A CAPSTONE PROJECT REPORT ON

DoConnect

Submitted by

C17- Group 2

Mentor:

1.Gyanendra Singh Yadav

2.Nuthanpreeth M

Group members:

1. TALAGANA SRIKANYA
2. VALETI BRAMA NAIDU
3. ANUJ KUMAR SINGH

PROBLEM STATEMENT

DoConnect is a popular Q and A form in which techniques questions was asked and answered.

FUNCTIONAL REQUIREMENTS:

User Stories:

1. As a user I should be able to login, Logout and Register into the application.
2. As a user I should be able to ask any question under any topic.
3. As a user I should be able to search the question on any string written in search box.
4. As a user I should be able to Answer any question asked
5. As a user I should be able to answer more than one question and more than one time.
6. As a user I should be able to chat with other users.
7. As a user I should be able to upload images to refer.

Admin Stories:

1. As an Admin I should be able to login, Logout and Register into the application
2. As an Admin I should be able to get mail as soon as any new Question is asked or any Answers given.
3. As an Admin I should be able to approve the question and Answer. Any Question or Answer will be visible on the platform only if it is approved.
4. As an Admin I should be able to delete inappropriate Questions or Answers.

SYSTEM SPECIFICATIONS

SOFTWARE REQUIREMENTS:

Technologies:

- Angular
- Java Spring Boot

Languages:

- TypeScript
- Java
- SQL Queries

IDE:

- Eclipse
- Vs code
- MySQL

HARDWARE REQUIREMENTS:

Operating System:

- Windows 7/8/10/11
- Linux distros
- MacOS X or later.

Processor:

- Intel or AMD dual core x86 processor.

Ram:

- 2 GB or above.

Hard disk:

- 500 MB of free disk space or more.

ABSTRACT

The Community Question Answering (CQA) websites that w sick give a huge number of inquiries and answers made by CQA clients to give rich wellsprings of data that are not accessible on web search engines and QA websites, have gotten extremely famous. Web clients can search for answers to their inquiries on CQA websites, however, they frequently need to (I) wait for quite a long time until other CQA clients submit answers to their important inquiries that give even improper, irritating, or spam, or (ii) reactions and limited answer sets made by CQA websites because of the specific closeness utilized and set between chronicled questions and client created questions. Critical upgrades in acquiring top notch answers to client Q question on the CQA website, we are presenting a CQA separating program, called Q AR. At QAR you first locate a bunch of CQA QS inquiries that are a similar Q, or Q-as per its determinations. In QAR you select as answers to Q significant level answers (among these questions in QS) in light of various scores of likenesses and length of answers.

ANGULAR ARCHITECTURE

Angular is a platform or framework to build client-based applications in HTML and TypeScript. It is written in TypeScript. It implements core and optional functionality as a set of TypeScript libraries that are imported into applications.

There are main eight blocks of Angular:

- Module
- Component
- Metadata
- Template
- Data Binding
- Service
- Directive
- Dependency Injection

Module:

Angular apps are modular and Angular has its own modularity system called Angular modules or Ng Modules. Every Angular app has at least one Angular module class, the root module, conventionally named AppModule. While the root module may be the only module in a small application, most apps have many more feature modules, each a cohesive block of code dedicated to an application domain, a workflow, or a closely related set of capabilities.

Ng Module is a decorator function that takes a single metadata object whose properties describe the module. The most important properties are:

- **Declarations** - the view classes that belong to this module. Angular has three kinds of view classes: components, directives, and pipes.
- **Exports** - the subset of declarations that should be visible andusable in the component templates of other modules.
- **Imports** - other modules whose exported classes are neededby component templates declared in this module.
- **Providers** - creators of services that this module contributes to the global collection of services; they become accessiblein all parts of the app.
- **Bootstrap** - the main application view, called the root component, that hosts all other app views. Only the root module should set this bootstrap property.

Component:

Components are the most basic UI building block of an Angular app. An Angular app contains a tree of Angular components. Angular components are a subset of directives, always associated with a template. Unlike other directives, only one component can be instantiated for a given element in a template.

Metadata:

Metadata is used to decorate a class so that it can configure the expected behavior of the class. Following are the different parts for metadata. Annotations - These are decorators at the class level. This is an array and an example having both the @Component and @Routes decorator.

Template:

A template is a form of HTML that tells Angular how to render the component. Views are typically arranged hierarchically, allowing you to modify or show and hide entire UI sections or pages as a unit. The template immediately associated with a component defines that component's host view.

Data Binding:

Data binding in AngularJS is the synchronization between the model and the view. When data in the model changes, the view reflects the change, and when data in the view changes, the model is updated as well.

Service:

Service is a broad category encompassing any value, function, or feature that your application needs. Almost anything can be a service. A service is typically a class with a narrow, well-defined purpose. It should do something specific and do it well.

Examples include:

- logging service
- data service
- message bus
- tax calculator
- application configuration

There is nothing specifically Angular about services. Angular has no definition of a service. There is no service base class, and no place to register a service. Yet services are fundamental to any Angular application. Components are big consumers of services.

Directive:

Directives are classes that add new behavior or modify the existing behavior to the elements in the template. Basically, directives are used to manipulate the DOM, for example adding/removing the element from DOM or changing the appearance of the DOM elements.

Dependency Injection:

Dependency injection, or DI, is a design pattern in which a class requests dependencies from external sources rather than creating them. Angular's DI framework provides dependencies to a class upon instantiation. Use Angular DI to increase flexibility and modularity in your applications.

