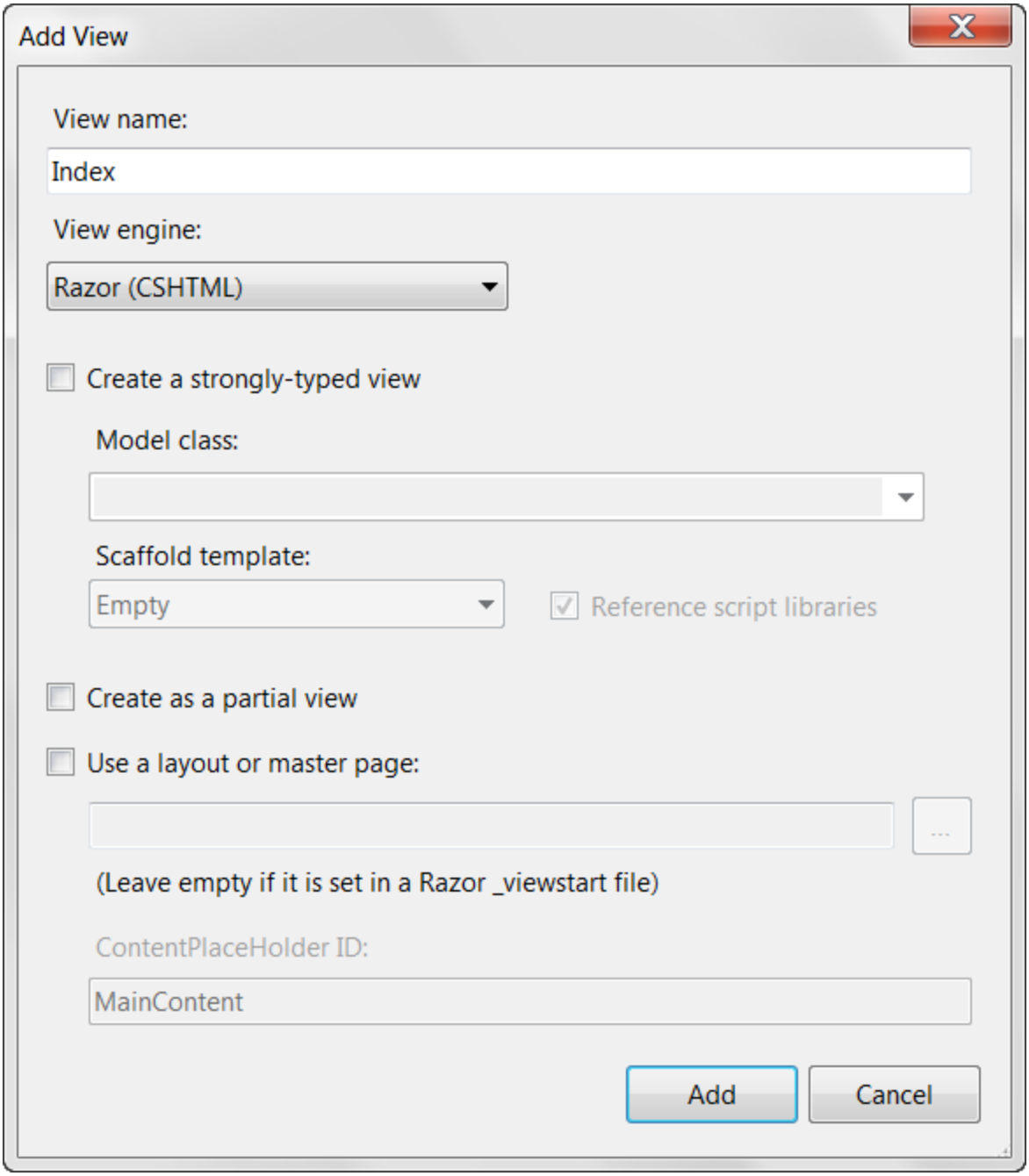


Figure 4.9 Adding Razor View In MVC

This will add a new **cshtml** file inside Views/Home folder with the following code −

```
@{
  Layout = null;
}
```

```
<html>
  <head>
    <meta name = "viewport" content = "width = device-width" />
    <title>Index</title>
  </head>

  <body>
    <div>

    </div>
  </body>
</html>
```

Modify the above View's body content with the following code −

```
<body>
  <div>
    Welcome to My First MVC Application (<b>From Index View</b>)
  </div>
</body>
```

Remember that this is a simplified example, and your actual implementation might involve more complexity, such as error handling, data validation, authentication, and more. Additionally, you need to adapt the code to the structure and requirements of your "Location Allocator App."

Figure 4.10 MVC Application In running Form

Incorporate these concepts of MVC into your project to create a structured and maintainable application that separates concerns, making it easier to manage and scale as your app grows.

4.2 Code

The code documentation provided here outlines key components and concepts used in the Location Allocator App project. It covers various sections such as JavaScript code, C# code, HTML code, database connections, API controllers, and class definitions.

4.2.1 Java Script Code

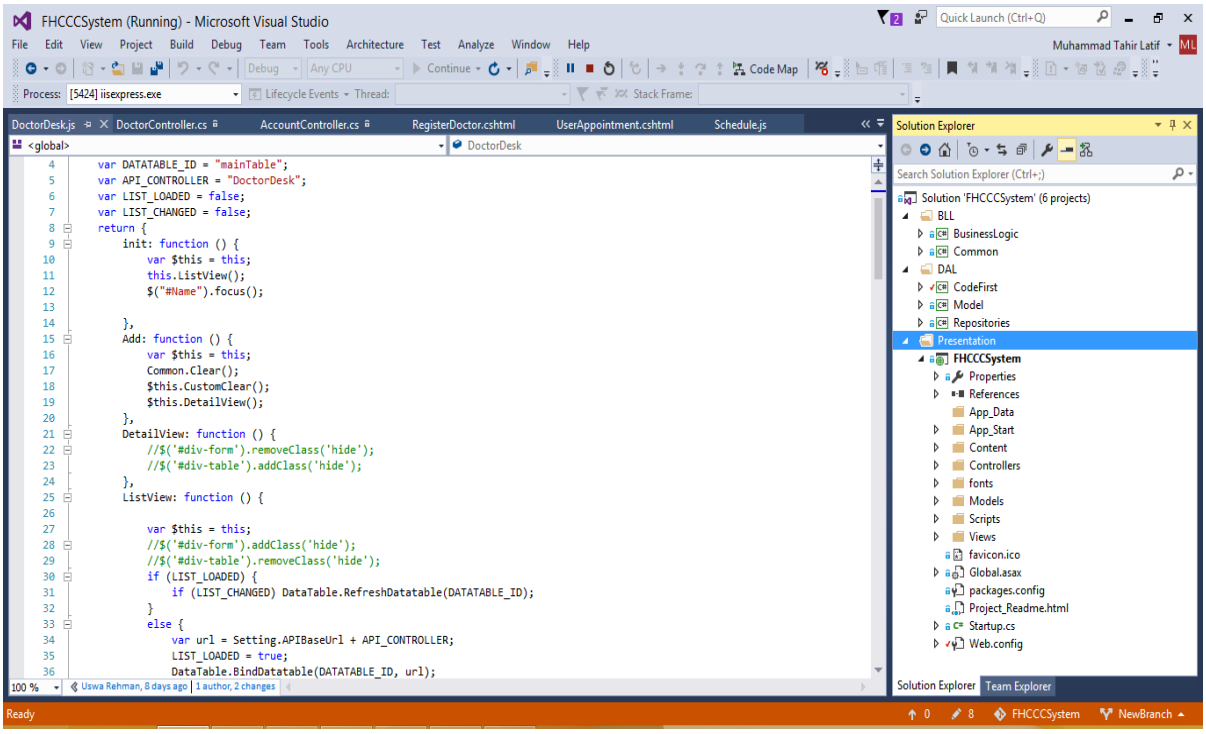


Figure 4.11 Java script Code for getting appointment from database

JavaScript (JS) is a dynamic, high-level programming language used to enhance webpage interactivity. It works alongside HTML and CSS to create a responsive user experience. The provided code illustrates how appointments are retrieved from the database.

The JavaScript code retrieves appointment data from the database, enhancing the page's responsiveness and user interaction.

