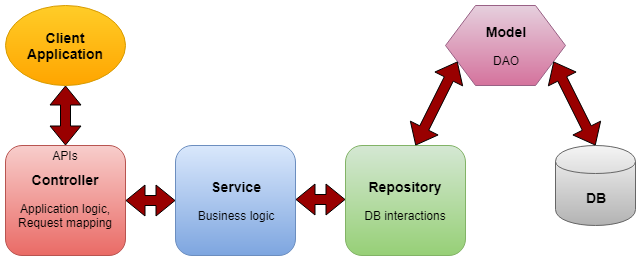
**StackOverflow Assigment**

**Overview**

This documentation provides an overview of the architecture for the StackOverflow Assigment. The system follows a Layered Architecture pattern, promoting separation of concerns by organizing the application into four main layers: Controller, Service, Repository, and Model. Below we describe the responsibilities and interactions of these layers within the context of our application.

**Architecture Diagram**



**Layers Description**

**Client Application**

The Client Application represents the user interface or external system that interacts with our StackOverflow clone. It sends requests to and receives responses from the APIs exposed by the Controller layer.

**Controller Layer**

The Controller Layer is responsible for handling incoming HTTP requests, interpreting user inputs, and returning the appropriate responses. It serves as the entry point for all REST-ful API calls and delegates the processing of these requests to the Service Layer.

Key Responsibilities:

* Mapping API endpoints to specific controller methods.
* Validating and binding request data to command objects.
* Invoking services to perform business operations.
* Handling and mapping exceptions to HTTP responses.

**Service Layer**

The Service Layer contains the application's business logic. It orchestrates the application's response to user input, communicates with the Repository layer to retrieve and persist data, and prepares the model that will be returned to the Controller layer.

Key Responsibilities:

* Implementing the core functionality and business rules of the application.
* Orchestrating data flow between the Controller and Repository layers.
* Handling transactions and ensuring data consistency and integrity.
* Abstracting the details of the persistence layer from the Controller layer.

**Repository Layer**

The Repository Layer provides a bridge between the business logic and the database. It is responsible for data access and manipulation by performing operations such as querying, inserting, updating, and deleting records within the database.

Key Responsibilities:

* Abstracting the data access implementation.
* Providing methods for searching and retrieving entities from the database.
* Implementing data access mechanisms to work with the underlying database.

**Model (DAO)**

The Model Layer, also known as Data Access Object (DAO), represents the domain model of the application. It includes entities that map to the database tables and is used by the Repository layer to interact with the database. In this project the Model Layer is implemented through the Spring API and as such its implementation is masked under the API calls.

Key Responsibilities:

* Defining entity classes that represent the data model.
* Mapping entities to database tables using Object-Relational Mapping (ORM).
* Encapsulating the database access logic and business data.

**Database (DB)**

The Database is where the data of the StackOverflow clone is stored. It includes tables for users, questions, answers, and other relevant data entities.

**Technologies Used**

* **Spring Boot**: Simplifies the bootstrapping and development of new Spring applications.
* **Spring Data JPA**: Provides repository support and ORM capabilities for accessing relational databases.
* **Spring Security**: Offers authentication and authorization capabilities.
* **MySQL**: Acts as the relational database management system.

**Endpoints**

A series of RESTful endpoints are exposed to allow interaction with the system, such as:

* **/users/register**: To register a new user.
* **/questions/insertQuestion**: To post a new question.
* **/answers/insertAnswer**: To post an answer to a question.

*(Endpoints should be listed in detail with their HTTP methods, paths, request/response formats, and descriptions.)*