SPRINGBOOT ARCHITECTURE

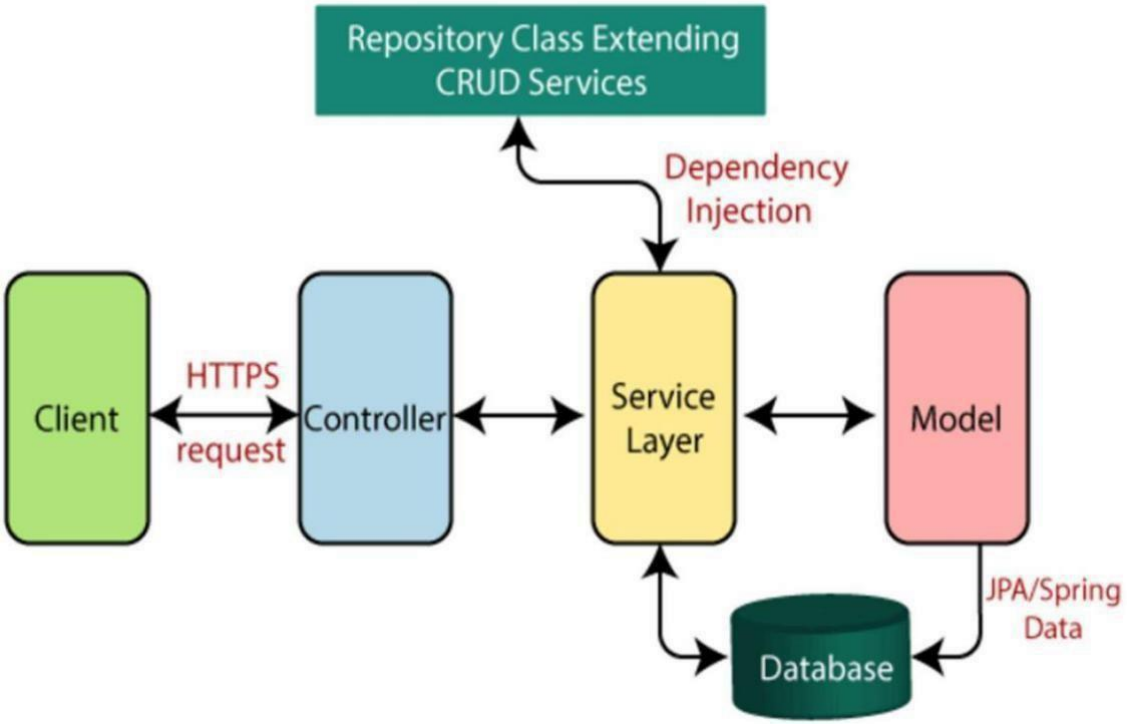
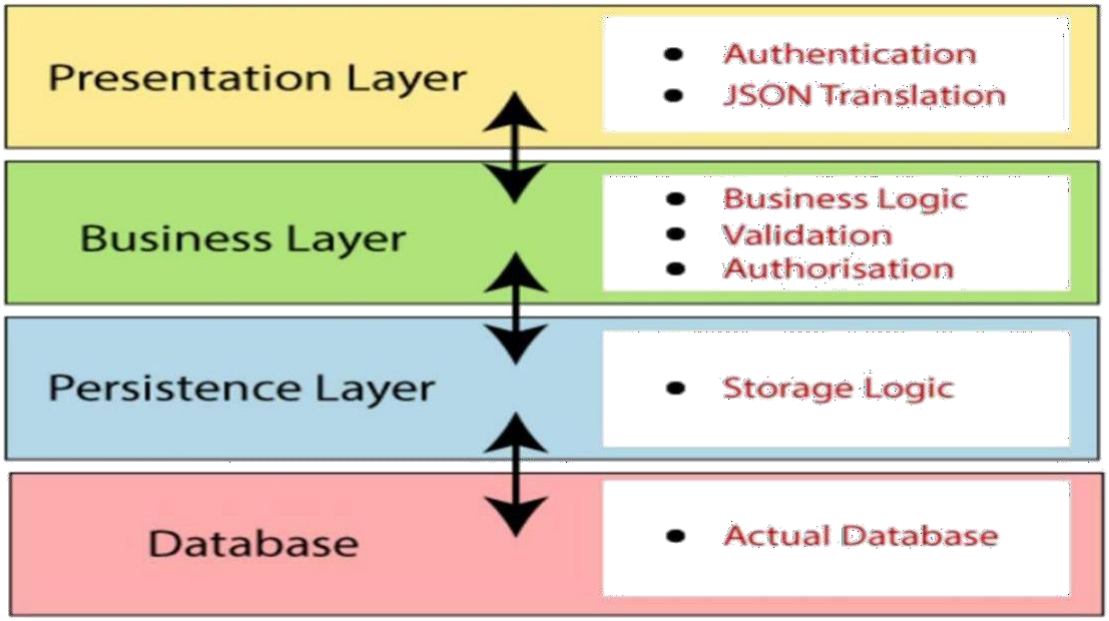


FIG: SPRINGBOOT ARCHITECTURE

The spring boot consists of the following four layers:

- **Presentation Layer:** Authentication & Json Translation.
- **Business Layer:** Business Logic, Validation & Authorization.
- **Persistence Layer:** Storage Logic.
- **Database Layer:** Actual Database.



Presentation Layer:

The presentation layer is the top layer of the spring boot architecture. It consists of Views. i.e., the front-end part of the application. It handles the HTTP requests and performs authentication.

Business Layer:

The business layer contains all the business logic. It consists of services classes. It is responsible for validation and authorization. The persistence layer contains all the database storage logic.

Persistence Layer:

The persistence layer contains all the storage logic and translates business objects from and to database rows. Database Layer: In the database layer, CRUD (create, retrieve, update, delete) operations are performed.

Database:

The database layer contains all the databases such as MySQL, MongoDB, etc. This layer can contain multiple databases. It is responsible for performing the CRUD operations.