4.2.2 C# Code

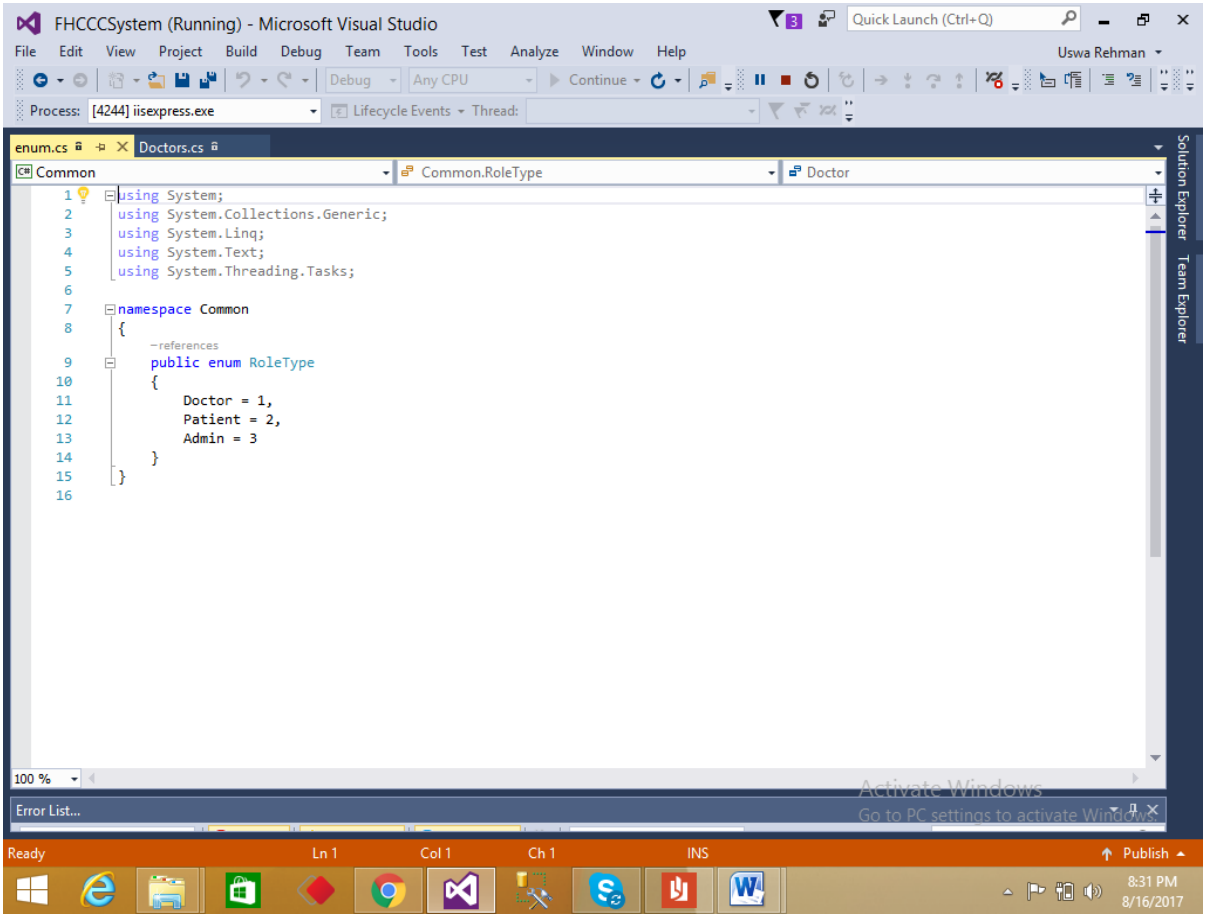


Figure 4.12 Code for Enum for in frugal healthcare collaboration system

C# is a dynamic programming language developed by Microsoft, adept at supporting multiple programming paradigms. It enjoys widespread use across

diverse application types. The following code snippets showcase how C# contributes to the project.

The utilization of an enumeration (enum) proves invaluable in delineating user roles within the LOcation Allocator app. This enhances the management of user roles and authentication protocols

4.2.3 C# Code for Controller in MVC

The foundational framework of Model-View-Controller (MVC) has been judiciously adopted in this project. This architectural pattern engenders a logical separation of application components, cultivating a harmonious environment for code organization and maintainability.

Presented here is a code snippet that elucidates the meticulous handling of Health information within the LOcation Allocator app. This code encompasses the full spectrum of operations – from data addition and retrieval to updates and deletions – fostering effective hospital data management within the system.

Traditionally employed in desktop graphical user interfaces (GUIs), the Model-View-Controller (MVC) architecture has seamlessly transcended its origins to conquer the realm of web applications. This influence extends across diverse platforms, encompassing mobile devices, desktop environments, and an array of other clients. This transition has been spurred by the proliferation of its effectiveness and adaptability.

Prominent programming languages such as Java, C#, Ruby, and PHP, among others, have embraced the MVC paradigm with open arms. The software development landscape is enriched by a plethora of MVC frameworks that serve as ready-made tools for crafting web applications. These frameworks expedite the process of building dynamic and interactive web solutions, directly out of the box. The marriage of MVC architecture with these

programming languages empowers developers to streamline development workflows, foster maintainability, and amplify user experiences across a multitude of platforms.

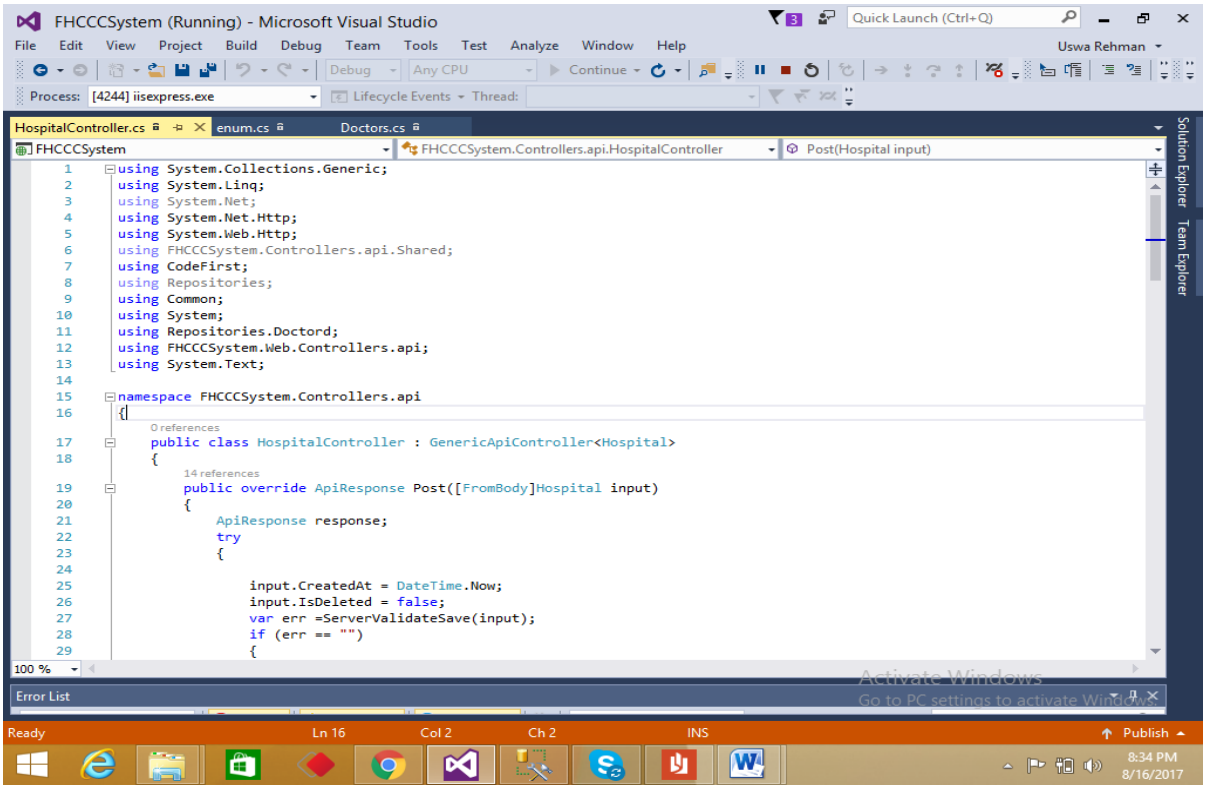


Figure 4.13 C# code for the hospital controller in frugal healthcare collaboration system

