HHCNN: HOME HEALTH CARE NURSE NETWROK

A WEB-BASED HOME HEALTH CARE

PLATFORM TO COVER NURSES

IN CASE OF ABSENCE

A Project Report

Presented to

The Faculty of the Department of Computer Science

California State University, Los Angeles

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

in

Computer Science

By

Nazli Rahpiefard

May 2018

The project report of Nazli Rahpiefard is approved.

Yuqing Zhu, Ph.D., Committee Chair

Mohammad Pourhomayoun, Ph.D.

Zilong Ye, Ph.D.

Raj Pamula, Ph.D., Department Chair

California State University, Los Angeles

May 2018

ABSTRACT

HHCNN: Home Health Care Nurse Network, a Web-based Home Health Care Platform to Cover Nurses in Case of Absence

By

Nazli Rahpiefard

HHCNN is a web-based software solution for Home Health Care facilities to cover nurses in case of nurse absence.

Home Health Care provides medical care by skilled medical professionals. Once the nurse calls in sick the facility must cover for the employee as fast as possible since the assignment must be done on the same date and time. Traditionally facilities make many phone calls to fill out an open assignment in case of nurse absence, which is costly and reduces productivity.

HHCNN platform allows facilities to post available assignments on the system and have the nurses who meet the criteria of the assignment get notified to fill out the assignment quickly. This software provides a user-friendly interface for both nurses and facilities, which makes it easy to use without spending lots of time on it.

The focus is on filling out the assignments fast and cost efficiently.

ACKNOWLEDGMENTS

I would like to express my great appreciation to my advisor Dr. Yuqing Zhu for his support, guidance, and comments.

I would also like to thank Computer Science department chair Dr. Raj Pamula for providing me the opportunity to work on this project to expand my skills and get closer to achieving my future goals.

I would like to offer my special thanks to California State University Los Angeles Alumni Sarang Nazari and Mohammad Yazdani for their guidance and advice.

Finally, I must express my profound gratitude to my family for providing me with support and encouragement throughout the development of this project.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF ABBREVIATION

HHCNN	Home Health Care Nurse Network
IDE	Integrated Development Environment
SQL	Structured Query Language
GUI	Graphical User Interface
JSON	JavaScript Object Notation
DB	Database
IP	Internet Protocol
XML	Extensible Markup Language
NPM	Node Package Manager
CLI	Command Line Interface
JS	JavaScript
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
UML	Unified Modeling Language
MVC	Model View Controller
DBMS	Database Management System
RDBMS	Relational Database Management System
UI	User Interface
STS	Spring Tool Suite
API	Application Programming Interface
URL	Uniform Resource Locator
SPA	Single Page Application

CHAPTER 1

Introduction

HHCNN is a web-based software solution for Home Health Care facilities to cover nurses in case of nurse absence.

HHCNN platform allows facilities to post available assignments on the system and have the nurses who meet the criteria of the assignment get notified to fill out the assignment quickly. This software provides a user-friendly interface for both nurses and facilities, which makes it easy to use without spending lots of time on it.

Without using this system, the process of filling out assignments is time consuming. This can lead to low productivity. Each facility has limited resource of nurses. If a nurse calls in sick in the situation that all other nurses are fully scheduled, there is a need to cover the assignment by accessing outside nurses. In these situations, facilities call Nurse Registry Agencies to provide them with nurses. The issue is that nurses who work for these agencies tend to be unqualifies and it leads to having unsatisfied patients. To prevent unsatisfied patients HHCNN suggests using nurses from the inner circle of the system. This way the nurses who fill out assignments are guaranteed to be qualified and keep the patients satisfied.

In chapter 2 tools and technologies used in development of this project will be introduced. Chapter 3 goes over system analysis and design of the project, chapter 4 is about implementing the project and finally conclusion of the thesis is on chapter 5.

CHAPTER 2

Tools and Technologies

The tools and technologies used in development of this project are as followed.

Integrated Development Environment

IntelliJ IDEA Ultimate

IntelliJ IDEA is the Integrated Development Environment used in developing this project, which is a Java IDE. The Ultimate version is the commercial edition which supports many frameworks as well as Spring Boot. IntelliJ is developed by JetBrains software development company.

Database

MongoDB

The database used for storing data in this project is MongoDB that is open-source. MongoDB is a cross-platform and open source database that is document-oriented. This NoSQL database uses documents and collections instead of tables and rows. MongoDB uses JSON-like documents with schemas.

Robo 3T (formerly Robomongo)

Robo 3T is a lightweight and free Graphical User Interface for MongoDB. This MongoDB management tool provides shell-centric GUI to work with documents, get queries and modify or delete collections or documents. This tool was used for purposes of test, debugging, and accessing data during development.

Version Control

Version Control Systems used in development of this project are GitHub and Bitbucket. Software development needs a version control system for many reasons such

as having a backup, keeping track of the amount of codes changed or added to the project, restoring to the previous version, and collaborating.

GitHub

GitHub is a version control which is web-based. It uses Git which is an open-source version control system but adds its own features to it. Git uses command-line, but GitHub provides the user with both web-based GUI and desktop. GitHub improves creating and publishing sites and provides features for request reviews, comment in context, protect branches, and track and assign tasks.

Bitbucket

Bitbucket is another version control system similar to GitHub and is owned by Atlassian enterprise software company. It uses either Mercurial or Git. Bitbucket provides free private repositories with limited number of users. Features supported by Bitbucket are IP whitelisting, Merge Checks, Git Large File Storage, Issue tracking, Smart Mirroring, and 2 step verifications.

Server-Side Technologies

Spring Boot

Spring Boot is a light weight framework to write Spring based java web applications. Comparing with Spring Framework, Spring Boot reduces the lines of code and boiler plate configuration. It provides default configurations which are overridable.

Gradle

Build Tools programs are used to automate the creation of applications from source code. It is possible to manually build projects in small applications, but it is almost impossible to keep track of all the changes and dependencies needed to be downloaded

and the sequence of the builds. It is needed to use a build tool to do the compiling, packaging, linking, and building automatically.

Gradle is a build tool which is open-source and builds upon the concept of Apache Ant and Apache Maven. Gradle doesn't use XML as it is used in Apache Maven to declare configurations of the project. Gradle performs downloading dependencies, compiling code to binary, packaging, running test, and deployment to production systems.

Client-Side Technologies

Angular 5

Angular is a platform that combines declarative templates, dependency injection, end to end tooling, and integrated best practices to solve development challenges. Angular is built and maintained by Google. The angular version used in this project is Angular 5. In installation of Angular 5, we need node.js with NPM (Node Package Manager) to install Angular-CLI (Angular Command Line Interface). Angular 5 is the latest version of Angular. And its framework was written using TypeScript in comparison with Angular JS which was written using JavaScript. Angular 5 makes angular smaller and faster to use.

TypeScript. Typescript is an open-source programming language which was built on top of JavaScript. It is developed and maintained by Microsoft. It adds new features to JS. TypeScript works better with complex software projects and is used in Angular 5.

HTML. Hypertext Markup Language is the standard markup language which is used in Angular 5 for creating web pages. HTML describes the structure of web pages using markup elements which are presented by tags.

CSS. CSS or Cascading Style Sheets describes the presentation of HTML pages. CSS has a simple syntax and uses English words to name different style properties. The basic style sheet used in styling HTML pages in this project is CSS.

Bootstrap

Bootstrap is a front-end library for designing web pages using HTML and CSS. It provides templates, tables, forms, buttons, and typography to build responsive web pages.

System Modeling

System Modeling is a process of developing abstract models of a system which is used during system analysis of a system. System modeling uses graphical notation to present different aspects of a system using different perspectives.

UML

The Unified Modeling Language is a modeling language used in software engineering and provides a standard way to visualize system design. There are different categories of diagrams such as Structural Diagrams and Behavioral Diagrams. Structural Diagrams include class, object, component and deployment diagrams. Behavioral Diagrams include use case, sequence, collaboration, state chart, and activity diagrams.

This project was developed by integrating different technologies, frameworks and languages which were mentioned earlier. Using tools like version controls, database user interface and modeling tools speeded up developing the project.

CHAPTER 3

System Analysis and Design

In this chapter System analysis, database, server-side, and client-side design of the project will be discussed.

System Analysis

In the process of system development, system analysis is one of the main phases. It is a process of collecting facts, identifying problems, and understanding the system in detail. Modeling is used at this step to visualize the results of analyzing the system. All diagrams in this chapter are created according to UML standards.

Use Case Diagram is the simplest representation of a user interaction with the system and identifies system requirements. Use case Diagrams are consist of Actors, Relationships, System, and Use Case Scenarios. It is a high-level diagram and doesn't include much details. Actors are the users interacting with the system. Use cases are the actions done by actors in the system. Relationships are the relationship between user and action which vary in type. In this project the System is a website.