MICROSERVICE ARCHITECTURE

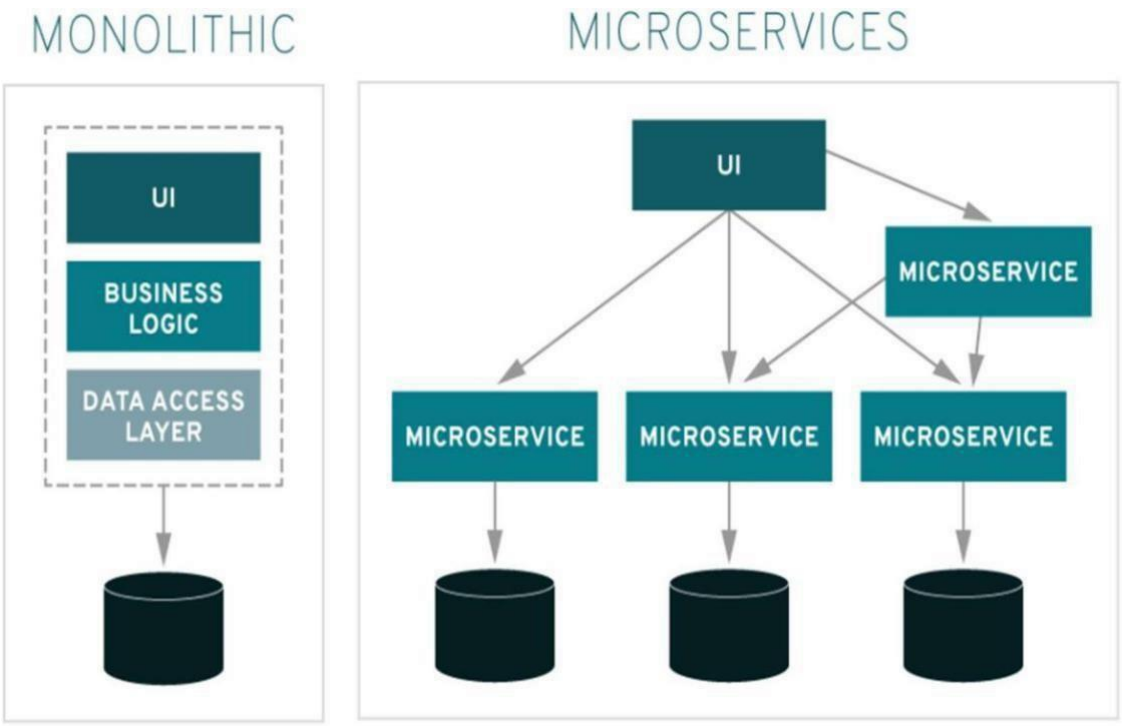


FIG: MICROSERVICE ARCHITECTURE

- Typically, micro services are used to speed up application development.
- Micro services architectures built using Java are common, especially Spring Boot ones.
- Microservices architecture (often shortened to microservices) refers to an architectural style for developing applications. Microservices allow a large application to be separated into smaller independent parts, with each part having its own realm of responsibility.

- When you are ready to start adopting a microservices architecture and the associated development and deployment best practices, you'll want to follow the three C's of microservices: componentize, collaborate, and connect.