4.2.4 C# Code for Controller in MVC

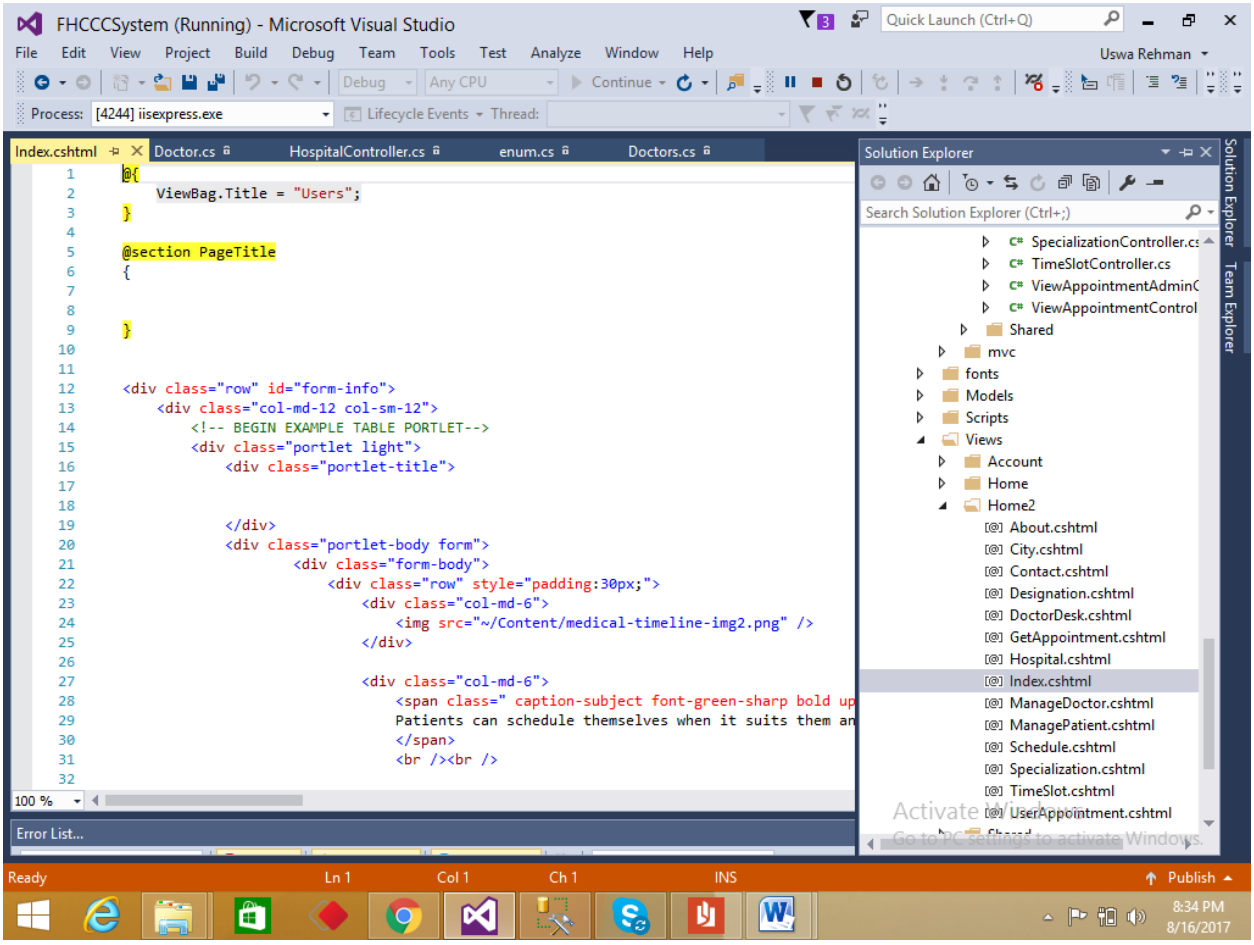


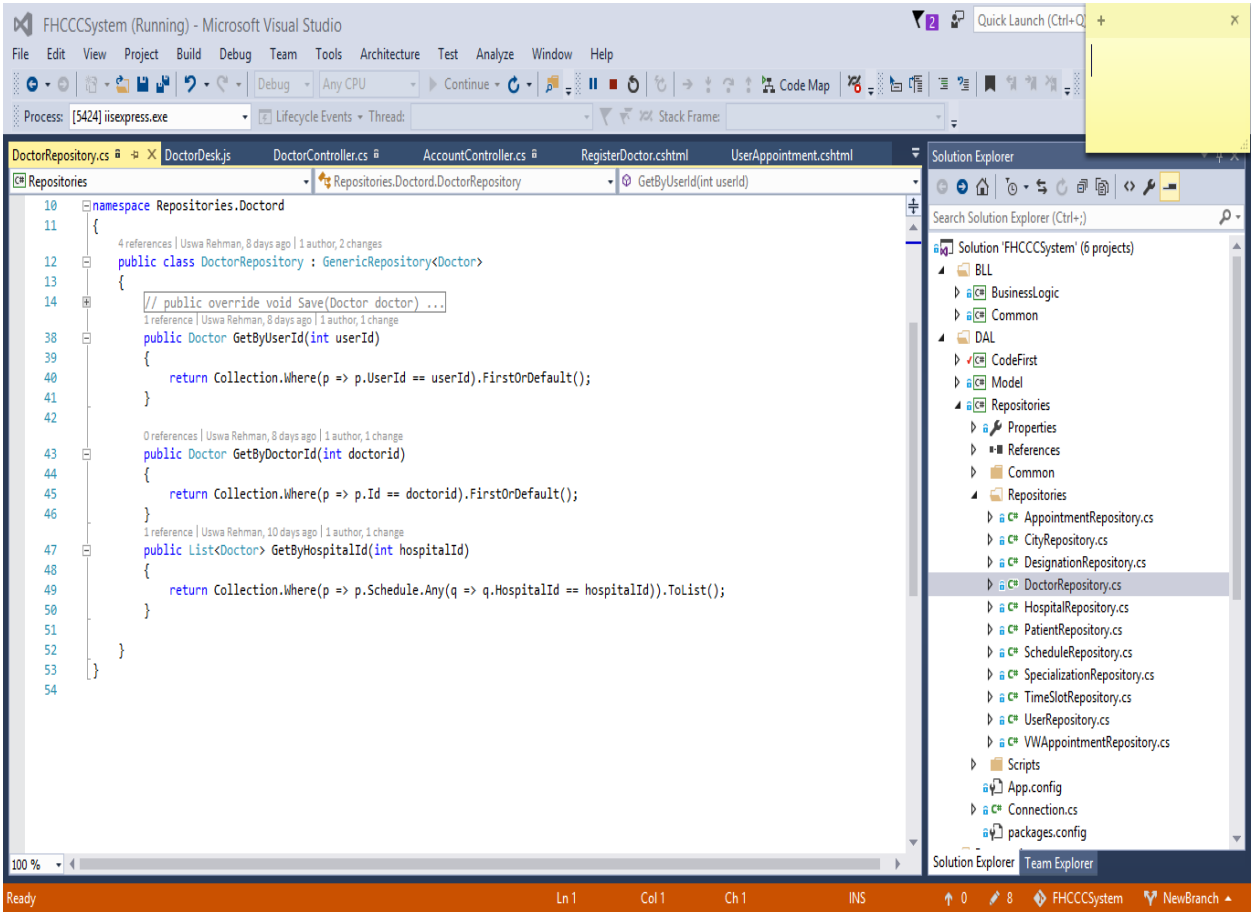
Figure 4.14 HTML code for index page in Location Allocator App

HTML (Hypertext Markup Language) plays a pivotal role in structuring and presenting content on webpages. When synergized with CSS and JavaScript, it bestows a captivating demeanor to user interfaces. The showcased HTML code centers on the index page of the Frugal Healthcare Collaboration System.

This HTML code forms the bedrock of the index page, artfully conveying information about the Frugal Healthcare Collaboration System's objectives and purpose.

4.2.5 Repositories For database connection

The facilitation of a robust Database Management System (DBMS) is instrumental in orchestrating interactions with users, applications, and data. In this project, the database connection serves as the conduit for data retrieval and manipulation.



'**Figure 4.15** Code for database connection in frugal healthcare collaboration system

This code segment establishes a steadfast connection with the database, paving the way for seamless data retrieval, updates, and queries across the system's multifaceted functionalities.

4.2.6 API Controllers

API controllers shoulder the responsibility of processing incoming HTTP requests and generating corresponding responses. Their instrumental role lies in fostering effective communication between the frontend and backend components.

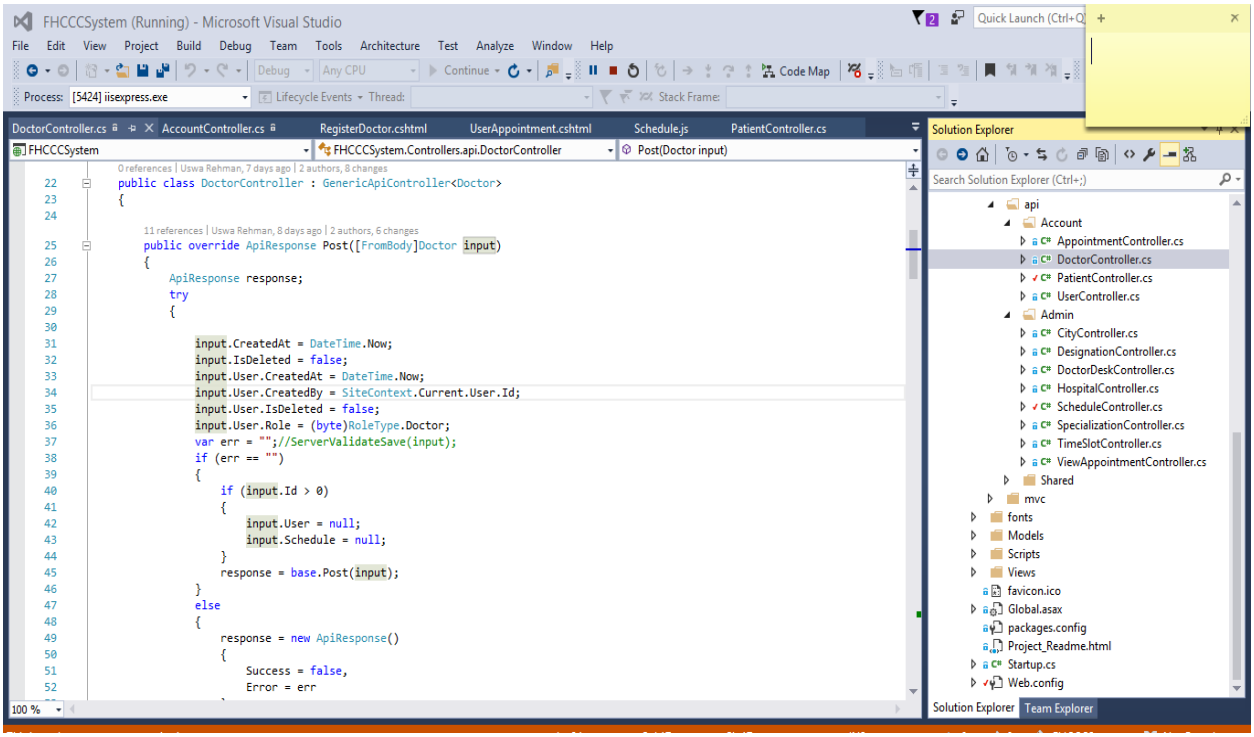


Figure 4.16 C# code for API Controller of doctor in Frugal healthcare collaboration system

The API controller assigned to doctors orchestrates the exchange of data between the frontend (JavaScript) and backend (C#) modules. It streamlines the creation and retrieval of doctor-related information, epitomizing the bridge that unites disparate components.

4.2.7 Class's code

The bedrock of object-oriented programming resides in the definition of classes – entities that encapsulate both data and behavior. This hallmark of code organization and reusability permeates this project.

