

CMPE 313 Software Engineering Project

Patient Management System

Renas Barış Özkal

Section 04

Source Code : <https://github.com/rbarisozkal/Patient-Management-system>



What are we aiming at with this Project?

- The purpose of the project is to help patients to get an appointment from a website instead of going to the hospital physically. This brings a benefit especially for elders to make their appointment easily. Therefore, our scope of the project consists of all people who need an appointment and have access to the internet.
- We are providing here a web-site that consists with an interface for customers (patients) to log-in, see the symptoms and get an appointment according to this information.
- We use agile development approach for our software and conducted several Scrum Meetings. With the help of our Scrum Master who is Renas Barış Özkal, we finished our software project with 3 sprints.




High level architecture of our software

Model-View Controller Architectural Pattern (MVC)

MVC is widely used to develop model user interfaces. Since we also design a web project, the MVC model provides us with a structure suitable for our system. We can define components of MVC in detail:

Model: The model can consist of multiple layers by the preferences of the project members. Division into multiple parts is useful in the management of the Project. In the Patient Management system, we created different modules such as component and view to manage the system and associated data.

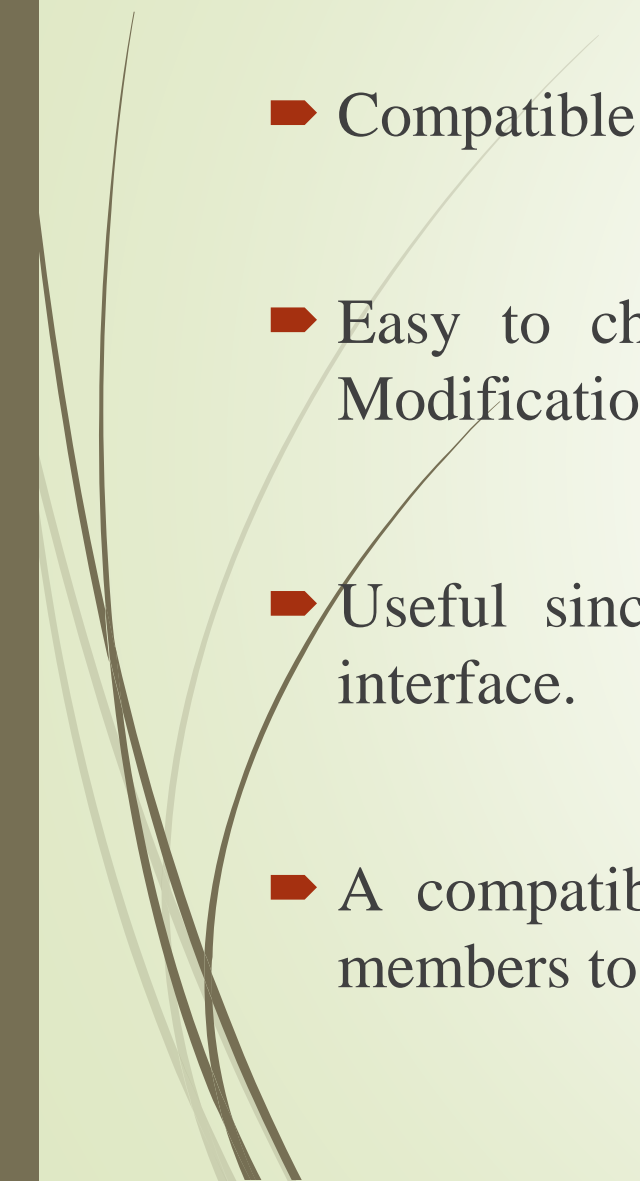


View: This is the part we created our interfaces that is going to be presented to the user. We include HTML files and other required files that is about JavaScript, CSS, etc. In our project, we divided parts of the project into components (foldering). By making this, we keep our code understandable, readable and organized.

Controller: It evaluates requests of the user and specifies which operation is going to be done and view that returns to the user. For instance, when our patient log-in the system to make an appointment (request), controller assesses the request of the patients and enables them the interface of appointment page (response).