DATABASE ARCHITECTURE

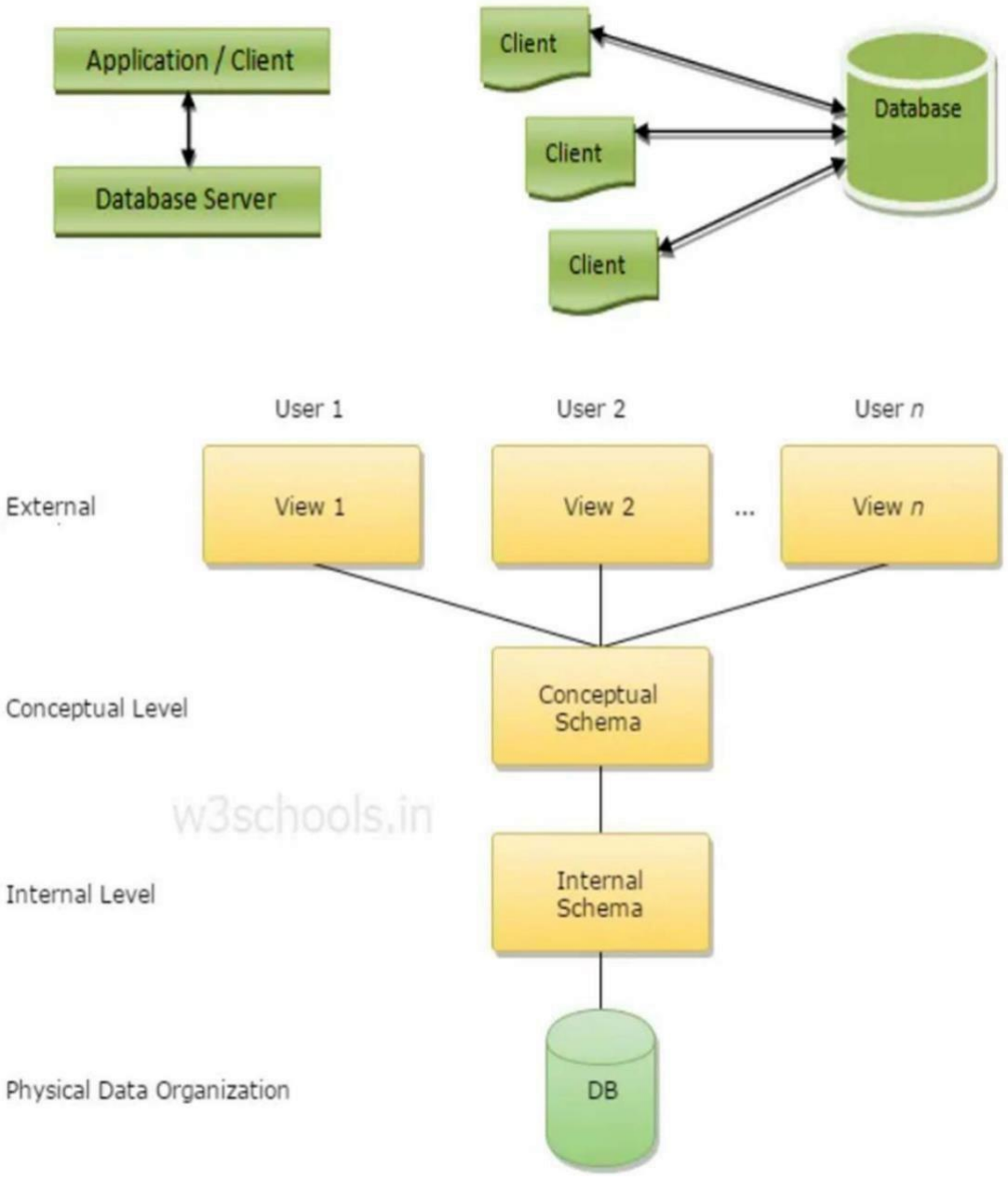


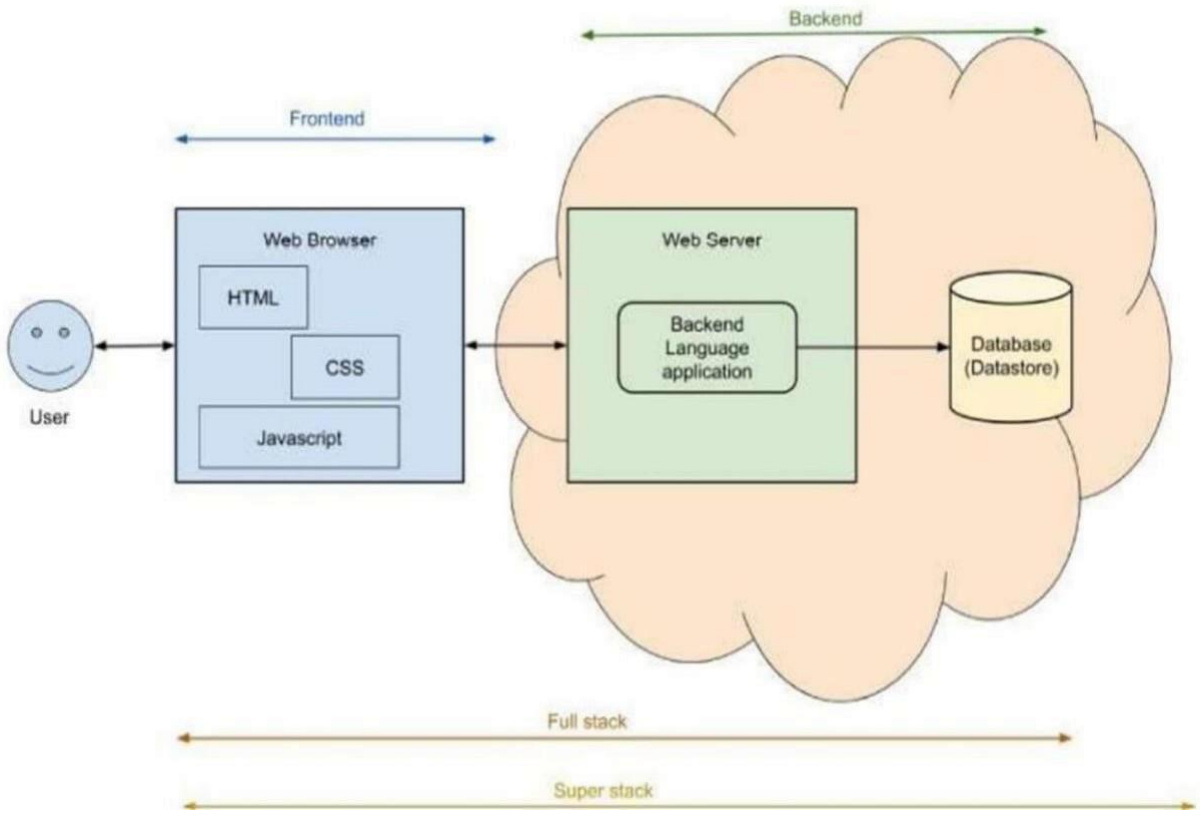
FIG: ARCHITECTURE OF DATABASE SYSTEM

- A Database Architecture is a representation of DBMS design. It helps to design, develop, implement, and maintain the database management system. A DBMS architecture allows dividing the database system into individual components that can be independently modified, changed, replaced, and altered.
- The Database Management System (DBMS) architecture shows how data in the database is viewed by the users. It is not concerned about how the data are handled and processed by the DBMS. It helps in implementation, design, and maintenance of a database to store and organize information for companies.

Entities in the Database include:

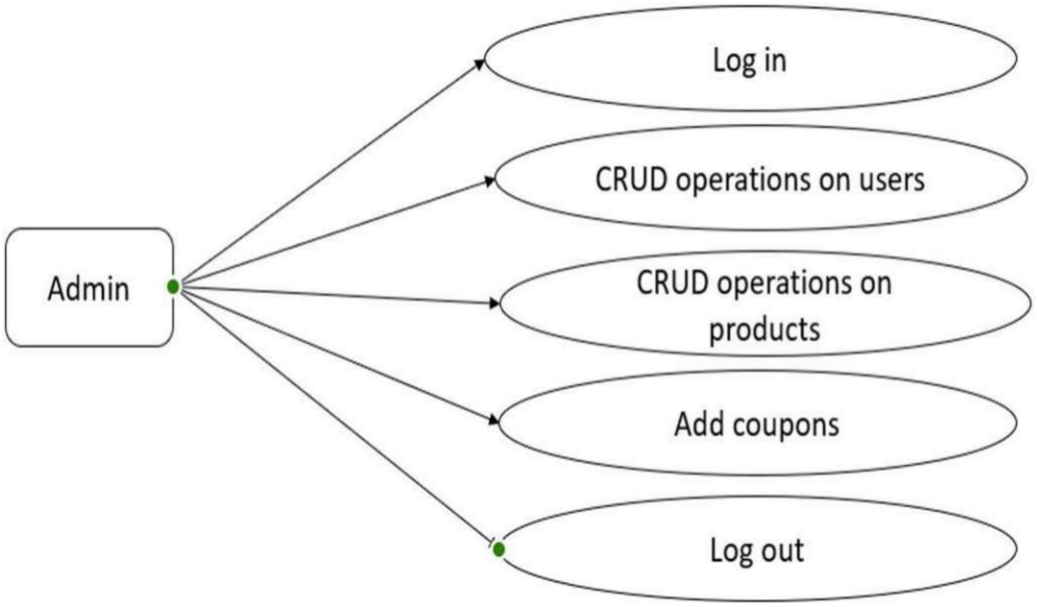
- Admin
- User
- Question
- Answers
- Image model