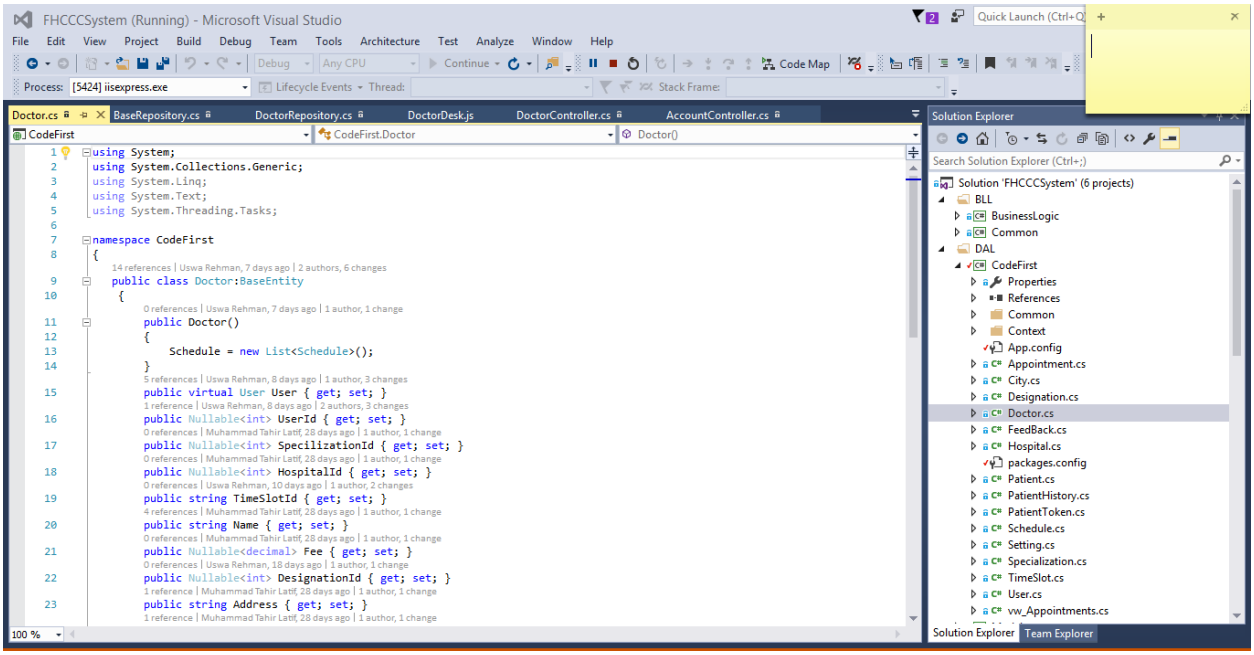


Figure 4.17 C# code for Class of doctor in Frugal healthcare collaboration system

The Doctor class emerges as a foundational cornerstone, delineating the essence of doctor-centric entities. The encapsulation of properties and methods empowers data manipulation and cements relationships with peer classes, fortifying the project's structural integrity.

4.2.8 DBContext

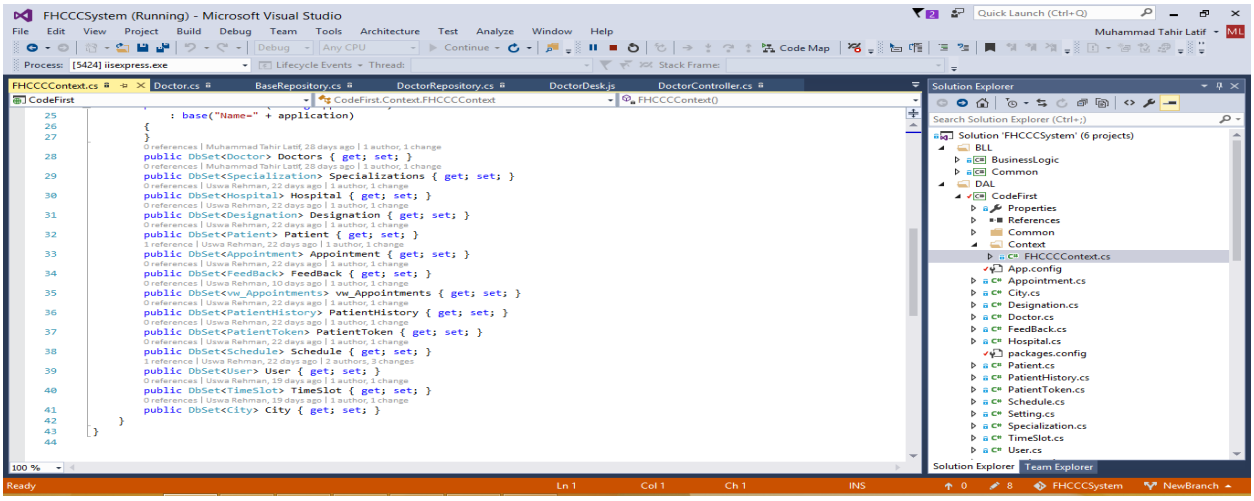


Figure 4.18 C# code for Class to connect model classes to database tables

The facilitation of a robust Database Management System (DBMS) is instrumental in orchestrating interactions with users, applications, and data. In this project, the database connection serves as the conduit for data retrieval and manipulation.

4.4 Creating SQL Database

This code segment establishes a steadfast connection with the database, paving the way for seamless data retrieval, updates, and queries across the system's multifaceted functionalities.

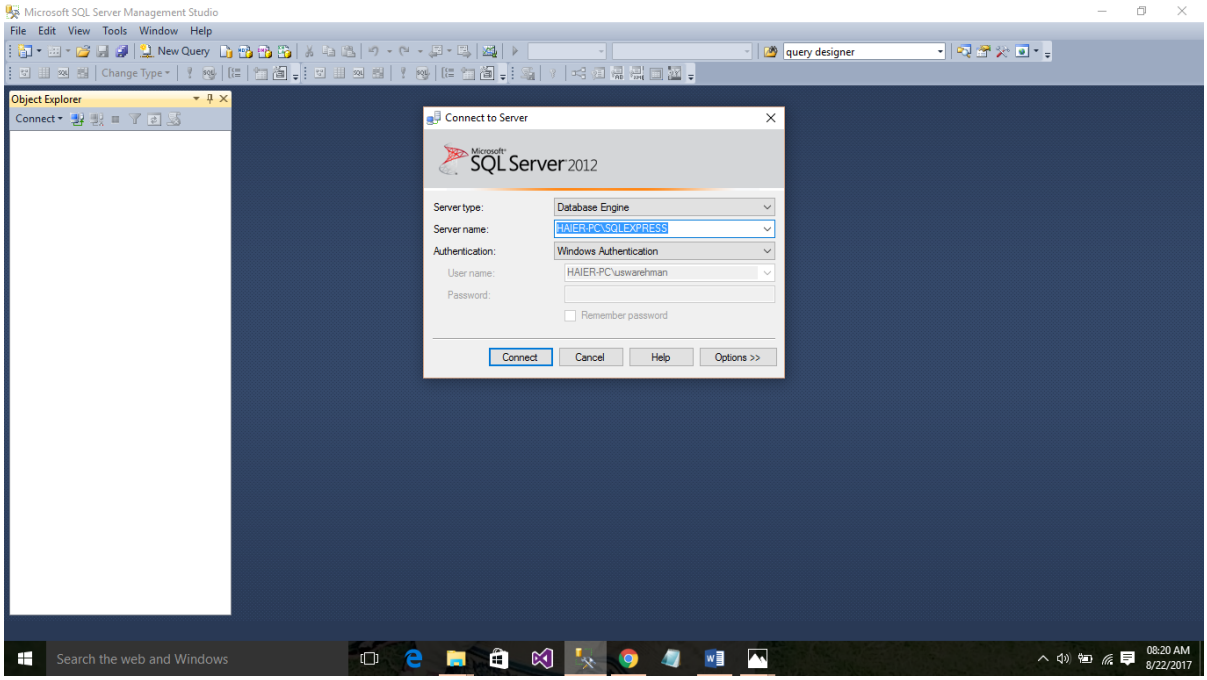


Figure 4.19 creating instance of SQL server database engine

- Right-click **Databases**, and then click **New Database**.