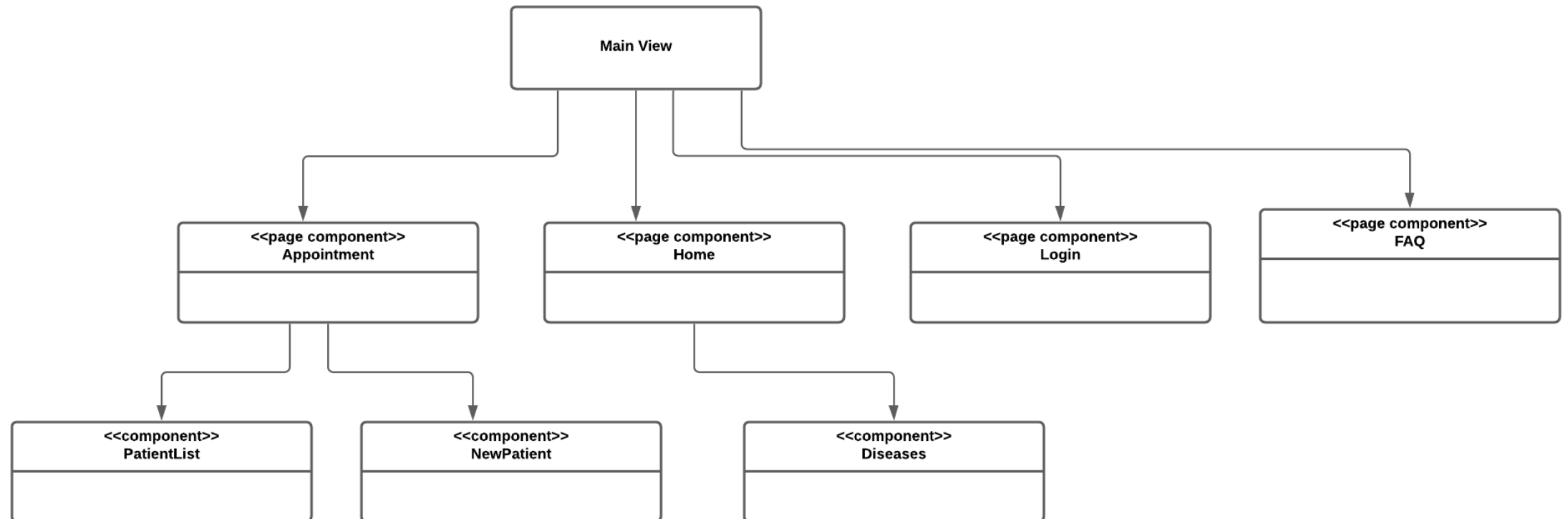


Why we decided MVC

- Compatible with web - development projects.
 - Easy to change while focusing on specific part of the project. Modification in the project does not require dependency at all.
 - Useful since we try to create a small web page that consist of interface.
 - A compatible architecture pattern which enables multiple project members to cooperate easily.
- 

Architectural view of the project

- **Logical view:** key abstractions in the system as objects or object classes.



Low level architectural decisions

Software Design Patterns we used,

- **Factory Method**: creational design pattern that allows different components to create objects for providing an interface. For example, in our project, we have views and components that can divide into sub-parts. For instance, in components, we hold patient list and illnesses that creates objects to contribute main component. Similarly, in the view, we have appointment, home and login and FAQ parts that are different parts that benefits the overall system. Component and view are dependent together and, by cooperating together, they occur just as a factory.

Low level architectural decisions

Software Design Patterns we used;

- ➡ **Adapter:** structural design pattern that enables objects to allow two different (not compatible among them) interfaces to work together. In our project, when we make an appointment in the system, we save this appointment such as patient data to our database. However, we can not apply this patient object in SQL and in the JavaScript files similarly. Therefore, we used adapter pattern that includes JSON to implement our patient objects for SQL operations in the development file.

Other details

- In the project, we used Visual Studio Code as an integrated development environment that enables to use various programming languages at the same software.
- At first, it was planned to use React.js. After that, we decided to use Vue.js which is the framework of JavaScript. Similarly, we planned to create a database to hold appointment data of patients at the beginning. However, it is required to focus on the fronted development of the project, we concentrated on appearance of interface and other requirements of the project.

Specific programming technologies we used/planned to use

- At the beginning of the Project, we planned to use a database system to hold specific information about patients, doctors, and hospitals in Turkey. However, since code interfaces and tools that provides the requirements is enough, we focused on fronted development of the project.
- As a programming technologies, we used vue js, javascript, html, and css by using Visual Studio Code environment.



Functional Requirements

- Our system provides an information about the diseases and symptoms to increase awareness of the patient before start processing of appointment.
- Taking an appointment from the desired hospital, hour, and doctor.
- Our software provides a FAQ page which includes common and popular questions about the system for the people who have confusions and questions about the system.
- There should be an internet connection for any device that has access to the site. While accessing to the site, there is no limitation about web browser type.

Non-Functional Requirements

► Performance Requirements:

For **Response Time**, our system gives a response time to the user between 0 and 1 second for any request. For instance, pressing log-in, get appointment and FAQ button.

For **Capacity**, our software supports multi appointment. That means, more than one patients can get an appointment at the same time. In the appointment page of our interface, as a result, more than person's information (doctor name, hospital name, appropriate hour) about appointment is printed.

► Safety and Security Requirements:

When patients try to log-in into site, their password is blurred, so that its invisibility provides **safety and security**.

After the appointment is taken, our software interface does not print ID Number of the patient which is a required data for log-in. By this way, after we take the information about ID Number, we do not include an ID Number in the list of appointments of patients to provide both **security and safety**.



■ Thank you for listening us.