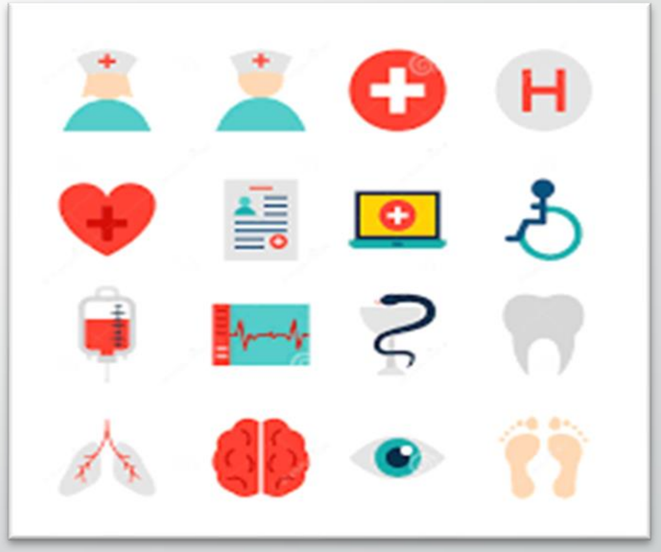
HOSPITAL MANAGEMENT APP

- Prachi Apale

Project Briefing

- This application allows the patients and the doctors to schedule various appointments in a predetermined format using their specific login credentials.
- This application stores various important data related to the appointment as well the prescriptions.
- Administrative controls are managed and limited access is granted to the patients and the doctors for the appointment related activities.



Tools Used

- Apache Maven
- Java
- Spring Boot
- MongoDB
- Docker
- IntelliJ IDEA

Maven

- 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.
- Maven provides developers ways to manage the build, documentation, reporting, dependencies, SCMs, releases, distribution and mailing lists.