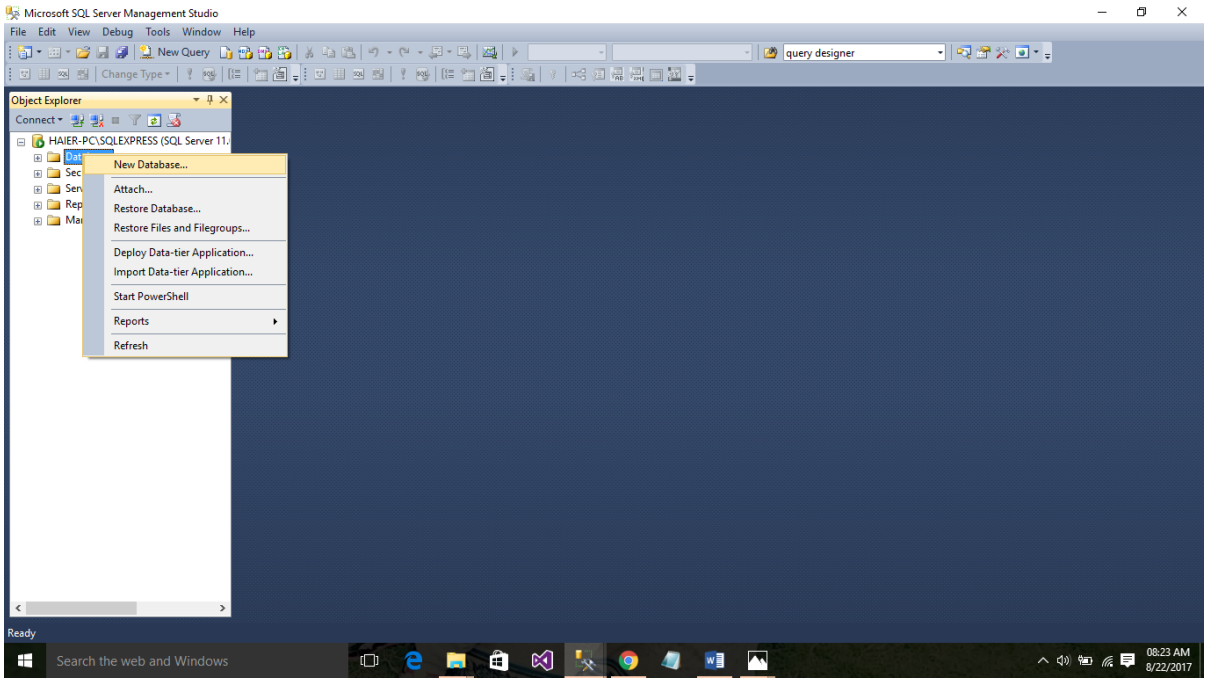


Figure 4.20 Add new Database in SQL studio

- In **New Database**, enter a database names

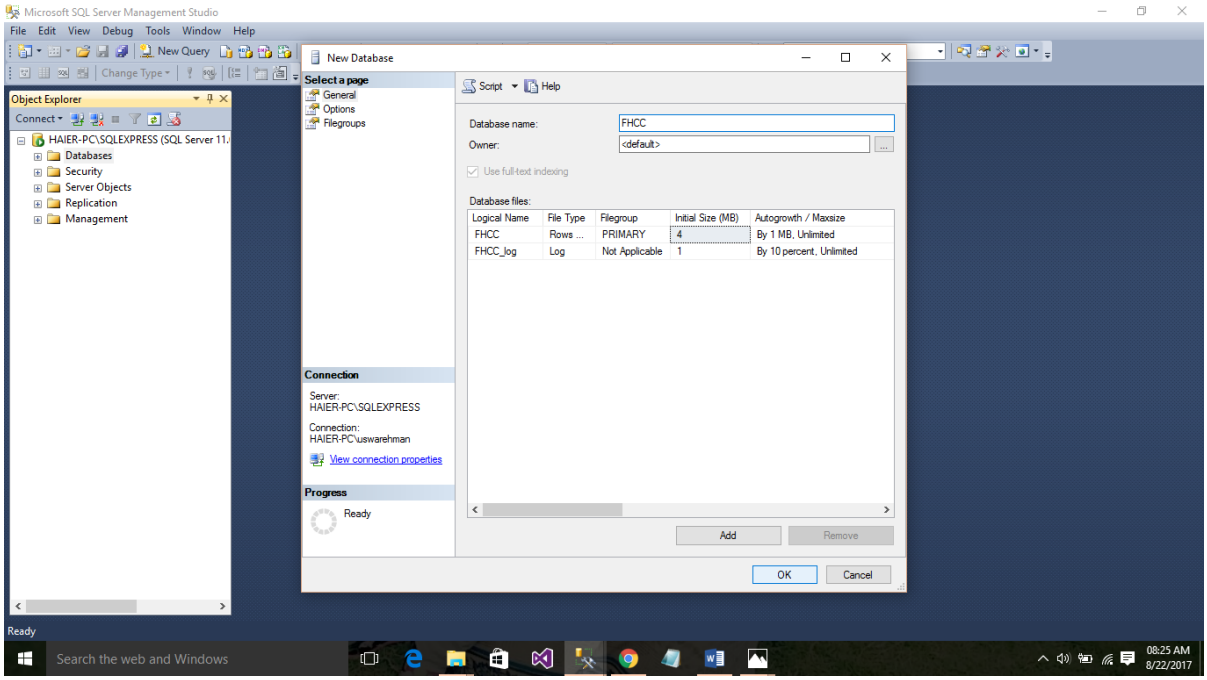


Figure 4.21 Create database by giving name FHCC System

- Add **New Table in** the database

  Enter entities of the table like name id etc and then give sortable name to the table