General Use Case Diagram

General Use Case Diagram illustrates general use cases in the system. A rectangle shows the system being analyzed which is Home Health Care Nurse Network here. Actions are shown with oval shapes which are called use cases.

There are two categories of actors including Nurse and Facility. Actors are external parts of our system, so they are put outside of the rectangle. The reason there is another user called Generic User is to avoid repeating common activities between actors.

To avoid repeating those use cases we generalize the actors to Generic User. The relationship Generic User has with Nurse and Facility is of type Generalization.

Generic Actor in this Diagram has an Association Relationship with two Use Cases called Registration and Login. Association Relationship Type is the basic relationship between actors and use cases.

Nurse Actor has an Association Relationship with Access Assignment use case. Facility Actor has Association Relationship with Manage Patient and Manage Assignments use cases.

According to figure1 use case diagram, generic user actor registers in the system then logs in. Depending on the type of the actor they can do different actions. Facility actor can manage patients and assignments. Nurse actor can access assignments. The details of these actions will be shown in other use case diagrams. General Use Case Diagram is shown on the next page.

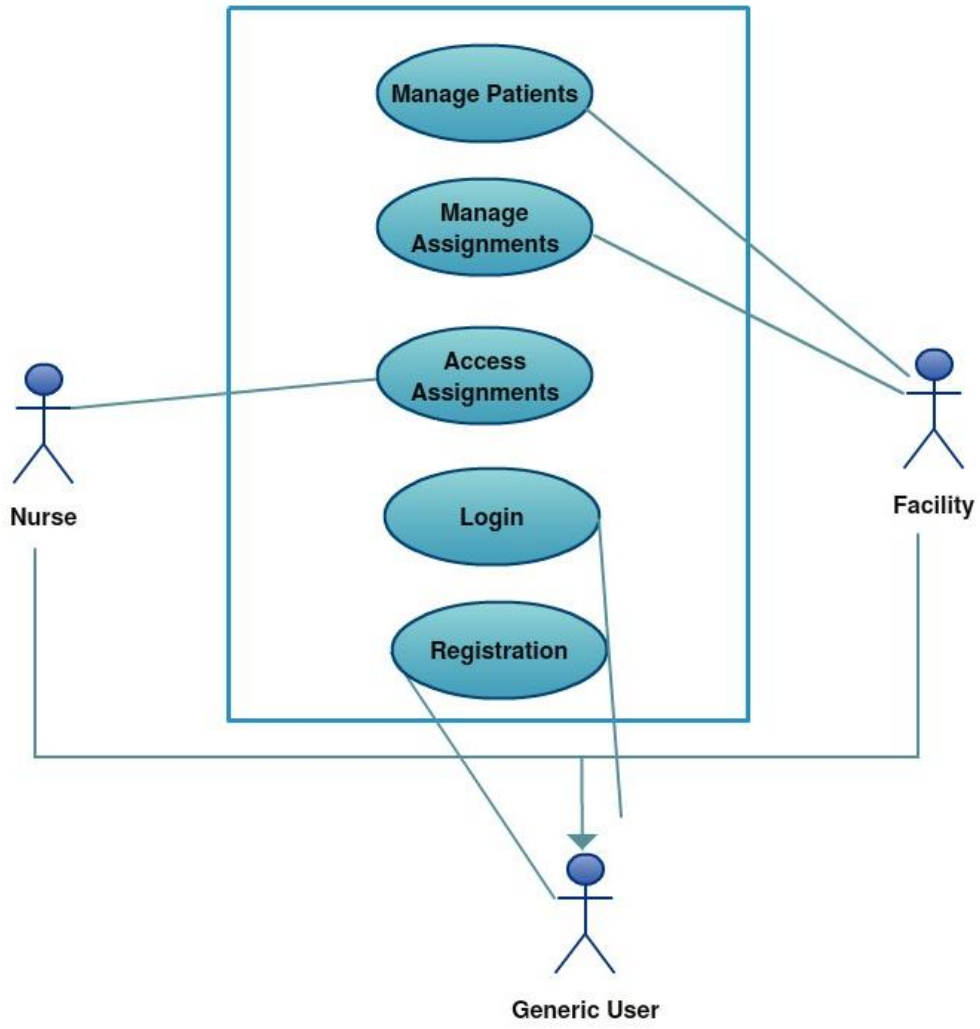

Figure 1. General Use Case Diagram

Patient Management Use Case Diagram

Patient Management use case diagram presents patient management with more details. There is one actor in this diagram called Facility. Facility has association relationship with manage patient's action. There are four other actions or use cases which have different types of relationships with manage patient use case. On one hand, View Patient use case has an include relationship with manage patient which means this action is included in the original action. On the other hand, Add Patient, Edit Patient and Delete Patient use cases have an extend relationship with manage patient use case. Patient management includes viewing patients but adding, deleting, or editing don't always happen. They only occur once there is a specific situation which leads to adding, deleting, or editing a patient.

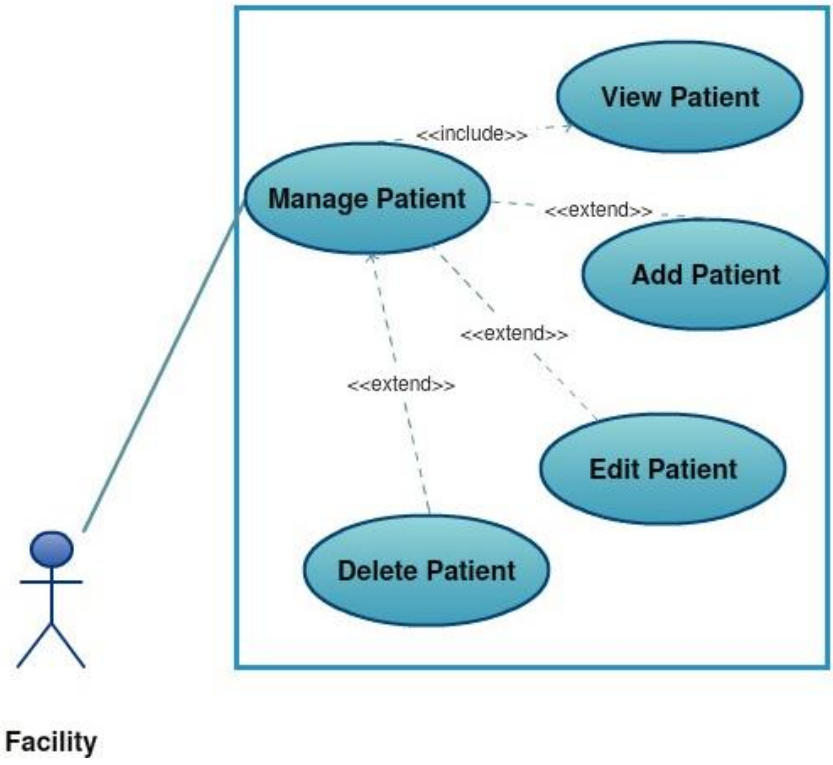

Figure 2. Patient Management Use Case Diagram

Assignment Management Use Case Diagram

Assignment Management use case diagram presents assignment management with more details. There is one actor in this diagram called Facility. Facility has association relationship with manage assignment's action. There are four other actions or use cases which have different types of relationships with manage assignment use case. On one hand, View Assignment use case has an include relationship with manage assignment which means this action is included in the original action. On the other hand, Add Assignment, Edit Assignment and Delete Assignment use cases have an extend relationship with manage assignment use case. Assignment management includes viewing assignments but adding, deleting, or editing don't always happen. They only occur once there is a specific situation which leads to deleting or editing an assignment.

