Hospital Management App

Hospital Management App

Submitted By-

- Adesh Awachat

Project Briefing

This application allows patients to check their appointments online using their login credentials and also allows hospital administration to post the available dates.

It also manages administrative controls along with standard access to doctors and patients for the appointment related facilities.

DoctorController Result

This microservice requests doctors appointment by giving patient name as a parameter.

This microservice can access data using the doctors credentials only.

For example- If doctor wants to get the details about patients, he/she can access by their name only.

PatientController

This microservice allows the patient to post, update and delete data regarding his/her health.

This microservice can manage data by patient credentials.

For example- If patient of the doctor wants to manage data he/she can access by their patient login credentials.

Prescription Controller

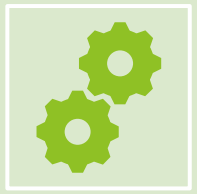
This microservice allows the doctor to post, update and delete data regarding prescription.

This microservice can manage data of prescription credentials by doctor.

For example- If doctor/patient wants to access their prescription then doctor/patient can access using their login credentials.

Tools Used

- Docker
- Java
- Spring Boot
- MongoDB



Maven is a project management and comprehension tool that provides developers a complete build lifecycle framework. Development team can automate the project's build infrastructure in almost no time as Maven uses a standard directory layout and a default build lifecycle.

In case of multiple development teams environment, Maven can set-up the way to work as per standards in a very short time. As most of the project setups are simple and reusable, Maven makes life of developer easy while creating reports, checks, build and testing automation setups.

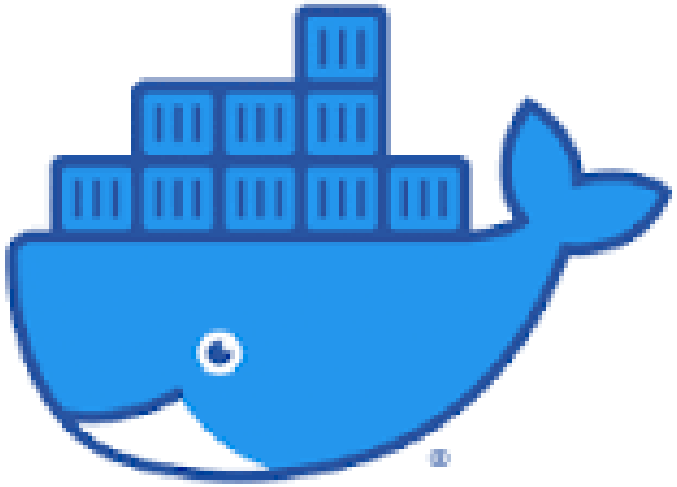
Maven provides developers ways to manage the following –

Builds
Documentation
Reporting
Dependencies
SCMs
Releases
Distribution
Mailing list

Java

- Java is a class-based, object-oriented programming language and is designed to have as few implementation dependencies as possible. A general-purpose programming language made for developers to write once run anywhere that is compiled Java code can run on all platforms that support Java.

Docker



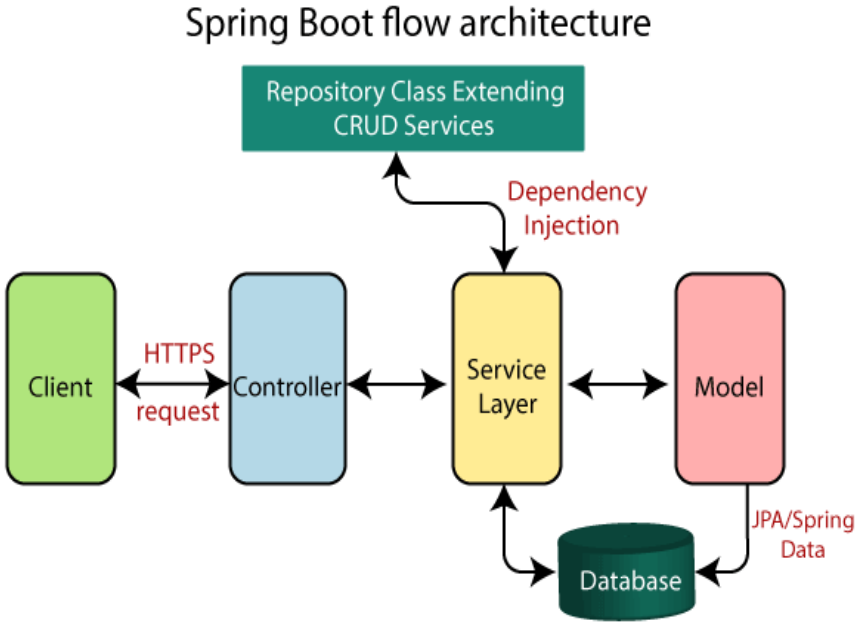
- Docker is a software platform that allows you to build, test, and deploy applications quickly. Docker packages software into standardized units called containers that have everything the software needs to run including libraries, system tools, code, and runtime. Using Docker, you can quickly deploy and scale applications into any environment and know your code will run.

What is Spring Boot?

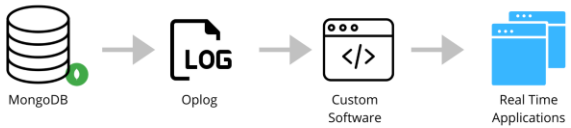
Spring Boot is an open-source Java based framework used to create a micro service. It is developed by Pivotal Team and is used to build stand-alone and production ready spring applications.

Spring Boot Main Components

- Spring Boot Starters- combine a group of common or related dependencies into single dependency
- Spring Boot AutoConfigurator- reduce the Spring Configuration
- Spring Boot CLI- run and test Spring Boot applications from command prompt
- Spring Boot Actuator- provides EndPoints and Metrics


Spring Boot flow architecture
Repository Class Extending CRUD Services
Dependency Injection
Client
HTTPS request
Controller
Service Layer
Model
JPA/Spring Data
Database

MongoDB



- MongoDB is a NoSQL database which stores the data in form of key-value pairs. It is an Open Source, Document Database which provides high performance and scalability along with data modelling and data management of huge sets of data in an enterprise application. MongoDB also provides the feature of Auto-Scaling.

mongoDB®

InjectMocks

- Creates objects and injects all dependencies annoted with @Mocks. @Mock allows shorthand creation of objects required for testing.
- Minimize repetition and improves readability.

```
public class DoctorControllerTest {

    @InjectMocks
    DoctorController doctorController;
    @Mock
    AppointmentRepository appointmentRepository;

    @Mock
    Appointment appointment;
    @BeforeEach
    void setUp(){
        appointmentRepository= Mockito.mock(AppointmentRepository.class);
        appointment= Mockito.mock(Appointment.class);
    }
```

Use Cases

- This application can be used for accessing and managing prescription and appointment schedule.
  - Doctor
  - Patient


THANK YOU