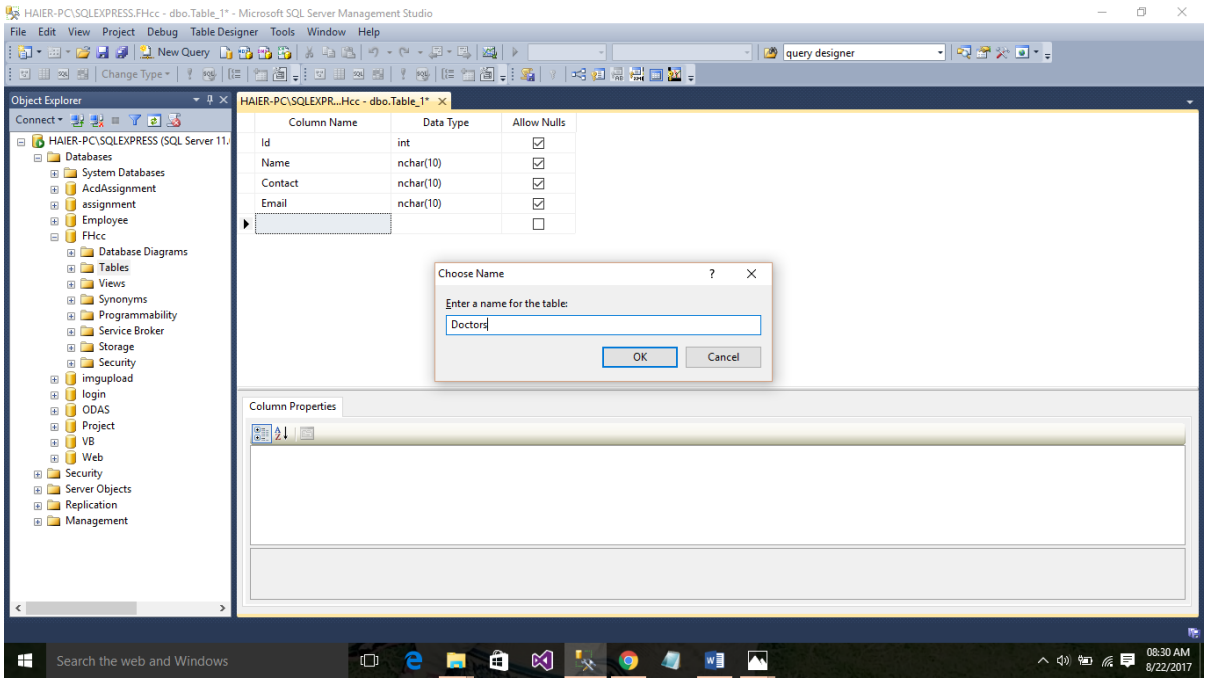


Figure 4.21 Add Doctors table in FHCC Database

- Add Stored procedure in Database

- Select programmability and then add a new stored procedure in database

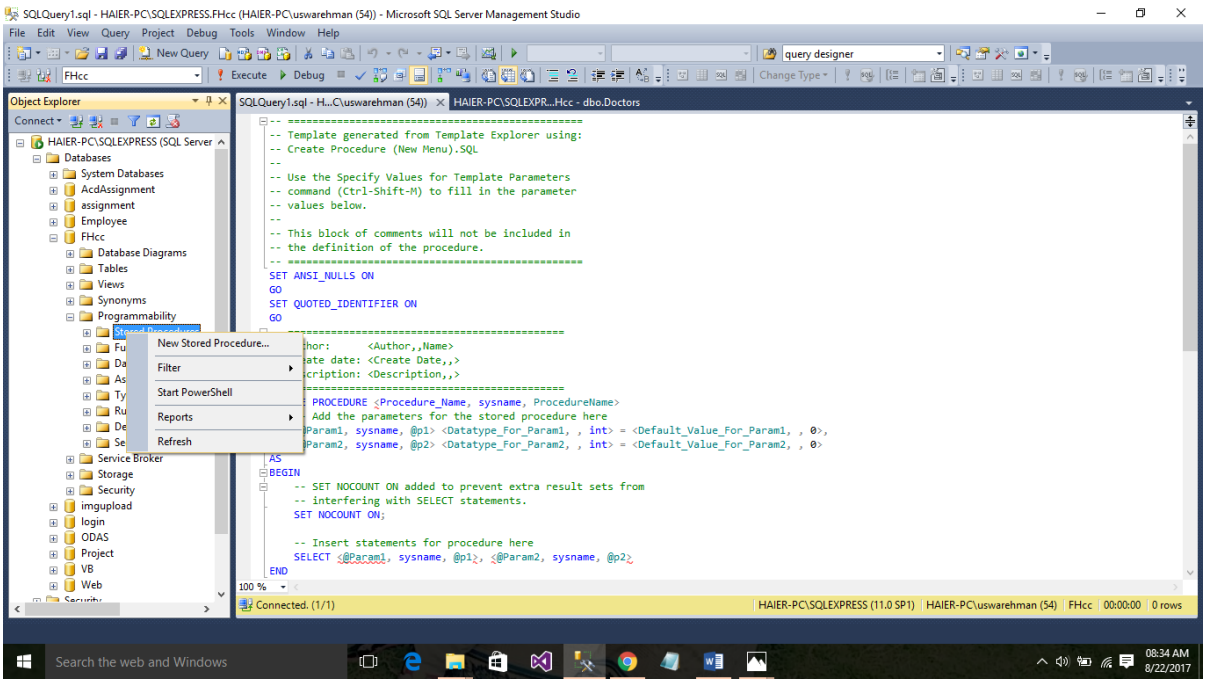


Figure 4.22 Adding Stored Procedure in FHCC Database

- Adding Views In Database

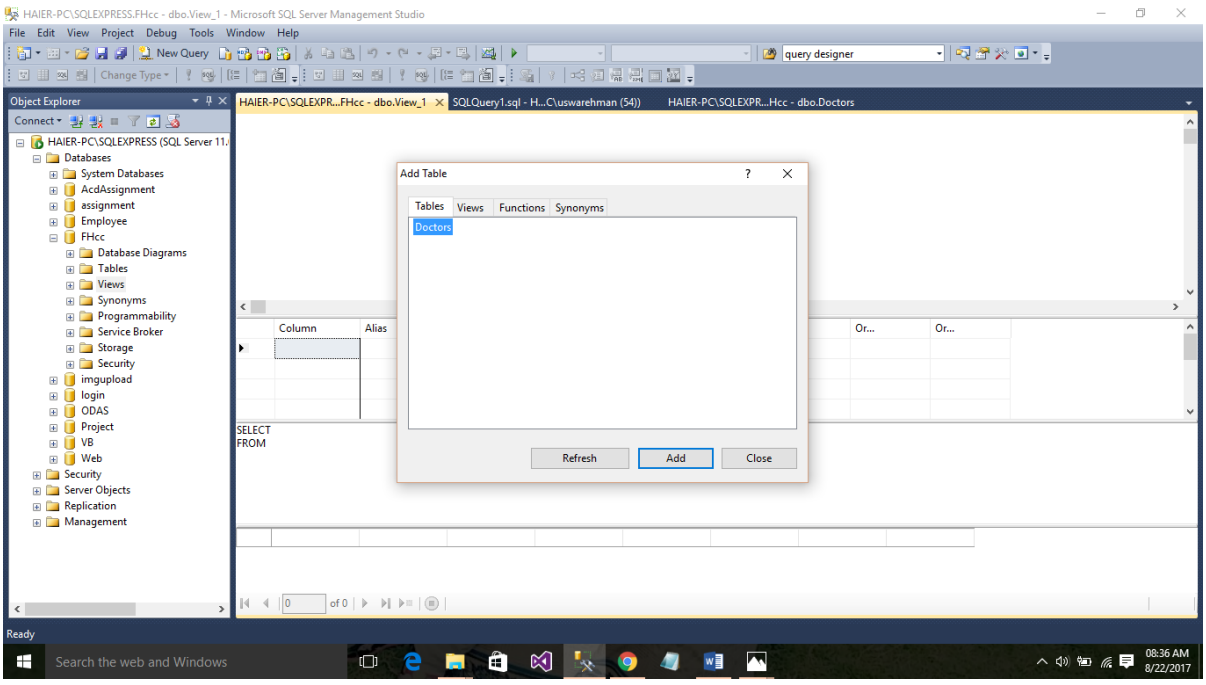


Figure 4.23 Adding View in FHCC database

Database views are used to combine more than one classes by adding relationship between them. It is used to show tow tables data in one place in an organized manner.

4.4 Tables In database

4.4.1 Appointment table

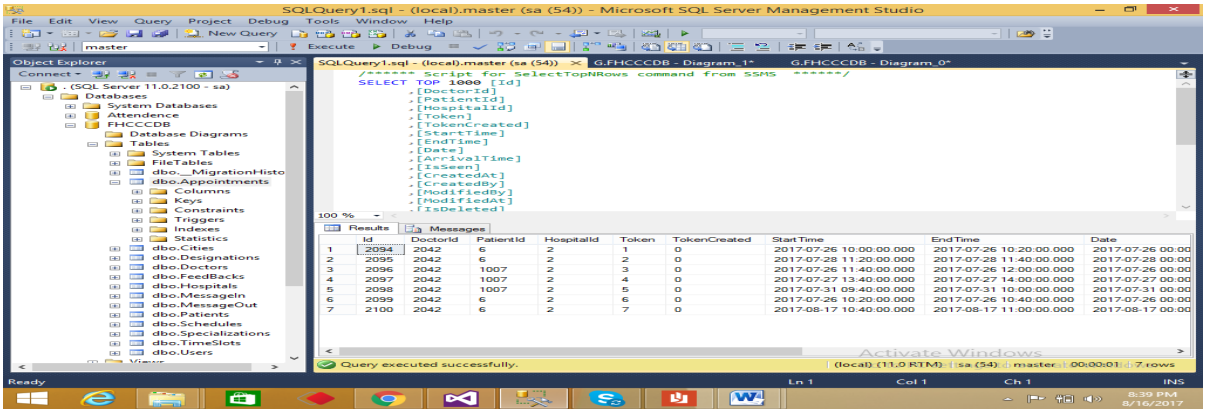


figure 4.24 Appointment table in FHCC database

The appointment table contain doctor id hospital id patient name time date end time of appointment start time of the appointment contact number of the patient created by and other entities that are used to make appointments.

4.4.2 Patient table

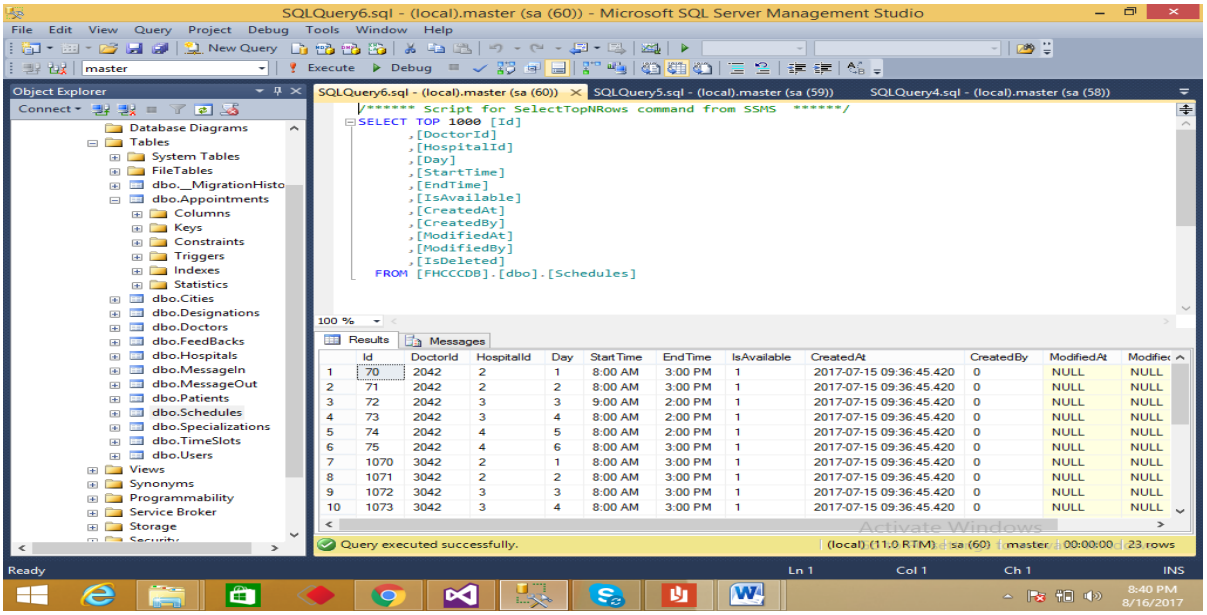


Figure 4.24 Patient table In FHCC database

The Location class serves as the cornerstone, encapsulating vital attributes of allocated locations. Within this class, essential properties and methods are housed, facilitating accurate data manipulation and nurturing harmonious relationships with related classes..

4.4.3 Doctors table

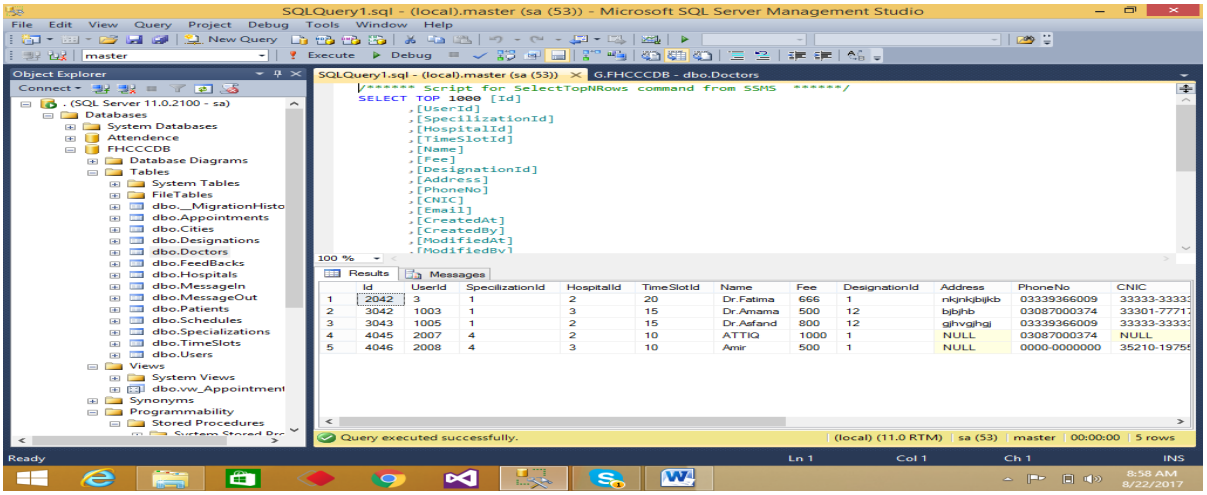


Figure 4.25 Doctors table in FHCC database

The Doctors table contains all the information of the Doctor name id, city contact number fathers name, email id and all other information that is needed by the patient regarding the doctor.