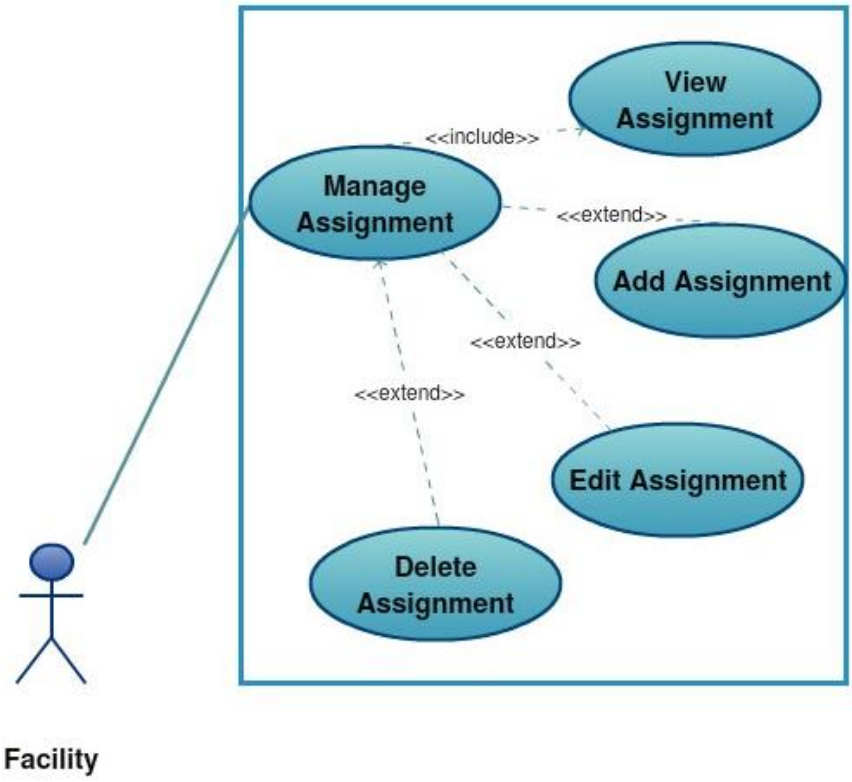

Figure 3. Assignment Management Use Case Diagram

Access Assignment Use Case Diagram

In this diagram the actor is Nurse and there are 3 use cases. Nurse has an association relationship with Access Assignment use case. View Assignment use case has an include relationship with Access Assignment action, which means nurse views and reads assignments every time it accesses assignments. Also, there is another use case for Accept Assignment and it has an extend relationship with Access Assignment. This action only occurs if the nurse is willing to accept the assignment and accepts one or multiple assignments.

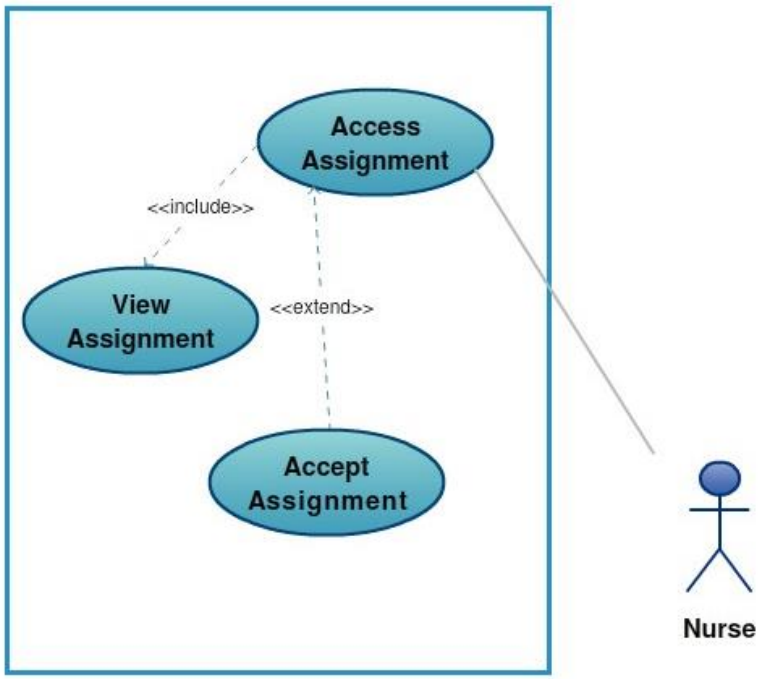

Figure 4. Access Assignment Use Case Diagram

Registration Use Case Diagram

In Registration use case diagram the actor is Generic User which is the result of generalizing actors Nurse and Facility. Generic user has association relationship with Registration action. Registration action has an include relationship with Send Confirmation Email and an extend relationship with confirmation link clicked use case.

As it is shown in the diagram below, Generic User registers in the system and once it registers a confirmation email is sent to the email address provided on the registration form. Once the user receives the email and clicks on the confirmation link his/her account will be confirmed.

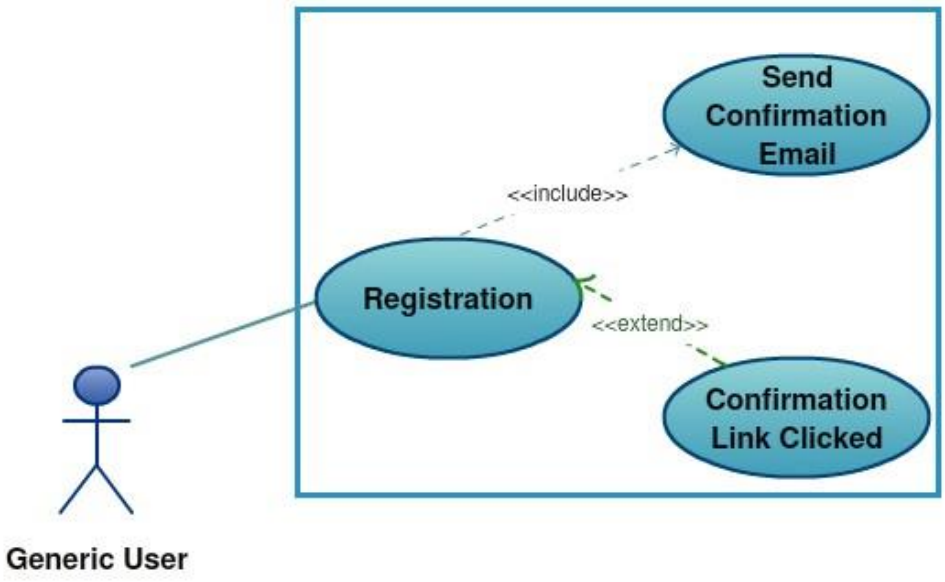

Figure 5. Registration Use Case Diagram

Login Use Case Diagram

In Login use case diagram the actor is Generic User which is the result of generalizing actors Nurse and Facility. Generic user has association relationship with Login action. Login action has an include relationship with Verify Password and Verify Account Confirmation. Login use case has an extend relationship with confirmation link clicked use case.

As it is shown in the diagram below, Generic User logs in the system and system verifies password and checks if the account has been confirmed or not and if both conditions were true lets the user log in to the system. If there is a problem with password or confirmation an error message will be shown.

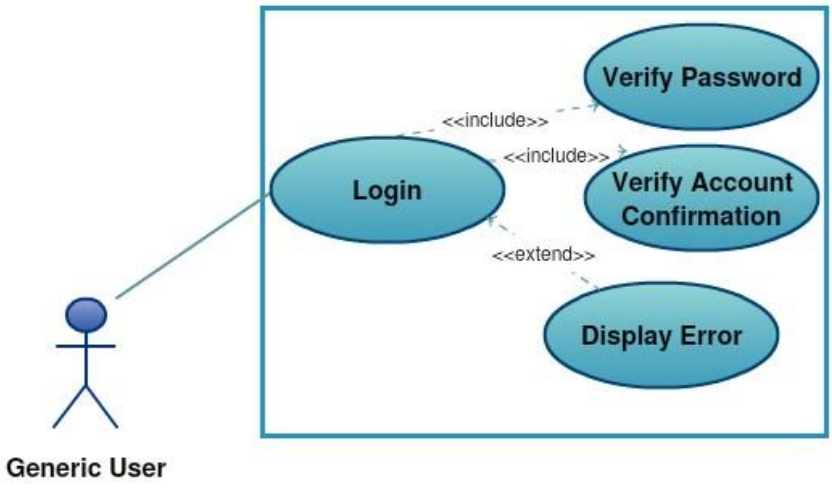

Figure 6. Login Use Case Diagram

System Design

System Design is the process of developing and defining systems to satisfy specific need of the user. The system design diagram on the next page has three main parts as client, server, and database.

MongoDB is used for database, which is a document base database. The architecture used on client-side is Single Page Architecture, which provides a light weight web application. Single Page Architecture is discussed in detail in the following pages.

Server-side of this application has MVC Architecture, which contains model, view, and controller. In addition, there are Business and Repository Layers. Business layer includes classes on the backend to do the logical part of the process. It is used to distinguish the logic part from controller classes, which handle request mapping in the server-side of the code. Repository layer contains repository classes to communicate with MongoDB. All queries, create, update, read, and delete operations are done through repository classes.

Client sends a request through the browser and client-side of the code does the router navigation. Then controller gets the request through Request Mapping. Then accesses the Business class needed. If there is a need to communicate to the database, the relevant repository class gets accessed and data model related to it gets used. After all the process is done in the server-side of code, client-side components access and show the view among with the response.