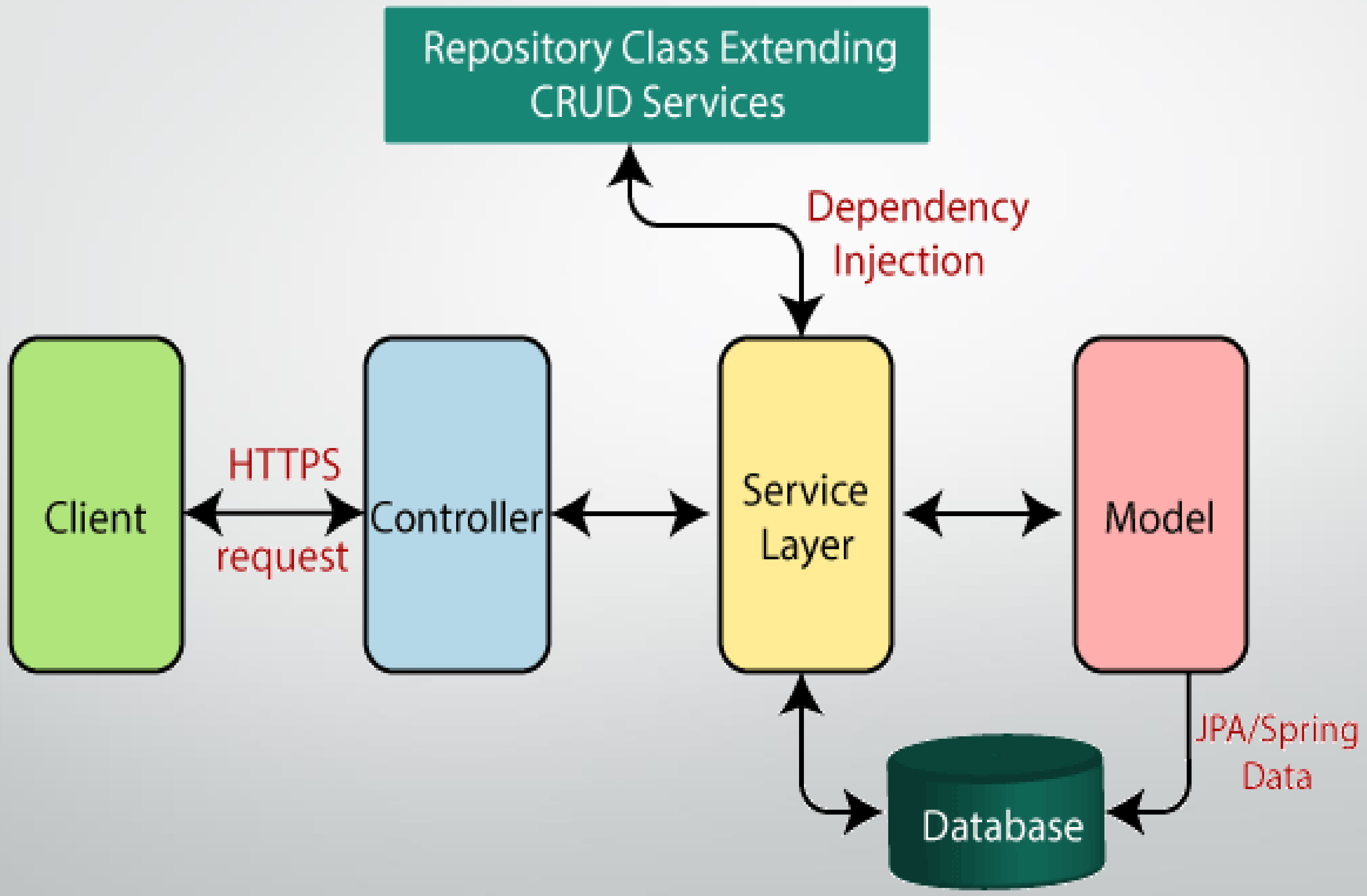
Java

- 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.

Spring Boot

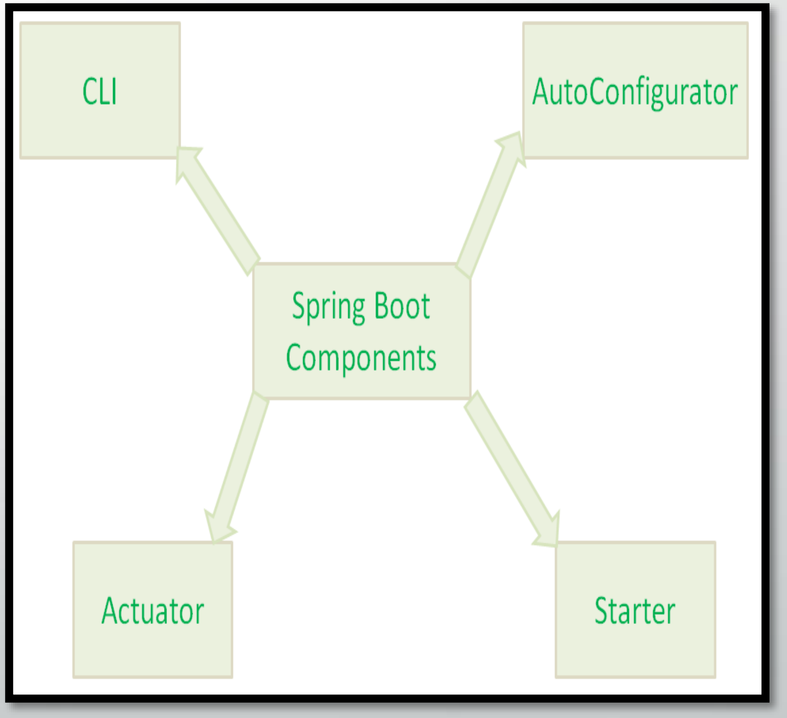
- Spring Boot makes it easy to create stand-alone, production-grade Spring based Applications that you can "just run".
- We take an opinionated view of the Spring platform and third-party libraries so you can get started with minimum fuss. Most Spring Boot applications need minimal Spring configuration.

