**Documentation A1**

Lacatus Arpad-Alex

Group 30433

**1.Introduction**

This assignment consists of a simpler version of the website StackOverflow. We will have 2 types of users (regular user and moderator). Some key feature include: users not being able to use the app without being logged, an user can post a question, answer a question, upvote/downvote and delete its own questions and answers. Questions have title, tags, descriptions, and pictures while the answers only descriptions and pictures. The passwords will be stored encrypted in the database.

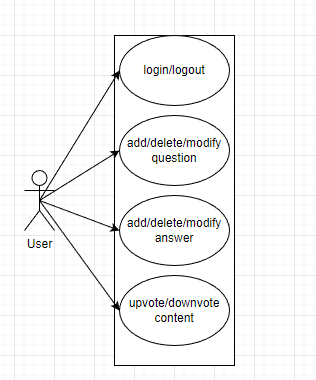
**2.Technology**

The main technologies used are Java Swing and SQL. Swing is a GUI widget toolkit for Java. It is part of Oracle's Java Foundation Classes (JFC) – an API for providing a graphical user interface (GUI) for Java programs. Structured Query Language (SQL) is a standardized programming language that is used to manage relational databases and perform various operations on the data in them. The IDE we are using is IntelliJ IDEA, an Integrated Development Environment (IDE) for JVM languages designed to maximize developer productivity.

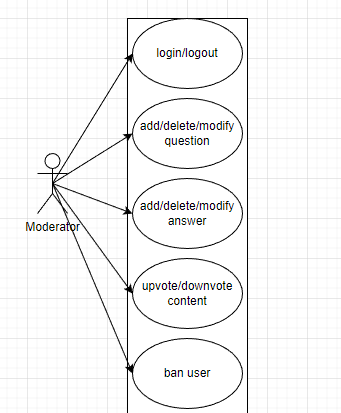
**3.Use case Diagram**

Use case diagram summarizes the details of our system’s users (or actors) and their interactions between them and with our system.

User use case:



Moderator use case:

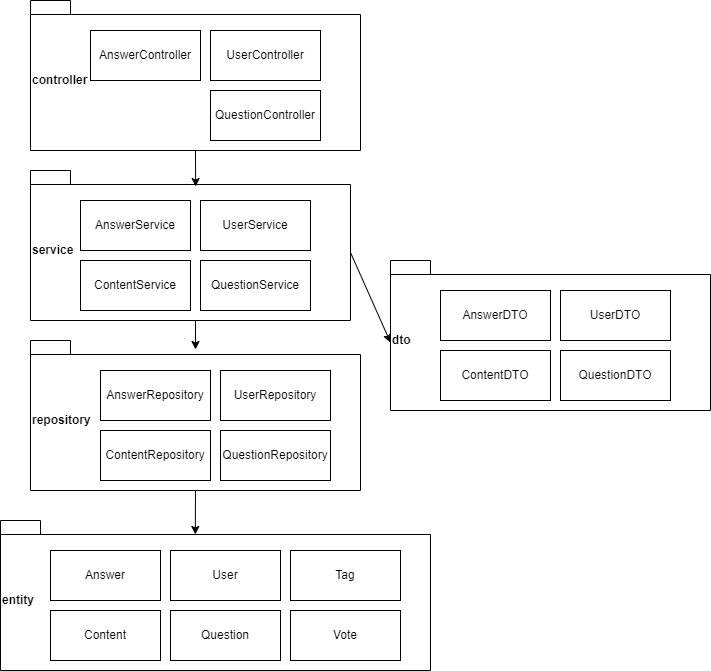


**4.Architecture**

For the architecture we will use a layered architecture which is comprised of controller, service, repository, model, DTO, GUI.