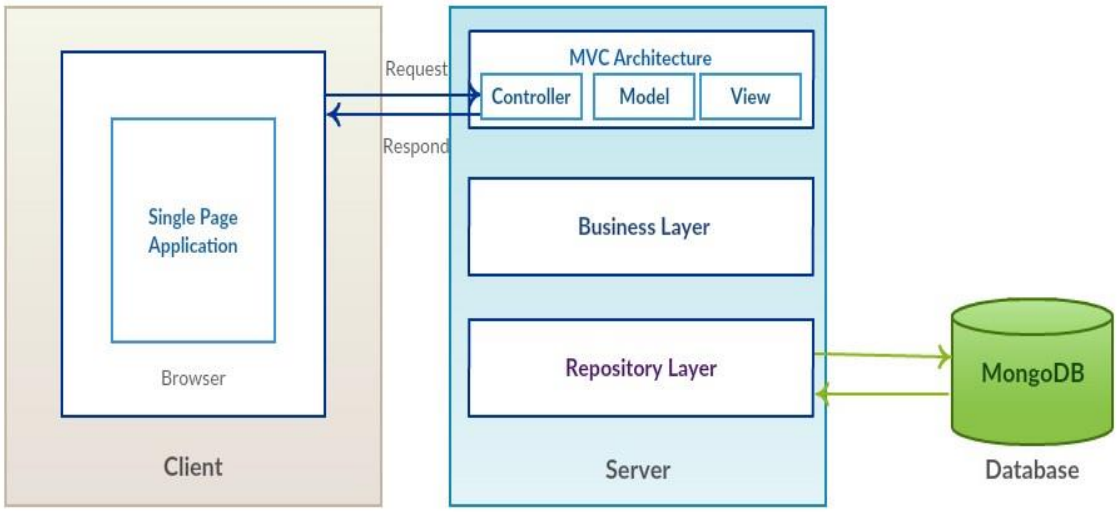

Figure 7. System Design Diagram

MVC Architecture Design

Model View Controller Architecture is a design pattern used in web application development.

Model is responsible for maintaining application data. View is the user interface. It displays data to the user and enables the user to modify data. View in this project is made of Angular components with HTML, CSS, and TypeScript. Controller is responsible to handle user request and response. Client receives the response through the appropriate view with the model data.

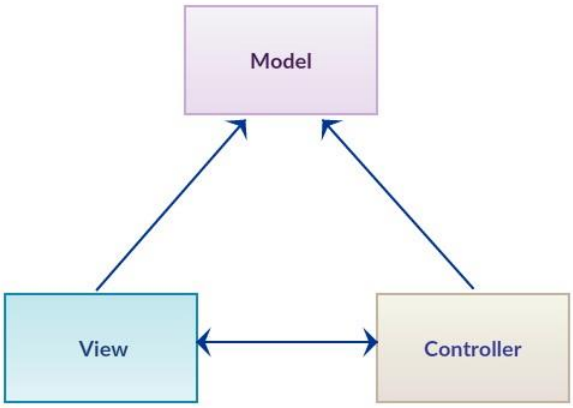

Figure 8. MVC Architecture Diagram

Single Page Architecture

Angular framework and platform was chosen to implement SPA (Single Page Application). Angular modules are the base part of an angular application. A set of angular modules define an angular app. They provide compilation context for components. Components define templates, which are ordinary html that use directives to bind data to the view. Each component provides different functionality using services. Service providers get injected to a component by dependencies.

Decorators mark a class as a service or component. Then provide the metadata for angular to how to use them. Metadata for a component associates it with a template. Also, Angular gets associated with the information it needs to make a service available to a component by metadata. In addition, metadata for directives associates the class with a selector.

In two-way data binding the data property value of a component flows to the input box on the template with property binding. Also, changes a user makes on the view flows back to the component with event binding.

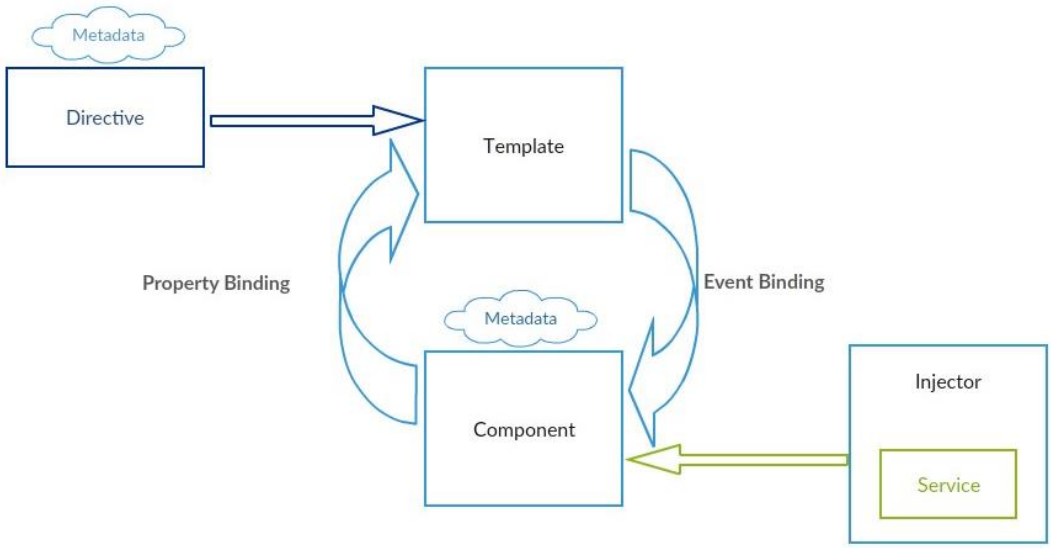

Figure 9. Single Page Architecture Diagram

Database Design

Database Management System provides a mechanism for storing, accessing, and managing data. The DBMS used in this project is a NoSQL DBMS.

NoSQL Database Management System. Comparing with Relational Database Management Systems(RDBMS) NoSQL databases overcome the limitation of big data. NoSQL databases have advantages such as scalability, performance, high availability (in terms of hardware failure), and supporting both structured and unstructured data. Scalability allows handling large amount of data). Performance in NoSQL databases is much higher than SQL databases. Supporting both structured and structured data gives the freedom of not specifying the exact columns and types.

MongoDB DBMS. MongoDB is one of the most popular document-oriented NoSQL Database Management Systems. MongoDB is the best option when it comes to Storing and retrieving great quantities of data, and when the list of elements is growing. In cases that storing relationships between the elements is not important and can be handled on application level, and constraints and validations logics are implemented in application level, MongoDB is considerable. Also using MongoDB speeds up the development of application. In addition, this DBMS is mostly used when there are no complex transactions. MongoDB works the best with data which is not structured or is changing with time.

Referenced Schema Design. There are two types of schema design in MongoDB, Referenced and Embedded. In Referenced design an id of the document which needs to be referenced is saved inside the document. In Embedded design the entire document that it has relationship with is saved inside the document. Referenced document design was

chosen in this project over Embedded design because the subdocuments of any given document continuously increase in size. Considering that in NoSQL databases there is no need to normalize and some data might be stored denormalized, it is recommended to define as many single documents as possible. If there is a need to use another document's data, simply the Id of it is added to the document.

As illustrated in Figure 1, there are five different documents. To refer to a document from another document, an id of the referee document is saved as a field inside the document. These id fields are used to refer to another document and get queries as needed.

Collections in NoSQL databases are a set of documents and is like tables in relational databases. Data in this project is saved inside 5 collections as mentioned below: Nurse, Facility, User, Patient and Assignment.