4.5Location Allocator system database Views

In the Location Allocator App, the database employs various views to enhance user experience and accessibility. A database view is a dynamic, searchable object defined by a query, serving as a virtual representation of data

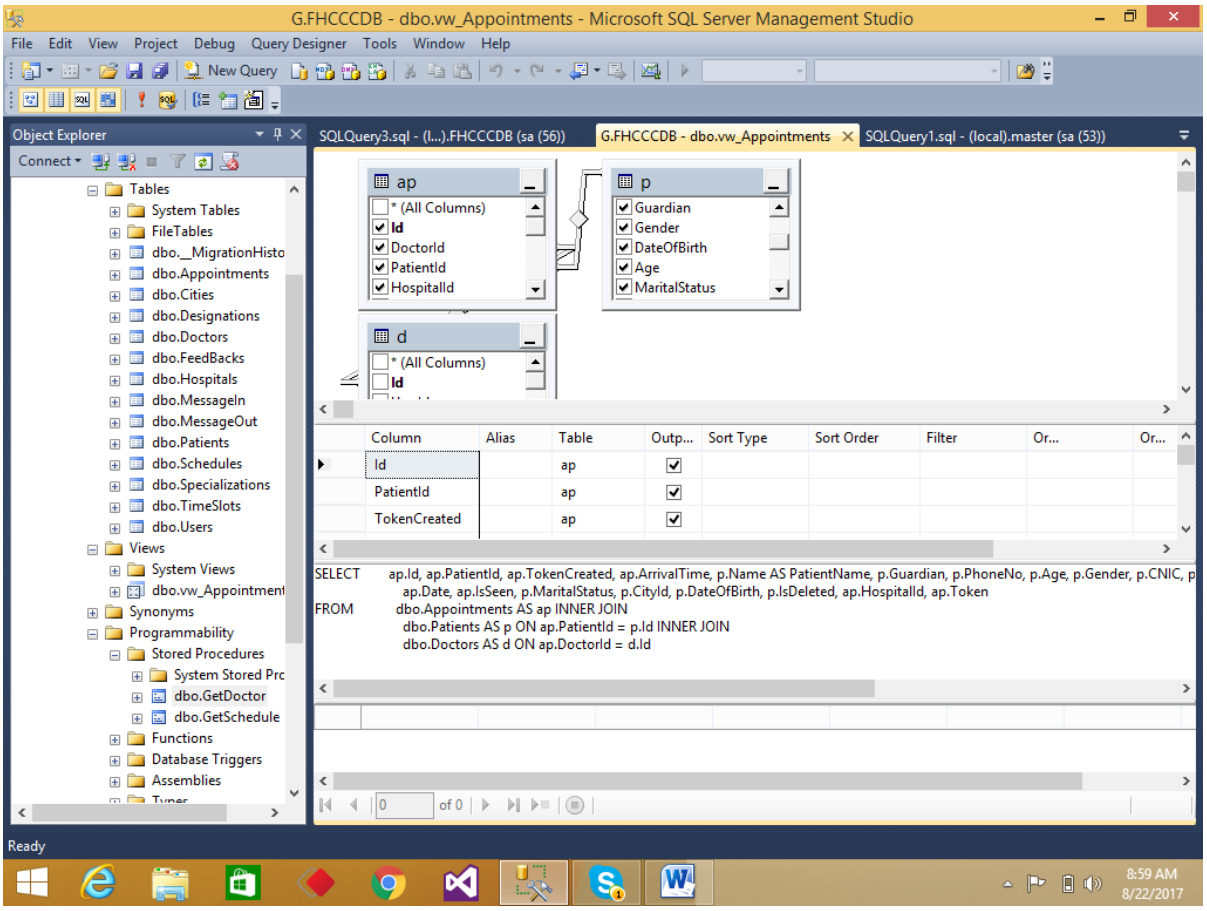


Figure 2.26 view created by doctor patient and Appointment table

This view is generated by combining data from the doctor, patient, and appointment tables. It facilitates the display of appointments to both patients and doctors and enables seamless communication between them.

4.6 Stored Procedure for LOcation Allocator App database

Stored procedures, essential subroutines for accessing the relational database management system, play a pivotal role in the app's functionality. One such procedure is the "Get Doctors by Location" procedure. This procedure is utilized to retrieve doctors within a specific location, identified by hospital ID and doctor ID. Users can easily find doctors within a particular hospital, view their schedules, and schedule appointments.

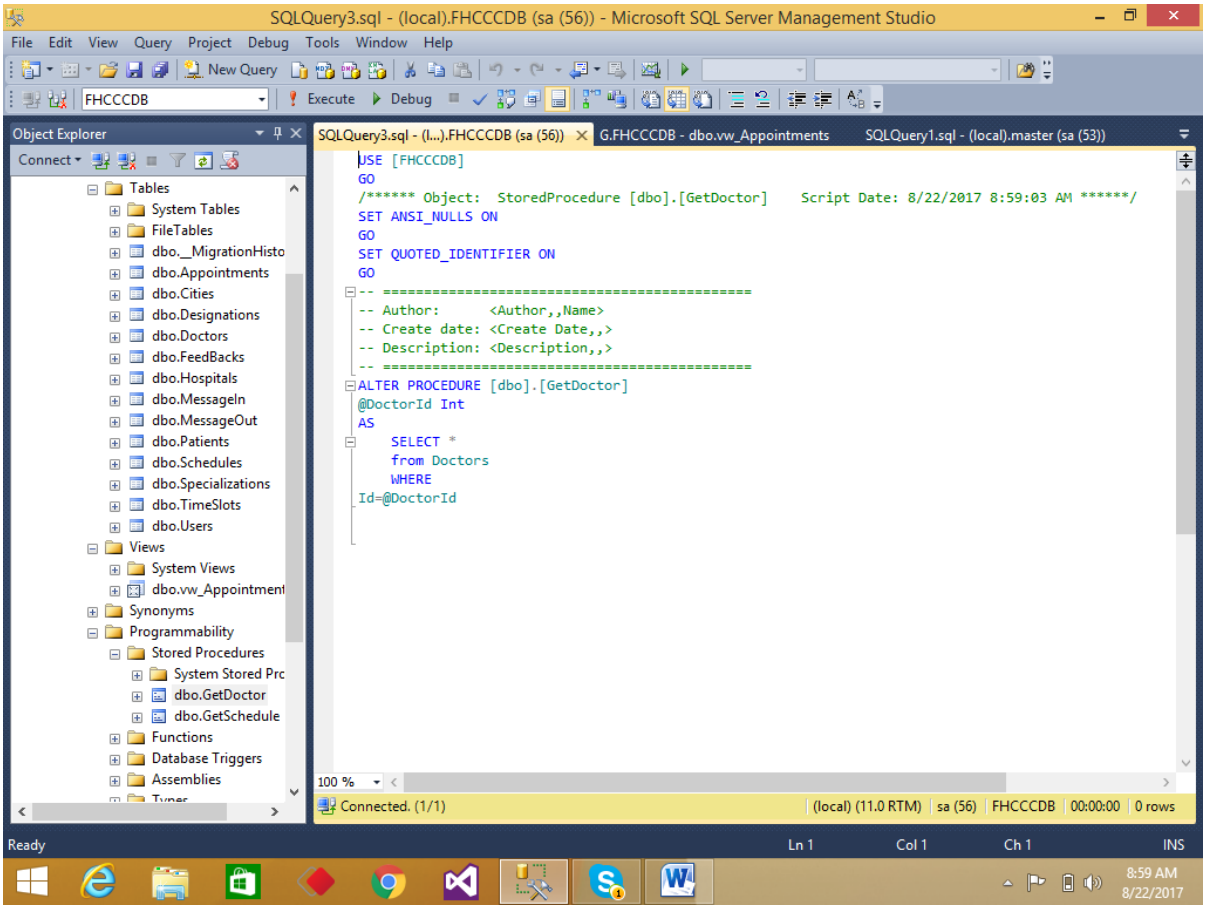


Figure 4.26 store procedure for getting doctors by hospital id and doctor id

4.6 Class Diagram for Location Allocator App database

The app employs class diagrams to represent the system's structure using the Unified Modeling Language (UML). These diagrams serve as the foundation for object-oriented modeling, describing classes, attributes, methods, and

relationships among objects. They provide a visual roadmap for both conceptual and detailed modeling, guiding the translation of models into programming code.