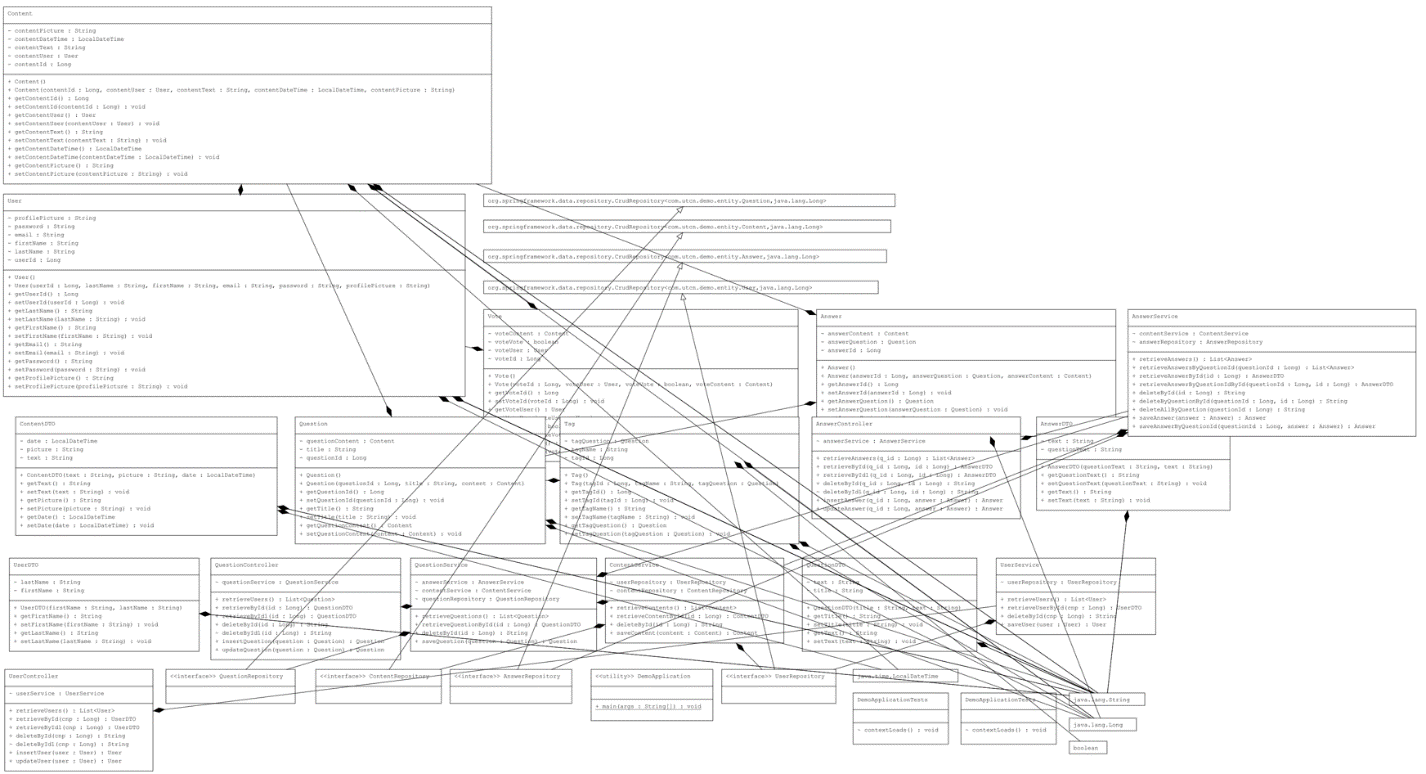
Controller is the one responsible for the request handling, it manages the requests and dispatches responses for each request. The service layer encapsulates the application's business logic, controlling transactions and coordinating responses in the implementation of its operations. The repository layer’s main goal is to isolate domain objects from details of the database access code and to minimize scattering and duplication of query code. Model layer is the one responsible for the implementation of the business logic. DTO has the responsibility of carrying data between processes. GUI layer constitutes the connection between the user and our application by providing tools to the user for interacting with the app.

**5.Package diagram**

The package diagram is a diagram that provides a way of visualizing the connections between the packages of our project.

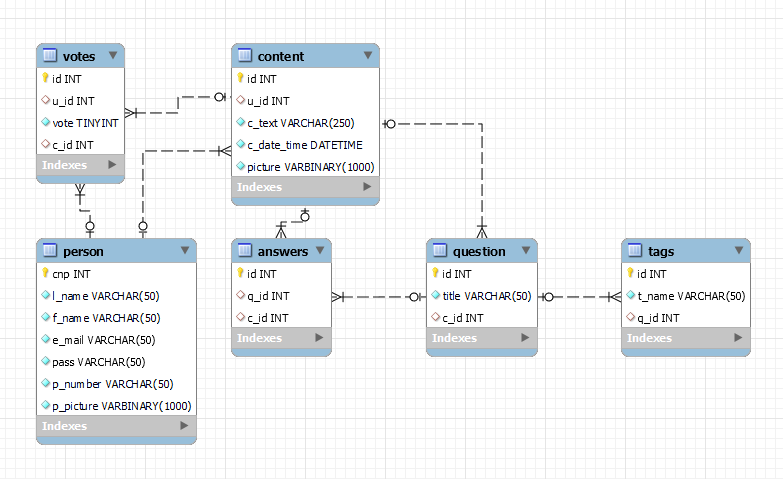
**6.Class diagram**

The class diagram is a diagram that is used to represent the connections between the classes in our application.

****

**7.Database diagram**

Here we have the diagram that describes visually the connections between the tables of our database.

****

**8.Endpoints requests**

Our application has endpoints for user, question or answer. These endpoints follow the CRUD principle so for each of them we have create, read, update, delete. The endpoints are as follows: for the create we have “insertQuestion” , for the read by id we have “getById/{id}” or just “getById”, for the delete we have “deleteById/{id}” and for the update we have “updateQuestion”. To note that for working with different services we have slight modifications question -> answer or user. For each controller these endpoints are preceded by a specification. For the questions we have “/questions” for the users we have “/users” and for the answers we have “/questions/getById/{cnp}/answers”. We used a different approach for the answers in order to bind them to a specific question. For the insert operation we have post mapping and we request a body that requires the object we work with(i.e. question/user/answer), for the read we get our id from a path variable, the delete works the same and as for update works similarly as the create and requires the same body.