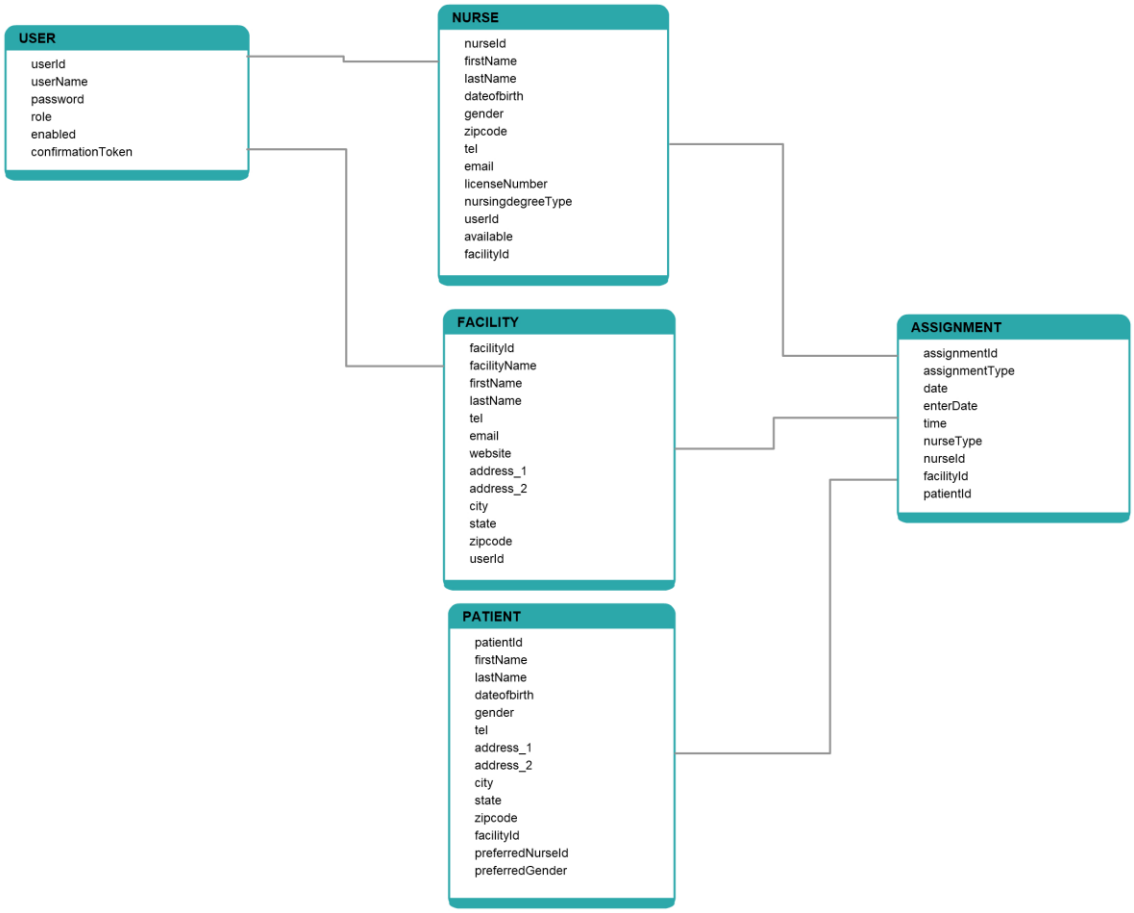

Figure 10. Database Design

CHAPTER 4

Implementation

Client-side Implementation

User Interface

User Interface usually refers to graphical design of the user interface. Focus is on designing a user-friendly UI which provides an easy way for the user to interact with the software.

Home Page. Home page is the first page the user interacts with. The Logo of the website is shown on the page header. A brief explanation of what this website does is displayed in the center of the page body. There is a hyper link to join and register in the system. Also, there is a log in form for those users who are already a member and want to log in to their dashboards. On page footer three main bullet points of the system is being mentioned and described.

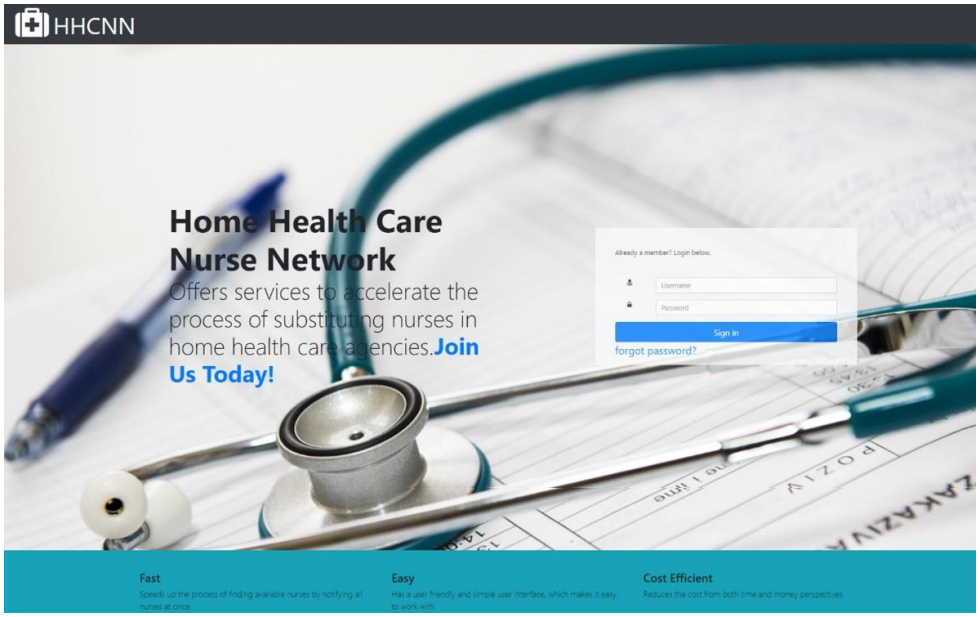

Figure 11. Home Page

Facility Registration. By clicking on Join Us Today link the following page appears and by choosing Facility and pressing Next, Facility Registration page appears. After filling out the form and submitting it, facility document gets saved in the database and a confirmation email is sent to the email provided.

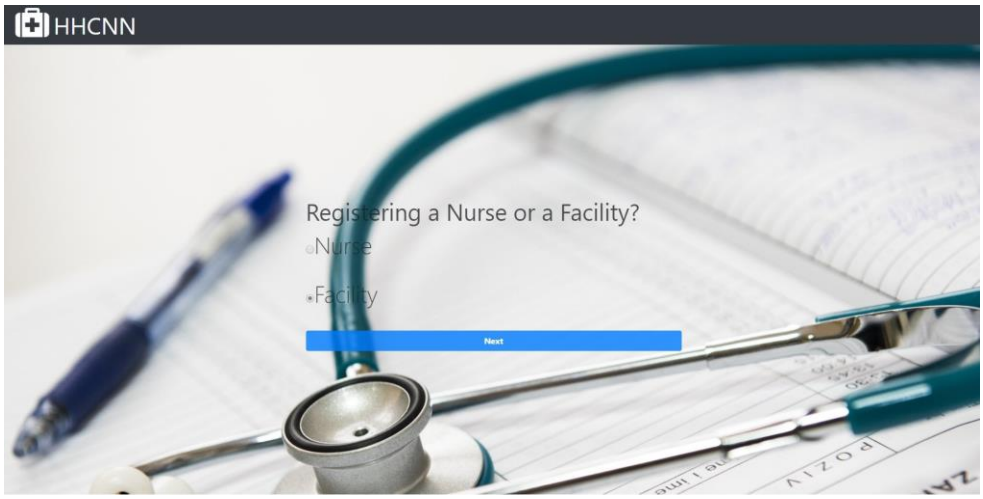

Figure 12. Facility Registration Page

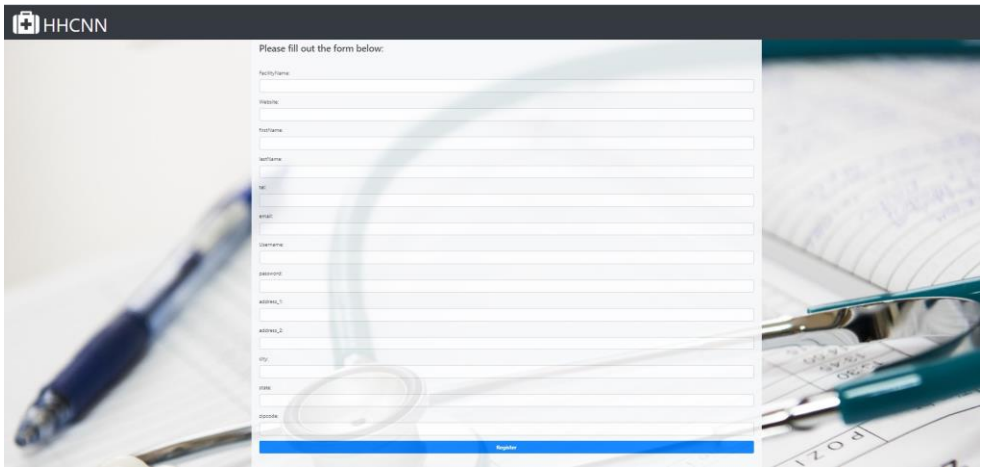

Figure 13. Facility Registration Form Page

Nurse Registration. By clicking on Join Us Today link the following page appears and by choosing Nurse and pressing Next, Nurse Registration page appears. After filling out the form and submitting it, a nurse document gets saved in the database and a confirmation email is sent to the email provided.