TOTAL PROJECT OVERVIEW



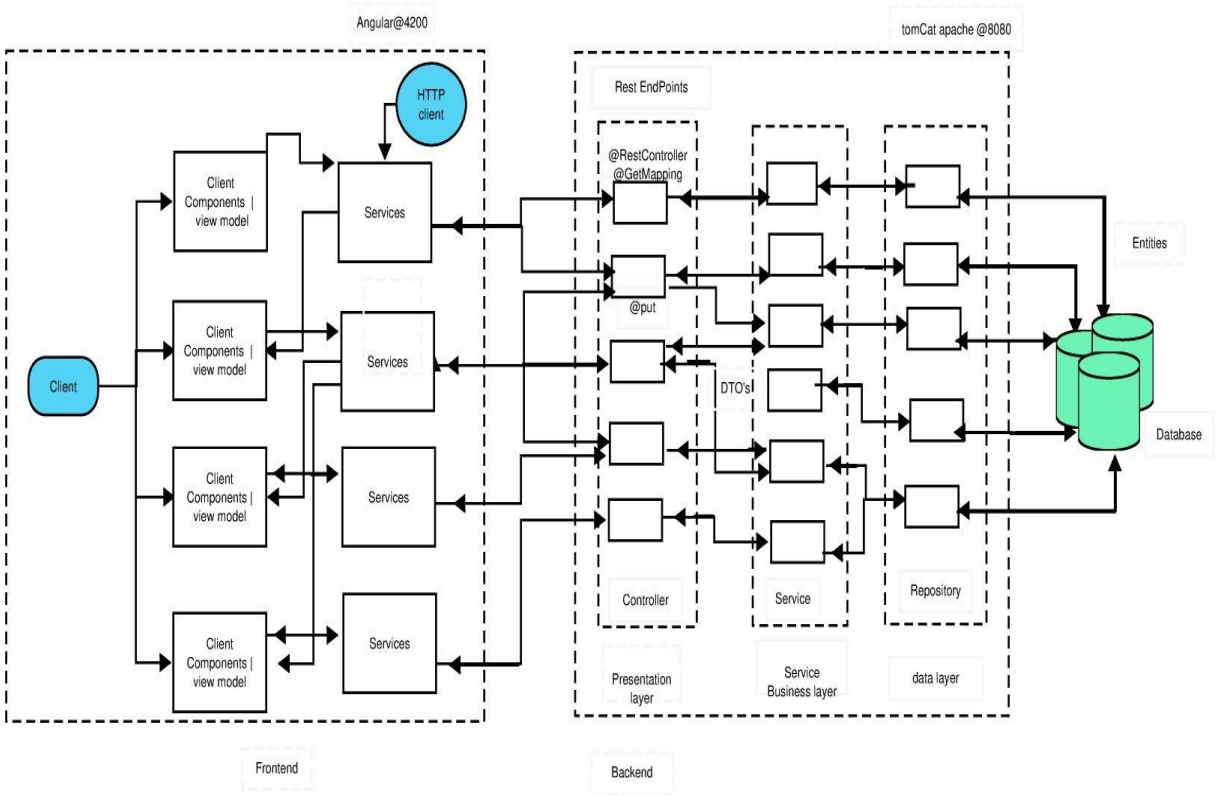
Full-stack-web-development entails the creation of both the frontend (or front end) and backend (or backend) of a website or application. Front end and back-end developers are usually necessary for every web development project; however, a full stack developer performs both.