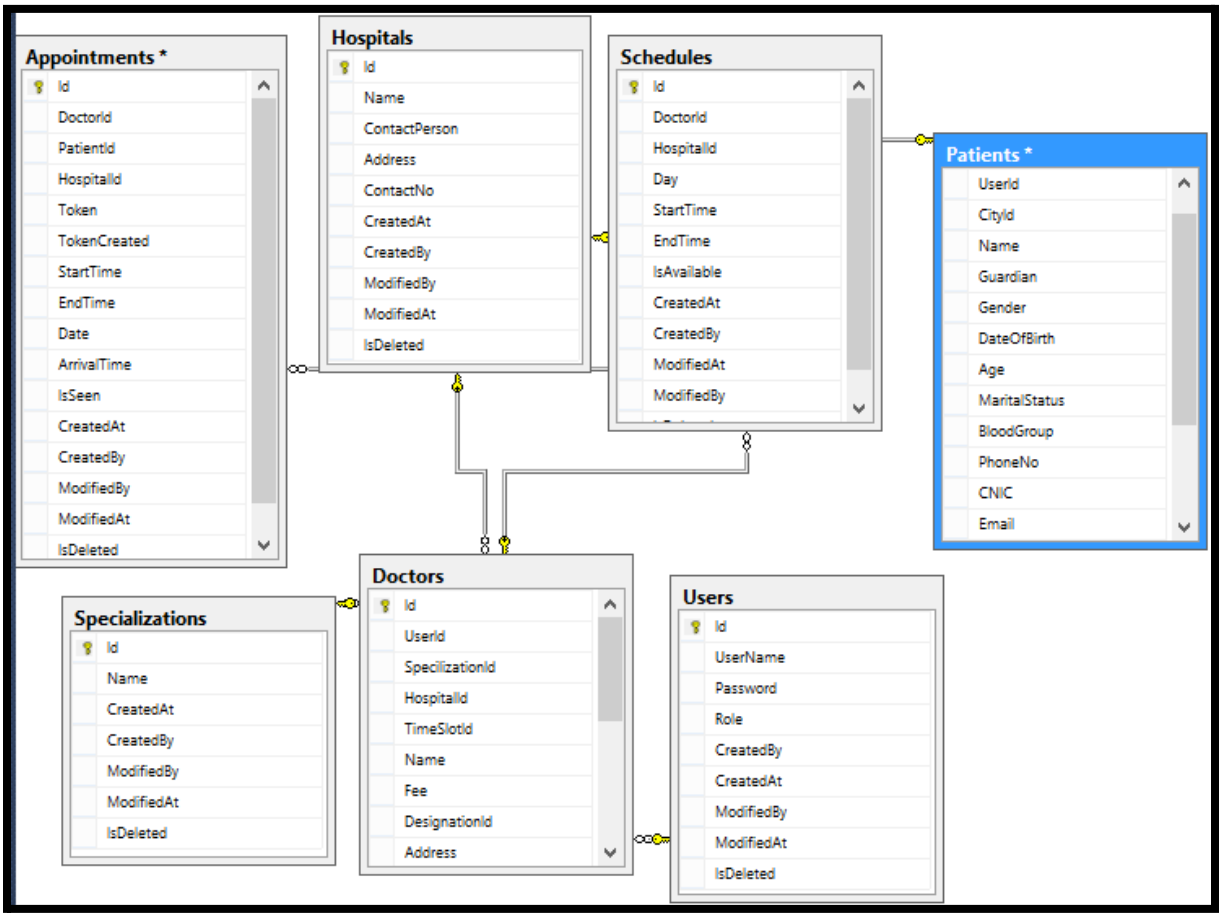


Figure 4.27 Class Diagram of Location Allocator App database

The Class Diagram of Location Allocator App Database is meticulously designed within SQL Server Management Studio. This diagram outlines the essential classes required for developing the app and illustrates their interrelationships. Symbols and lines are employed to succinctly describe the associations between these classes, aiding in the comprehension of the app's underlying architecture

.

Chapter 5

Software Quality Assurance & Testing

5.1 SOFTWARE QUALITY ASSURANCE PLAN:

5.1.1 Purpose

Your Software Quality Assurance Plan (SQAP) seems to be well-structured and comprehensive, addressing various aspects of quality control and assurance in your project. It's great that you've included detailed procedures for different stages of software development and testing. Here are some observations and suggestions to enhance your plan further

5.1.2 Clarity and Consistency:

While your plan covers a lot of ground, some sections could benefit from clearer language and consistent formatting. This will make it easier for team members to understand and follow the procedures.

5.1.3 Introduction and Overview:

Consider adding an introductory section that explains the importance of software quality assurance in your project. Give a brief overview of what the plan entails and how it will contribute to the success of the project.

5.1.4 Roles and Responsibilities:

Explicitly outline the roles and responsibilities of team members involved in quality assurance. This will help ensure that everyone understands their roles and avoids misunderstandings.

5.1.5 Traceability:

Emphasize the importance of traceability. Explain how requirements will be traced through each stage of development, testing, and verification. This ensures that all requirements are met and validated.

5.1.6 Testing Strategy:

Expand on the testing strategy for each testing phase. Describe the types of tests (unit, integration, system, validation) you will perform, the criteria for passing each test, and how defects will be tracked and resolved

5.1.7 Reporting and Communication:

Include a section on how defect reporting, tracking, and resolution will be managed. Describe how communication will occur between different teams, including developers, testers, and project managers.

5.1.8 Change Management:

Address how changes to requirements, design, or code will be handled and communicated to ensure consistency and avoid scope creep.

5.2 Risk Management:

Your brief risk management section is a good start. Consider expanding on potential risks, their impact on the project, and the specific mitigation strategies you plan to employ

5.2.1 Training:

The training section should include not only developer training but also training for testers, project managers, and other relevant stakeholders involved in the quality assurance process

5.2.2 Review Process:

.

Provide details on the review process for various project artifacts, including documents, code, and design. Clarify how feedback from reviews will be collected, documented, and acted upon.

5.2.3 Continuous Improvement:

Consider including a section on how lessons learned from each project phase will be incorporated into future projects to promote continuous improvement.

5.2.4 Validation and Acceptance Criteria:

System testing begins when software has been integrated .The purpose of system testing is to identify the operational envelope of the project. SPM will ensure that the results of the stress tests are addressed where the results show that the operating envelope does not meet requirements identified in the Software Test Requirements Document.

5.2.5 Validation Testing

Detail the scope and approach for validation testing. Describe the scenarios that will be tested and how the results will be evaluated against the specified acceptance criteria.

5.3 Document problems

Address how documentation will be managed, including version control, access permissions, and archiving.

Tool Usage: If you plan to use any specific tools for testing, version control, or project management, mention them and explain how they will be integrated into your quality assurance process.

5.3.1 Code problems

- Lack of functionality: The location allocator app should be able to locate businesses and services in the user's area. If the app is missing any functionality, this will be considered a code problem.
- Wrong functionality: The location allocator app should provide accurate and up-to-date information about businesses and services. If the app is providing incorrect or outdated information, this will be considered a code problem.
- Noncompliance with coding or commentary standards: The location allocator app should be coded in a consistent and readable manner. If the code is not compliant with coding or commentary standards, this will be considered a code problem.

5.3.2 Problem solving procedure

When a problem is detected in the location allocator app, the following procedures will be followed:

1. The developer will be responsible for solving the problem.
2. Once the problem is solved, the developer will notify the QA team to check whether the changes solve the problem.
3. If the problem cannot be solved, or cannot be solved within a reasonable amount of time, a meeting will be set up with the project manager (PM), the

quality assurance manager (QAM), and the team leader of the responsible team. During this meeting, a decision will be made about further dealing with the problem.

5.3.3 Code control

- Documents related to the location allocator app will be available to all people who are authorized to access them and to no one else.
- All versions of documents related to the location allocator app will be available.
- No file related to the location allocator app will be unnecessarily locked.
- Name conventions will be consistently used for all documents related to the location allocator app

5.3.4 Hardware control

- Hardware that is used to develop and test the location allocator app will be available to all people who are authorized to access them and to no one else.
- All types of hardware that are used to develop and test the location allocator app will be available.
- Only authorized personnel will use hardware that is used to develop and test the location allocator app.
- Name conventions will be consistently used for all hardware that is used to develop and test the location allocator app.

.

5.3.5 Media control

The SQA team will check to make sure that the procedures and standards described above are being handled properly. This will be done through

reviews and random checks. Any problems found will be reported back to the developers.

5.3.4 Supplies control

- The tools that will be used for development and testing of the location allocator app (Visual Studio and SQL Server) will be available to all project members.
- All external software components that are used in the location allocator app, that have an unreliable source, will be tested according to the ESA standards.
-

5.3.5 Training

The developer has taken the following courses at GC University and has research experience under the guidance of professors:

- Software Engineering - 1
- Software Engineering - 2
- Software Management
- Software Specification

5.4 Risk management

The SQAP includes a risk management strategy to address potential challenges and uncertainties that may arise during the development and deployment of the location allocator app:

- Budget overruns: Strategies will be in place to monitor and manage project costs, ensuring adherence to the allocated budget.
- Time overruns: Proactive scheduling and time management techniques will mitigate the risk of project delays.

- Complexity management: The team will employ effective complexity management strategies to handle intricate components and requirements of the location allocator app.
- Verification risks: Rigorous verification processes will be followed to identify and rectify issues early in the development cycle.
- Validation risks: Comprehensive validation testing will be conducted to ensure the location allocator app meets user expectations and requirements.

Postmortem report For Location Allocator App

Sure, I can help you with that. Here is a postmortem report for the Search Around Me Smart Location Allocator Application project:

Project Name: Search Around Me Smart Location Allocator Application Project Manager: Ayesha Arooj Project Start Date: January 1, 2023 Project End Date: March 8, 2023

Project Goals:

- To create a user-friendly mobile app that enables users to discover and explore various places within a specified range based on selected categories.
- To simplify the process of finding nearby places of interest, streamlining user experiences, and enhancing their ability to explore their surroundings conveniently.
- To provide a mobile-friendly, efficient tool for allocating and managing locations while ensuring a seamless mobile experience.

Project Accomplishments:

- The project was completed on time and within budget.
- The app was well-received by users, with positive feedback on its user-friendliness and functionality.
- The app was able to successfully simplify the process of finding nearby places of interest, providing users with a convenient way to explore their surroundings.
- The app was secure and reliable, with no major security incidents or outages during its operation.

Project Challenges:

- The project faced some challenges, such as the need to integrate with multiple APIs and the difficulty of finding reliable data sources.
- The project team also faced some challenges in managing the project effectively, such as the need to coordinate with multiple stakeholders and the difficulty of staying on track with the project schedule.

Project Lessons Learned:

- The project team learned the importance of careful planning and project management in order to deliver a successful project on time and within budget.
- The team also learned the importance of using reliable data sources and integrating with multiple APIs in order to provide a comprehensive and user-friendly app.
- Finally, the team learned the importance of communication and collaboration with stakeholders in order to ensure that the project met the needs of all users.

Overall Assessment:

The Search Around Me Smart Location Allocator Application project was a success. The app was well-received by users and was able to successfully simplify the process of finding nearby places of interest. The project team learned a lot from the experience and is confident that they can apply these lessons to future projects.

Recommendations for Future Projects:

- The project team recommends that future projects use a more agile development methodology in order to be more responsive to change and deliver projects on time and within budget.

- The team also recommends that future projects use a more robust testing methodology in order to ensure the quality of the product.
- Finally, the team recommends that future projects use a more collaborative approach to project management in order to ensure that the needs of all stakeholders are met.
-