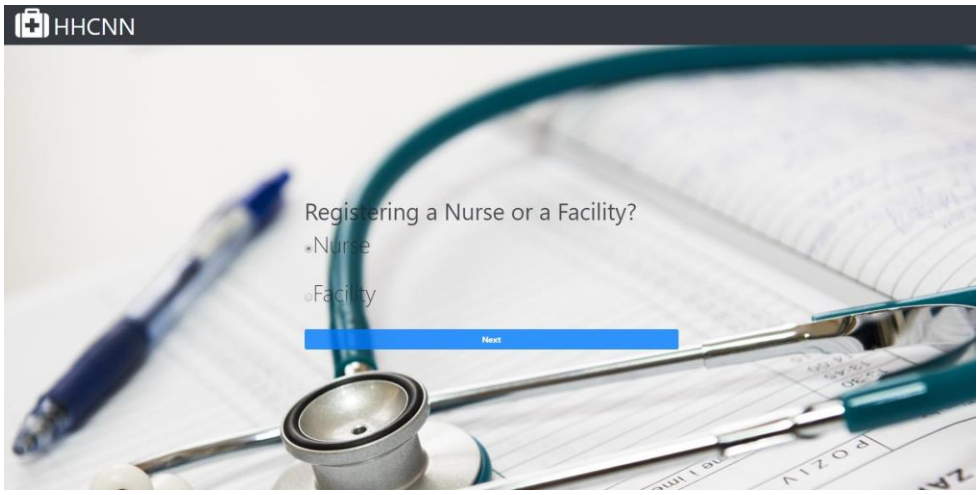

Figure 14. Nurse Registration Page

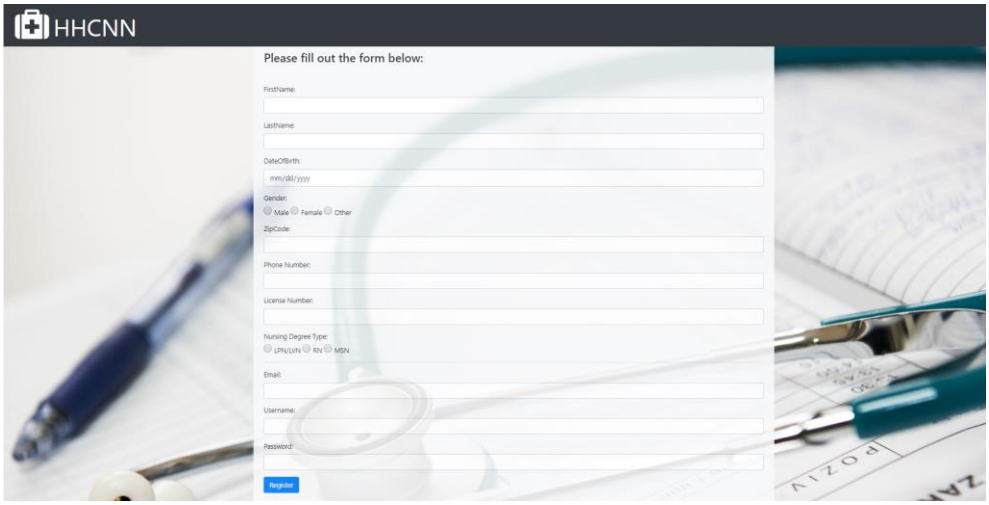

Figure 15. Nurse Registration Form Page

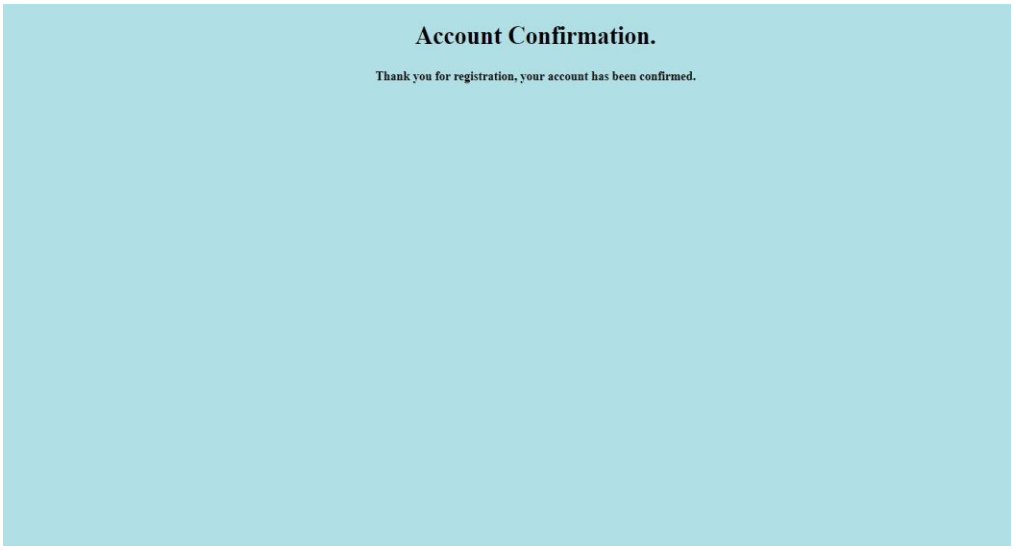

Figure 16. Account Confirmation Page

Confirmation Page. Once the user clicks on the confirmation link in their email Account Confirmation page appears. To confirm the account enabled field value in user document changes to true. Every time a user tries to log in the system this field gets checked to see if the account is a confirmed account or not. If it is confirmed, then the user can log in an access his/ her dashboard.

Facility Dashboard. After logging in as a facility, it navigates to Facility Dashboard page. On the navigation bar there is a logout link, which logs the user out and navigates to the home page. On the side bar there are four list items. In the center of the page the facility can see all the available assignments which haven't been filled out yet.

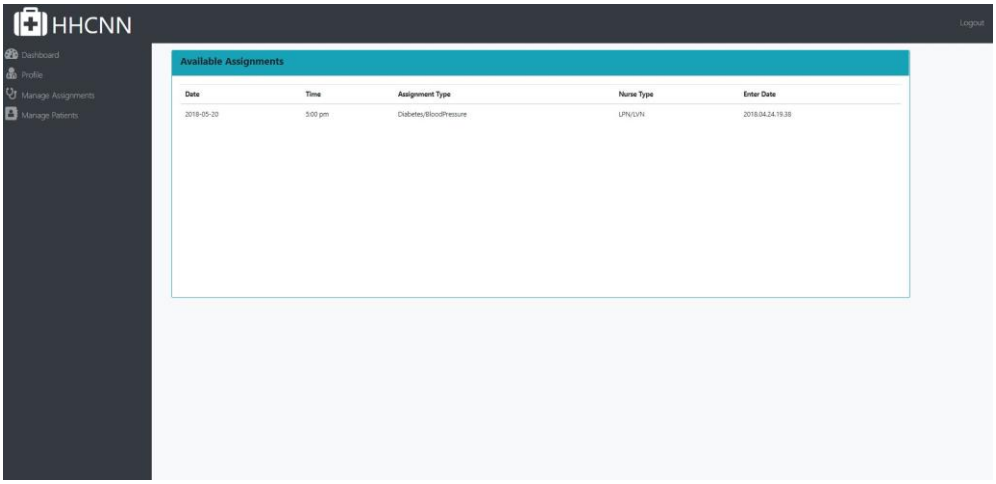

Figure 17. Facility Dashboard Page

Facility Profile. By clicking on Profile navigation item, it navigates to Profile page. The user can edit his/her profile and submit it by pressing Save button. If there is no need to edit the profile, the user can simply press Cancel button or Dashboard navigation item. Either way it gets navigated to the facility's dashboard.

Figure 18. Facility Profile Page

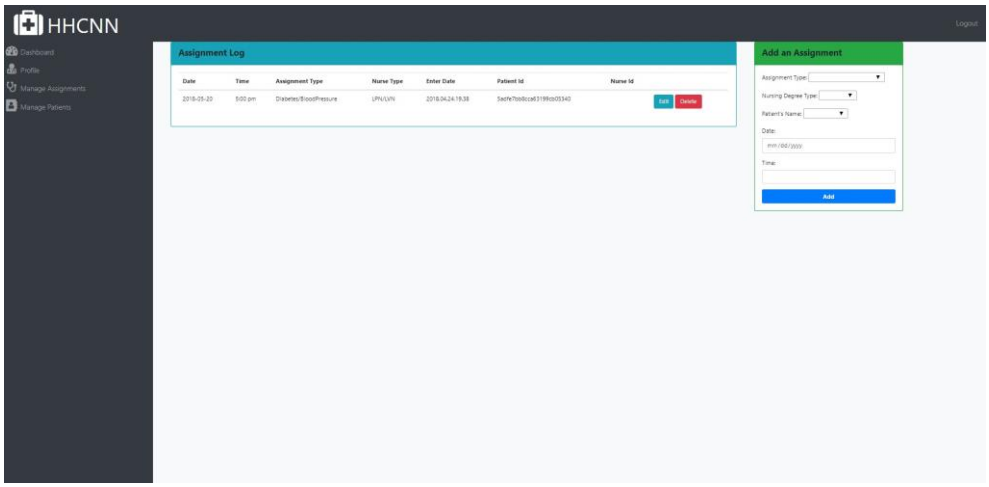

Figure 19. Facility Assignment Management Page

Assignment Management. Assignment Management page can be accessed by pressing Manage Assignment side navigation bar item. There are two cards displayed on this page. Assignment Log card shows a list of all assignment logs of the current facility. In addition, there are two buttons to Edit or Delete each of the assignments shown in the list. Also, there is another card to Add Assignment, which lets the facility to add a new assignment easily by choosing the type of assignment and nurse from the drop-down menu.

Patient Management. Patient Management page can be accessed by clicking on Manage Patient side bar navigation item. It includes a list of patients associated with the facility. Also, there is an option of deleting or editing each patient. There is another card on the right side the page to add a new patient. The patient can have preferred nurse gender or preferred nurse.