FUNCTIONALITY OF ADMIN



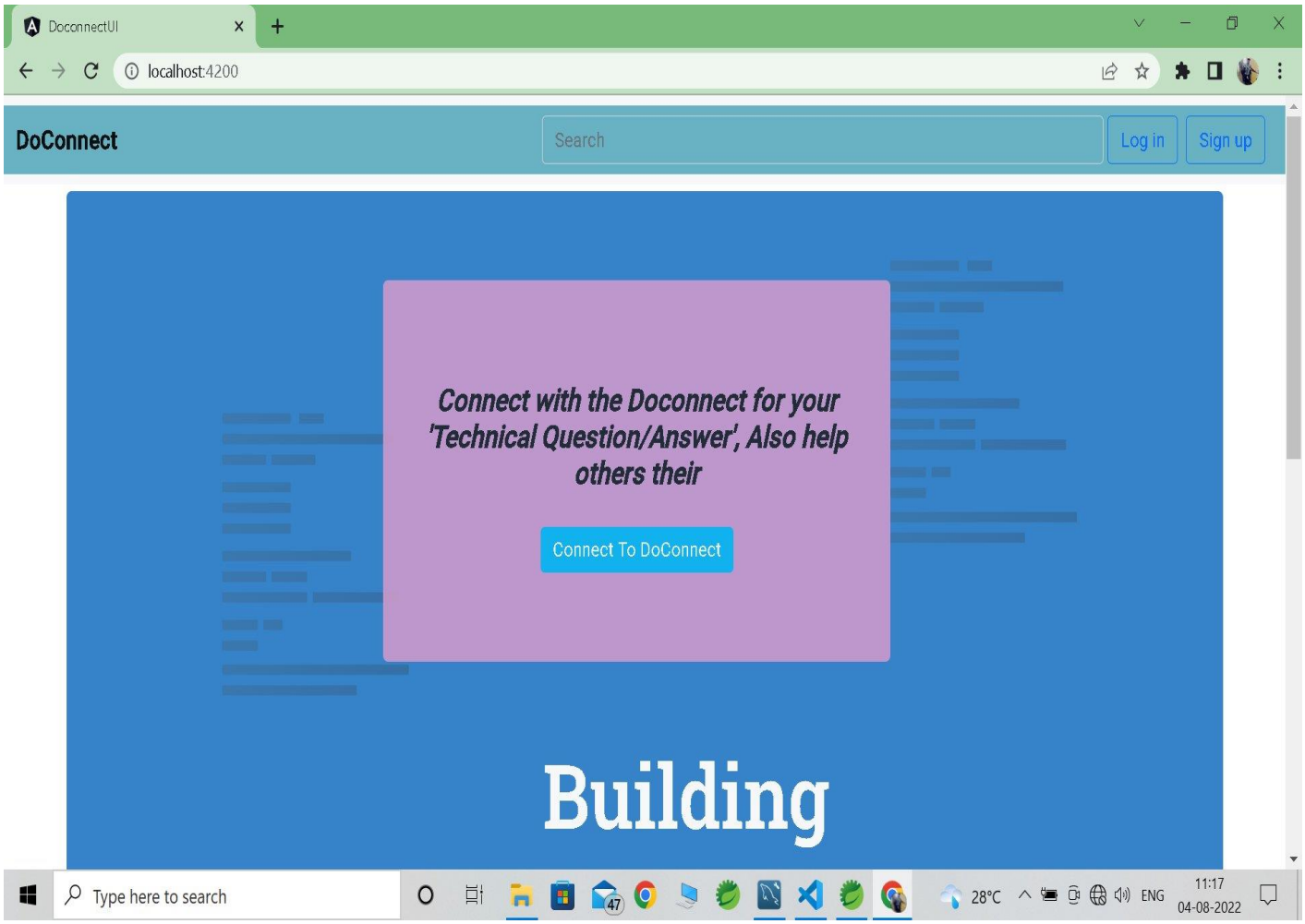
- Admin can be able to login, logout and register into the application.
- Admin can perform the CRUD operations on Users and Products and can also upload the bulk Questions.