Spring Boot flow architecture



Spring Boot Main Components

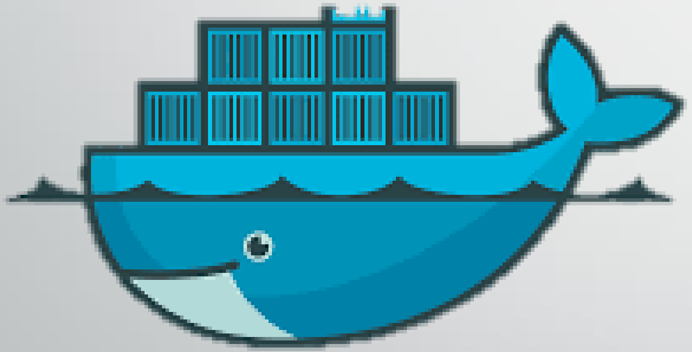
- **Spring Boot Starters** - The main responsibility of Spring Boot Starter is to combine a group of common or related dependencies into single dependencies.
- **Spring Boot AutoConfigurator** - The main responsibility of Spring Boot AutoConfigurator is to reduce the Spring Configuration.
- **Spring Boot CLI** - Spring Boot CLI(Command Line Interface) is a Spring Boot software to run and test Spring Boot applications from command prompt.
- **Spring Boot Actuator** - Spring Boot Actuator components gives features such as providing Management EndPoints to Spring Boot Applications.



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.