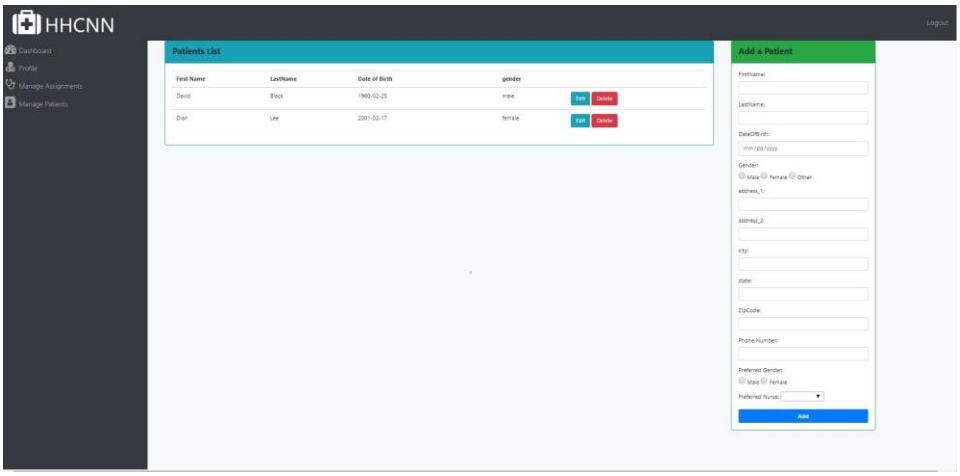

Figure 20. Facility Patient Management Page

Nurse Dashboard. Once a nurse logs in the system it redirects to Nurse Dashboard page. Like facility dashboard there is a logout link on the navigation bar on the page header. Also, a side navigation bar is located on the left side of the page, which has three navigation items. There is a checkbox to have Notify me option on and off.

On the body of the page there are two cards. Available Assignment card shows all the available assignments which the current nurse meets its criteria. There are checkboxes next to each assignment and by selecting the assignments and clicking on Submit button nurse's Id gets saved to the assignments selected. Another card is Upcoming Assignments, which displays all the future assignments the current nurse has accepted.

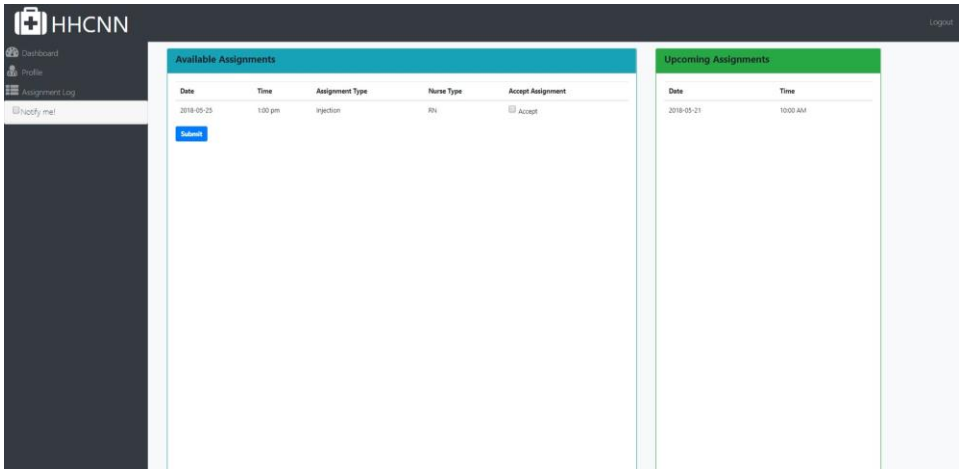

Figure 21. Nurse Dashboard Page

Nurse Profile. By clicking on Profile side bar navigation item, it navigates to Profile page. The user which here is a nurse can read and edit his/her profile and submit it by pressing Save button. To go back to Nurse Dashboard page simply press cancel or click on Dashboard on side bar.

Figure 22. Nurse Profile Page

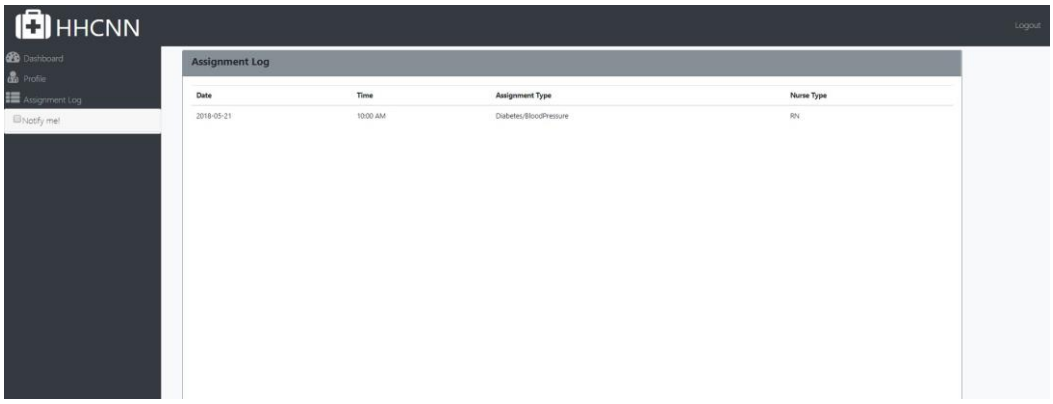

Figure 23. Nurse Assignment Log Page

Nurse Assignment Log. Nurse Assignment Log page can be accessed through Assignment Log on the side navigation bar. On the center of the page a nurse can see all assignments which he/she has ever accepted.

Angular Implementation

Using node package manager, Angular-CLI got installed. All components get created using Angular-CLI by using command *ng g c Component Name*.
Once a component is created four files get created automatically with types html, css, ts and spec.ts.

In this project we have thirteen components in addition to app component which is the root one. First component being discussed is Registration component. Inside its typescript file, it has a property for registration type. On the html page there are two radio buttons with values of nurse registration or facility registration form link. These values are bind to the property of registration type. There is a hyperlink with type of button which redirects to the link with registration type property value.

Nurse Registration and Facility Registration components both have properties with type of nurse object or facility object to store the form data in them. In each component there is a Save method to save the object to the database. To do so, Http

Client module has been imported from angular/common/http package. Http Client module provide an API to send requests such as get, post, update, and delete.

To post the object, we can send a URL and an object in the request. This way, the request goes to the controller classes in the server-side of the project and looks for the URL. Then it saves the object sent through the post request to the database. There is a need to subscribe to this request because the result of the request is an observable. An observable is a sequence of items that arrive over time and if we do not subscribe to it, the result won't be affective.

At the end the page gets redirected to the home page of the website using Router module. Router module needs to be imported from angular router library package. Router service provides navigate method to redirect the active router to the page needed.

Home component has a User property and gets username and password. It sends a post request to Login controller in the server-side to check the authentication. Then subscribes to the request. If the user is valid, it gets user object. In login method inside home component, user role gets checked. If the role equals to Admin it navigates to facility dashboard page. Otherwise it navigates to nurse dashboard page.

Facility Dashboard component has an assignments property. Since facility dashboard shows a list of available assignments once the page loads, Show Available Assignments method should get the data on page initialization. This method gets called inside ngOnInit() method, which is the first method that runs once the page loads. Show Available Assignments method sends a get request to backend code among with the id of the facility. The reason is that the facility needs to only see its own assignments. In the backend a query of assignment table will give the needed result. At the end we subscribe

to the request and save the data into assignments property and later in html bind it to ng model attributes on the table rows.

Nurse Dashboard component is similar to Facility Dashboard component. In addition, it need to have another get request to get all the upcoming assignments that the nurse has accepted. Also, needs a put request to accept assignments. By accepting an assignment, nurse Id gets saved to assignments document.

Facility and Nurse Profile components both need to have get and put requests. A get request to load the details of the user and an update request to save the changes to their profile. Patient and Assignment components have methods with get, update, and delete requests to manage patients and assignments.

By using angular 5, some conditions can be checked on client-side. Also, communicating with the server-side and accessing the database became easier.