PROJECT FLOW

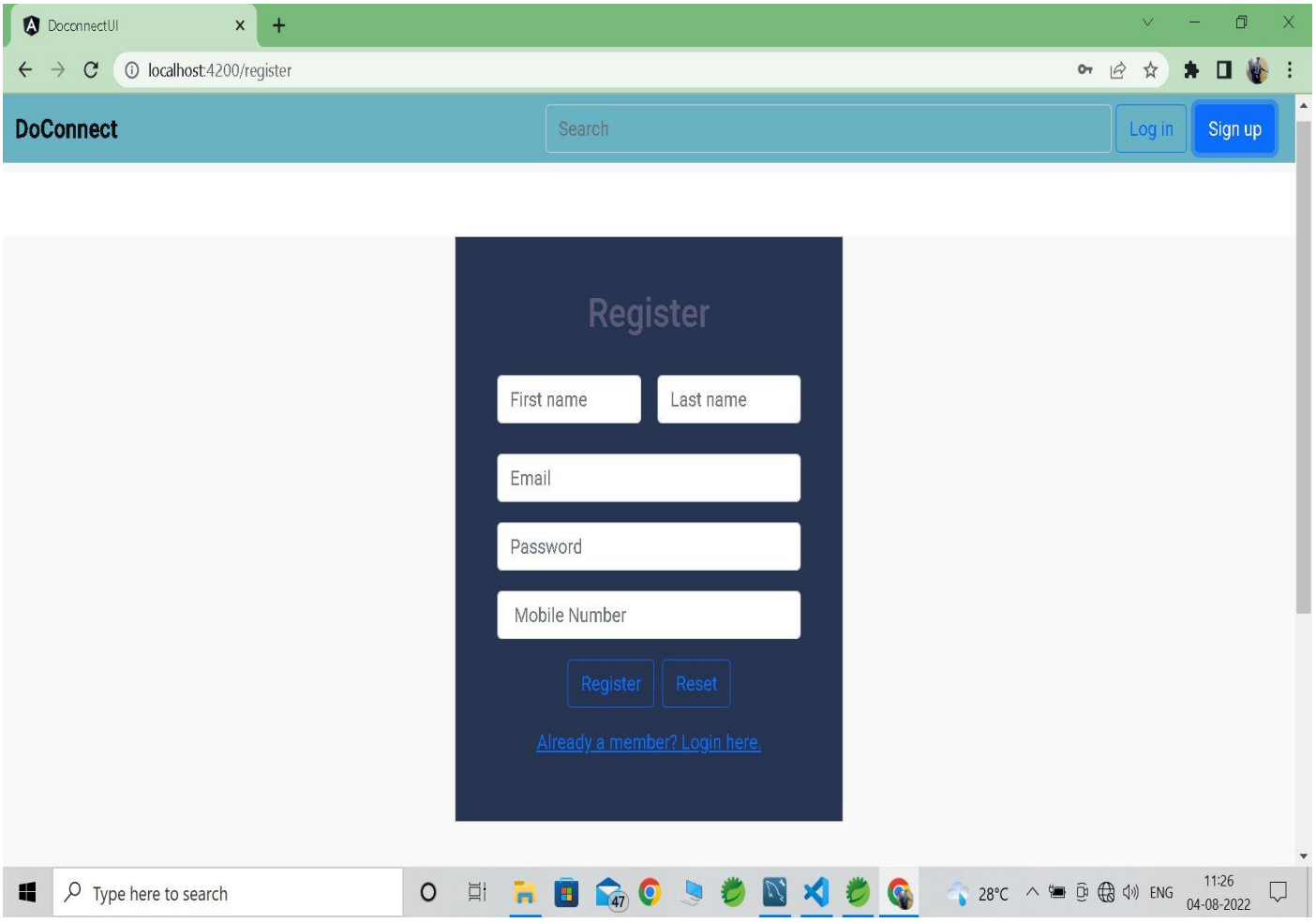


RESULTS

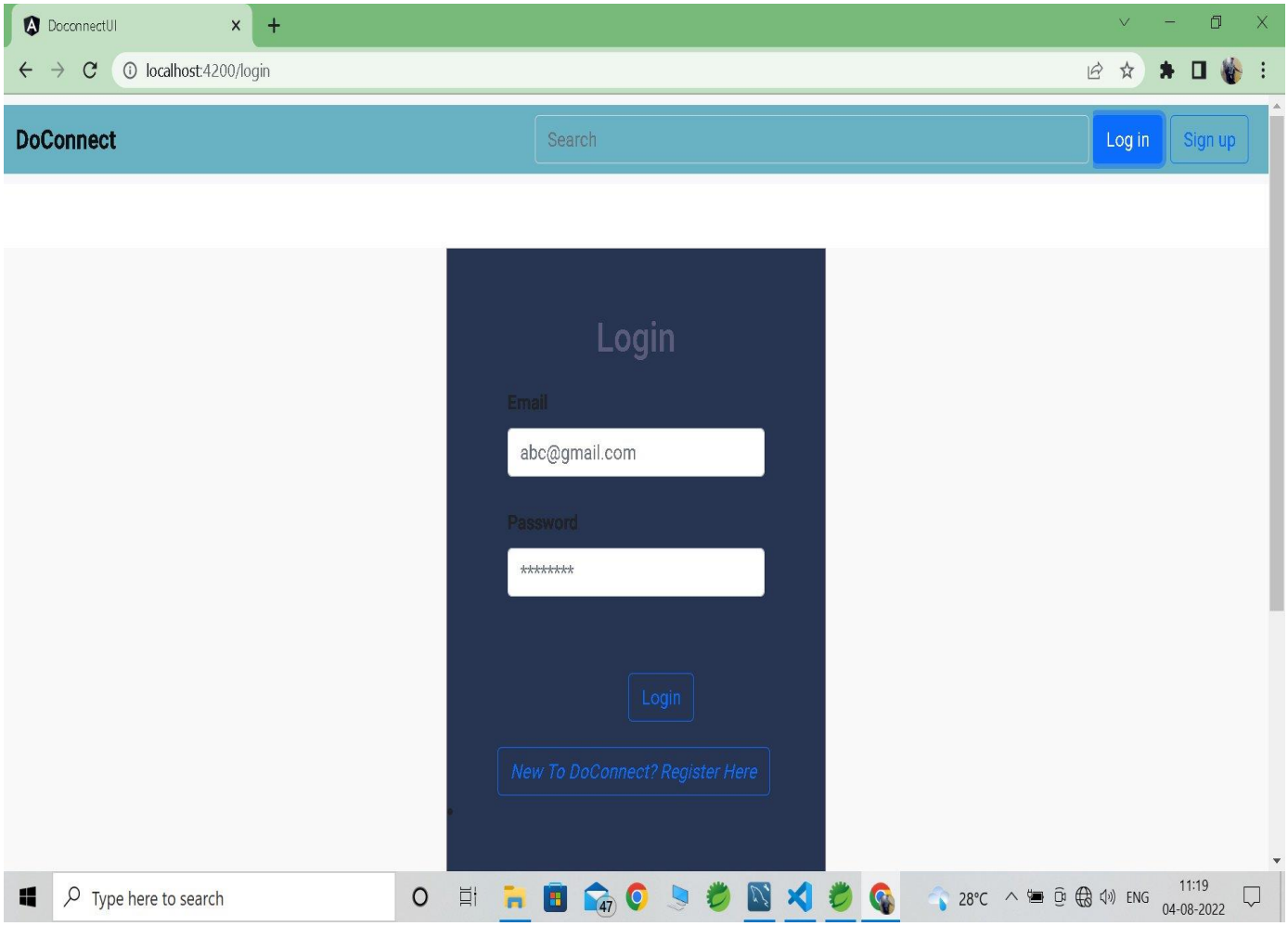
- Below image represents the **Homepage** of DoConnect where all thecomments can be viewed and can be answered.



- This represents the REGISTRATION form to register in the DoConnect.