Docker

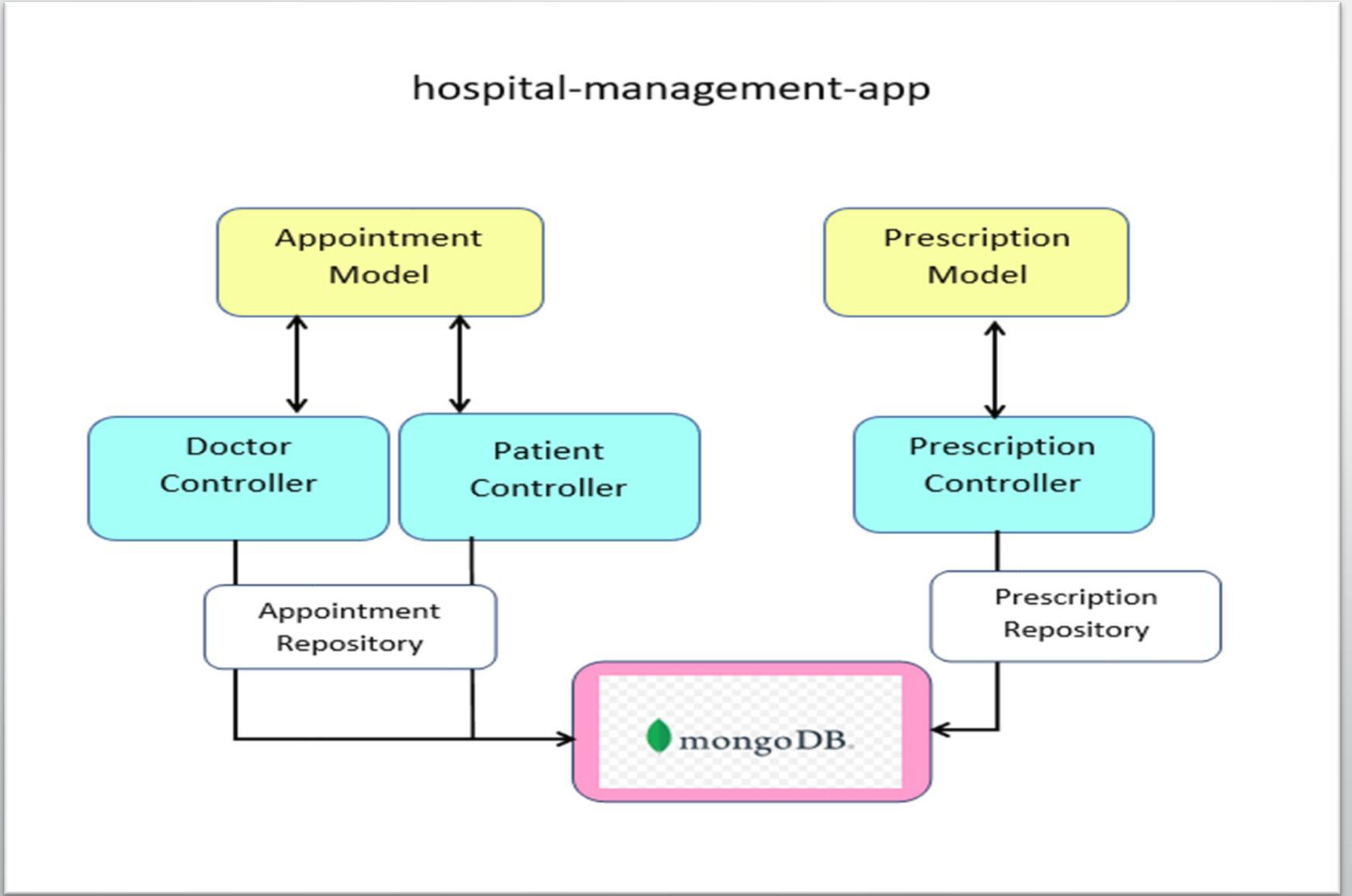


- Docker is a software platform that allows you to build, test, and deploy applications quickly. Docker packages software into standardized units called containers that have everything the software needs to run including libraries, system tools, code, and runtime. Using Docker, you can quickly deploy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

IntelliJ IDEA

- **IntelliJ IDEA** is an integrated development environment (IDE) written in Java for developing computer software written in Java, Kotlin, Groovy, and other JVM-based languages. It is developed by JetBrains (formerly known as IntelliJ) and is available as an Apache 2 Licensed community edition, and in a proprietary commercial edition. Both can be used for commercial development.

App Architecture



Model

Appointment

- This is the first model of the microservice that stores important data related to appointments such as appointmentId, patientName, doctorName, date as well as the prescription object.
- The two controllers namely DoctorController and PatientController refer to this model.

Prescription

- This is the second model of the microservice that stores important data related to prescriptions such as prescriptionId, patientName, doctorName, appointmentId and its description.
- The controller called PrescriptionController refers to this Prescription model.

Controllers

This microservice in this application has three controllers as follows:

- DoctorController
- PatientController
- PrescriptionController

DoctorController

The DoctorController exposes the following two Rest EndPoints:

- /doctor/save – This is a POST method which is basically used when a doctor wants to schedule an appointment and he provides the specific data which is then stored into the MongoDB.
- /doctor/doctorappointment – This is GET method which is used to view the appointment data that is stored by the doctor. It uses the doctorName as a parameter.

PatientController

The PatientController exposes the following two Rest EndPoints:

- /patient/save – This is a POST method which is basically used when a patient wants to schedule an appointment and he provides the specific data which is then stored into the MongoDB.
- /patient/myappointment – This is GET method which is used to view the appointment data that is stored by the patient. It uses the patientName as a parameter.

PrescriptionController

The PrescriptionController exposes the following two Rest EndPoints:

- /saveprescription – This is a POST method which is basically used when details about a specific prescription are to be stored. This data is saved into the MongoDB.
- /viewprescription – This is GET method which is used to view the prescription data that is stored in the database. It uses the patientName as a parameter.

Repositories

There are two repositories in this microservice. They are as follows:

1) AppointmentRepository – This repository is basically used to store the appointment data into the mongoDB. It extends the MongoRepository and consists of various entities that are used to store and retrieve the appointment data.
2) PrescriptionRepository - This repository is used to store the prescription data into the mongoDB. Like the AppoinmentRepository, this too extends the MongoRepository and it consists of various entities that are used to store and retrieve the prescription data.

Security

- The security aspect in this microservice is provided by the WebSecurityConfigurerAdapter.
- There are two roles namely PATIENT and DOCTOR, each of which has their own specific login credentials.
- The PATIENT user can access the Swagger web page along with the PatientController and the PrescriptionController.
- Whereas the DOCTOR user can access the Swagger web page, DoctorController and the PrescriptionController.

Use Cases

Following are the various use cases of the hospital management app :

- Avoid errors and track every single detail.
- Improved clinical decision-making.
- Improve data security.
- Establish your hospital as technically advanced.
- Efficient and accurate administration of every doctor, medical history of patient and distribution of medical aid.

Thank You!