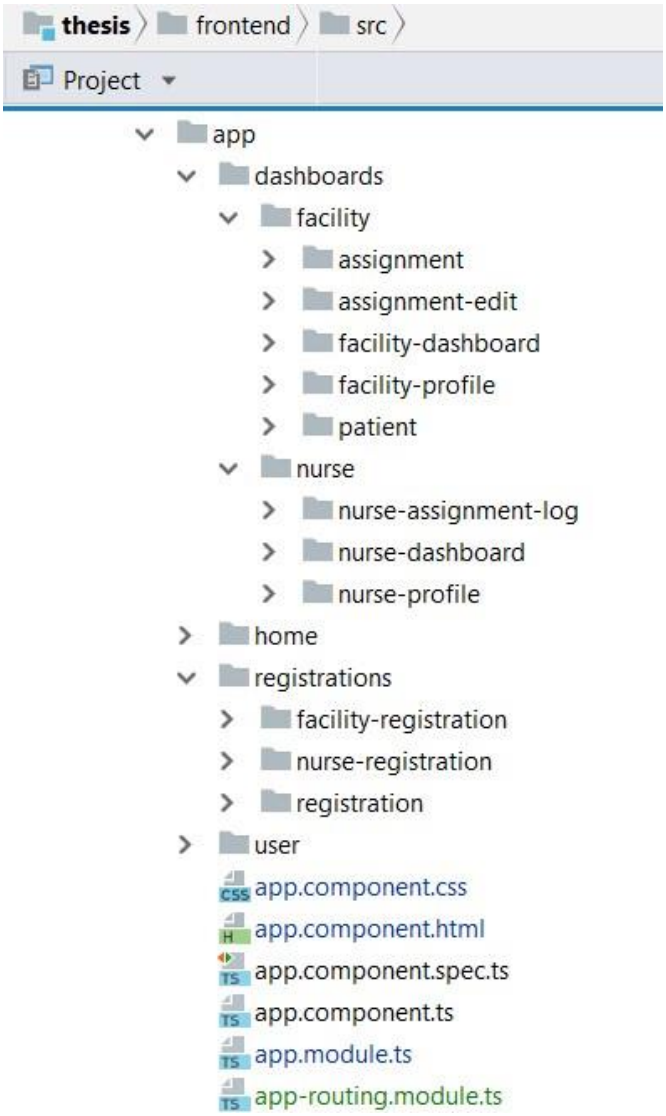

Figure 24. Angular Components

Server-side Implementation

To implement model part of MVC architecture, a data model class got created for each entity in the project. There are six different entities such as nurse, facility, mail, assignment, patient, and user.

Controller concept got implemented through controller java classes. Each controller class is annotated by @Controller. This annotation triggers Autoconfiguration inside Boot library of spring framework, which sets up Spring MVC and enables Tomcat support.

In addition to controller annotation there is @Request mapping annotation, which is for mapping web requests. It has different method signatures for get, post, delete, and update.

To implement business layer, java classes got created with @Component annotation. Component annotation marks the class as a bean to have the component-scanning mechanism of spring to put it in the application context.

Each Business class contains some methods which get called through controllers. Business classes need to be injected to controllers by @inject component. To use @inject, javax inject library must be declared as a Gradle dependency.

To implement Repository layer, java repository classes got created with annotation @repository. Repository annotation is a specialization of @Component. It has similar use and functionality. It takes care of communicating with the database.

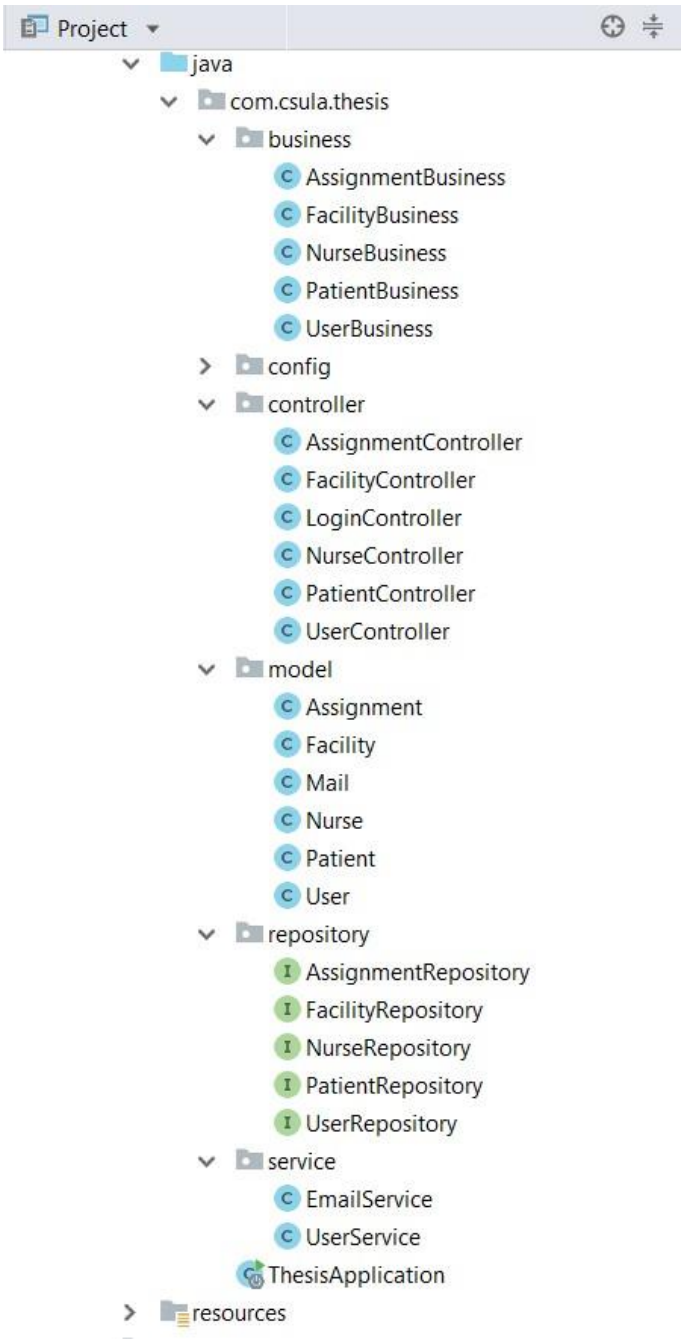

Figure 25. Java Classes

Database Implementation

Since in this project NoSQL database is being used, there is no need to manually create any tables or relationships as it is needed in relational databases. This characteristic speed up implementing and developing the project because no time will be used on database implementation.

Simply by defining Models in backend of the project, creating repository interface for each one and extending Mongo Repository, everything will be handled. There is no need to use MongoDB query language because Spring Framework Data MongoDB Repository has pre-defined methods for queries, create, read, update, and delete.

For each entity in the project a model class needs to be created. Each class includes all the attributes of the entity. Attributes of the model class are fields of the database document. Also, model classes need to have getters and setters for each attribute.

Every model class should have an id attribute. Id attribute should be annotated by @Id. There is no need to use any specific MongoDB annotation for this id field because it fits the standard name for MongoDB id.

As it has been mentioned before in chapter 2, MongoDB stores data in collections. Spring Data MongoDB maps the model class into a collection in MongoDB. Only if there is a need to have a different collection name than the model class, @Document annotation can be used.

In this project there are five data collections such as Nurse, Facility, Patient, User, and Assignment. Data gets saved into these collections once the related methods for creating or updating into MongoDB gets called.

CHAPTER 5

Conclusion

Developing this project started in Spring 2017. Some technologies and tools used in developing of this project got changed during the development. The reason for this change was using newer technologies to take benefits of their features.

Frontend technology used in this project changed from AngularJS to Angular 5 in Winter 2017. In the beginning STS (Spring Tool Suite) was used as the IDE, but later in the process it had to be switched to IntelliJ IDEA Ultimate to avoid some configuration problems of integrating with Angular 5.

HHCNN is so flexible because of using NoSQL database. Changes can be made with less effort comparing with RDBMS and fields and entities can be added to it easily. There is no need to make any changes to database implementation.

This project is low weight because of using Single Page architecture for client-side of the project. The reason is that there is no need to load a new html page every time there is a need to navigate to another page. Simply a template gets replaced by the previous template.

In conclusion HHCNN meets the basic needs of home health care facilities to fill out assignments in cases of nurse absence in a fast peace. Also, HHCNN is expandable and new features, entities and functionalities can easily be added to it.

REFERENCES

(n.d.). Retrieved from https://angular.io/guide/architecture

(n.d.). Retrieved from https://angular.io/guide/router

Free Image on Pixabay - Medical, Appointment, Doctor. (n.d.). Retrieved from https://pixabay.com/en/medical-appointment-doctor-563427/

IntelliJ IDEA: The Java IDE for Professional Developers by JetBrains. (n.d.). Retrieved from https://www.jetbrains.com/idea/

Learn Version Control with Git. (n.d.). Retrieved from https://www.git-tower.com/learn/git/ebook/en/desktop-gui/basics/why-use-version-control

O. (2017, November 14). Angular 5 Release: What's New? Retrieved from https://auth0.com/blog/whats-new-in-angular5/

Spring Security and Angular. (n.d.). Retrieved from https://spring.io/guides/tutorials/spring-security-and-angular-js/#add-a-home-page

Studio 3T. (n.d.). Retrieved from https://robomongo.org/

What is Build Tool? - Definition from Techopedia. (n.d.). Retrieved from https://www.techopedia.com/definition/16359/build-tool

What's home health care? (n.d.). Retrieved from https://www.medicare.gov/what-medicare-covers/home-health-care/home-health-care-what-is-it-what-to-expect.html