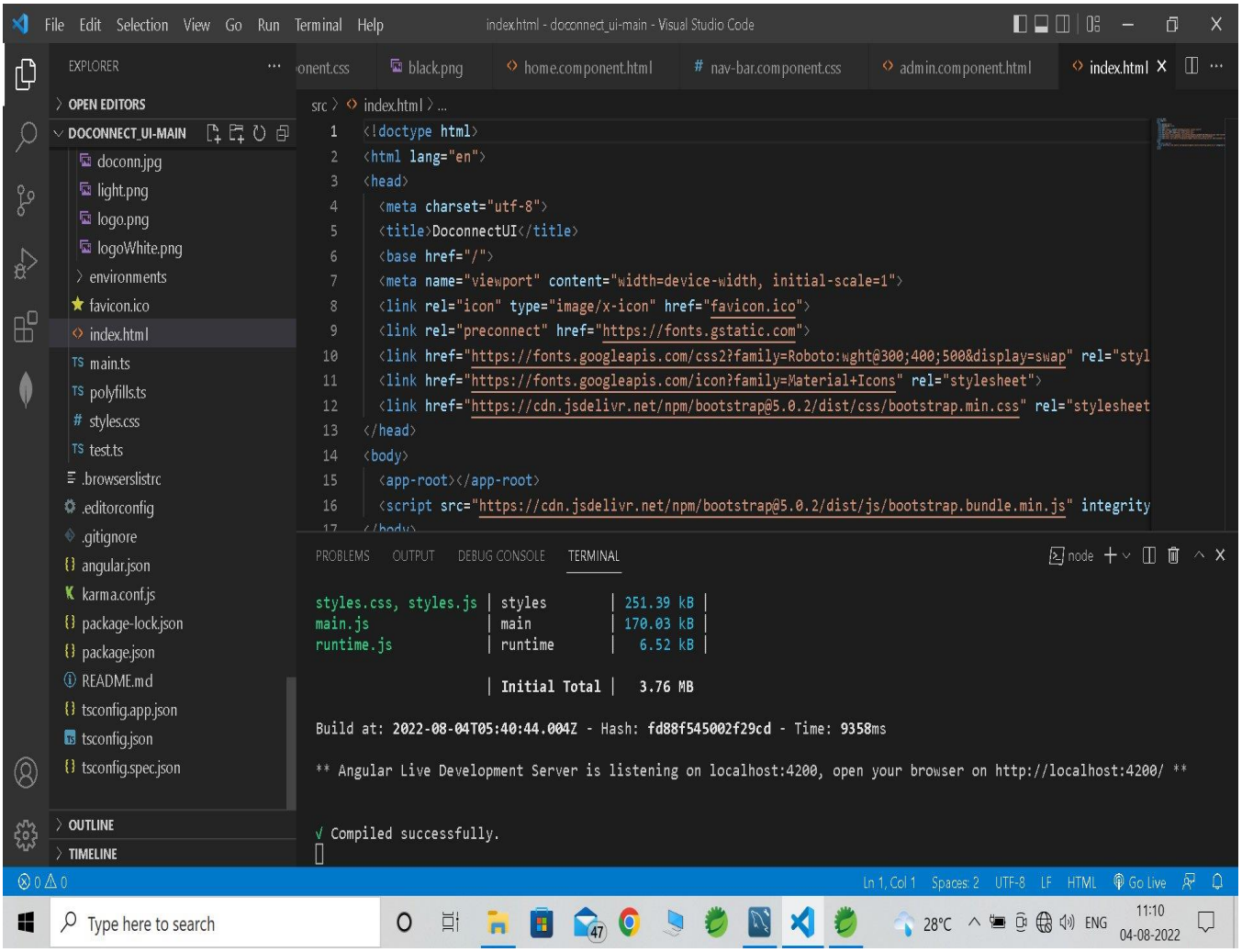


- Below represents the LOGIN form to log in to the registered account.

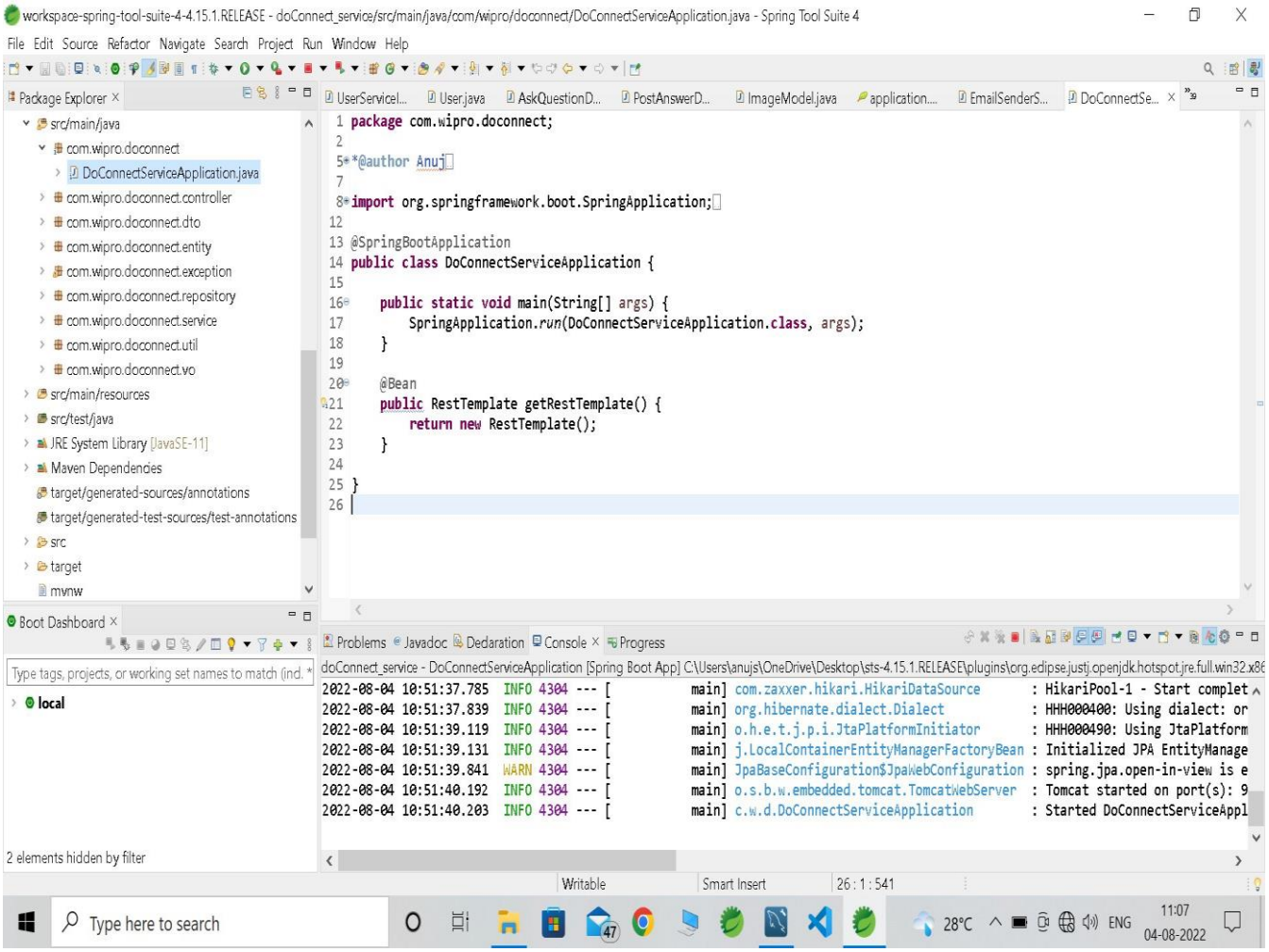


STRUCTURE OF FRONTEND AND BACKEND

STRUCTURE OF FRONTEND:



STRUCTURE OF BACKEND:



THANK YOU…