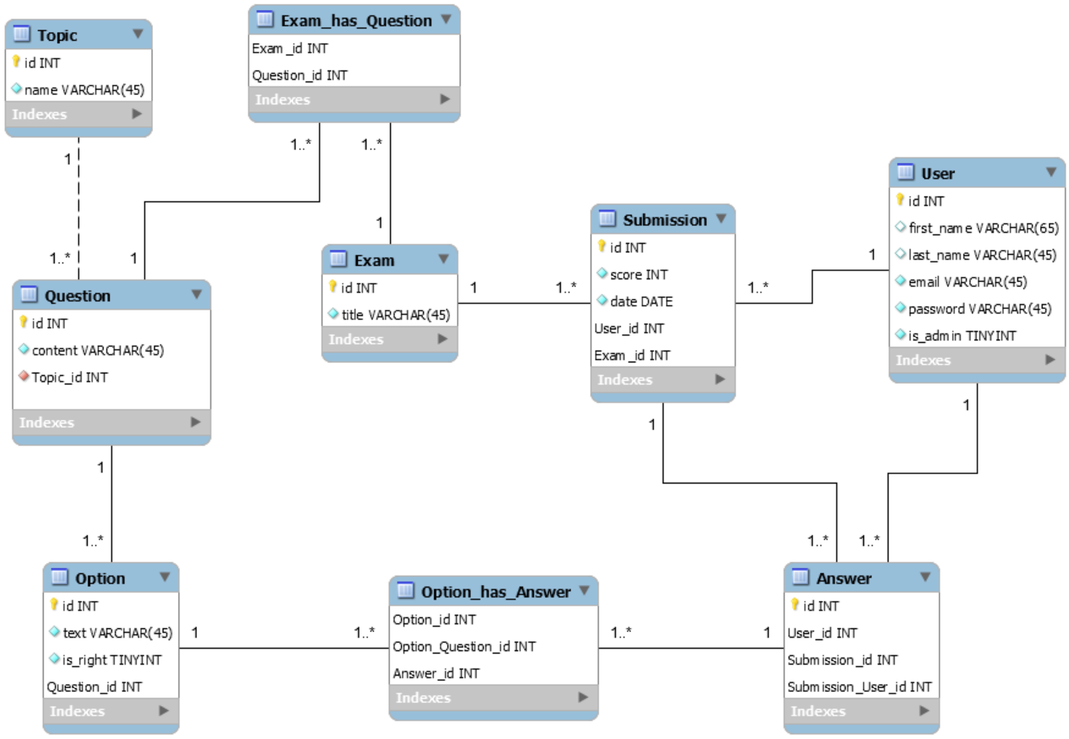
Gabriel Garces Chavez

Technical Specification

Database Schema

The database was designed according to the specifications of the project to make a quiz manager for students and faculty. It allows a student to take exams as many times as they want and receive their results instantaneously after they finish. Each table has their own id to identify themselves. The schema was implemented with Hibernate while respecting the JPA specifications. As such each class contains their own mapping which allows us to manipulate the objects as well as their tables.



Maven

The project was built with Maven in order to facilitate the project structure as well as provide easier maintainability and updates for our dependencies.

Injection Service

In order to provide the injection service for our various project dependencies in the classes I used the Spring framework due to its position in the market. Spring allows us to either use their own annotations or the JPA annotations for our classes. In order to maintain maintainability and easier manipulation I followed the JEE guidelines.

Persistence Service

Hibernate was selected as the JPA implementation service for the project as previously mentioned. It gives us ORM annotations which facilitates saving to the database via manipulation of objects. In combination with Spring and Maven these technologies make developing an application faster. The database chosen is H2. It is used in its embedded form which means that the user must create data before being able to log in.

Angular

Angular was selected for the front end since the application structure resembles that of a Java project. It provides various technologies that facilitate development of a single page front end as well and is one of the leading frameworks in the market. The standard port for angular was used.

Classes

Quiz-Core

Datamodel

All data model classes contain their respective getters and setters as well as some add and remove methods to maintain relationships to other classes in accordance to the ORM.

Answer

Stores the relationships to the options the user selected as well as to the user himself, the submission and to the question to which the answers belong.

Exam

Contains attribute title which is unique. It is identified by an id generated by the database. It also contains add and remove methods for the relationships to answer and questions.

Option

Contains attributes text to store the option value and is_right Boolean to specify its correctness. It maintains bidirectional relationship with answer and question. Get methods for question and answers are omitted to prevent the server to include them in the json response.

Question

Refers to a question that may belong to an exam. It manages the relationship with options via hibernate annotations. It is identified by a unique Id in the database and may contain as many options as the user desires. Its value in a quiz is calculated by the submission service.

Submission

Aggregates the user results for a quiz/exam. It stores the date on which it was taken as well as the score. Subsequent modifications to the score are not reflected if the question answers are changed. Future implementation will take care of this. It stores the user who did the exam, the exam data, as well as maintain the relationship with answers.

Topic

Refers to a theme for questions and subsequent exam. It is identified by a unique id generated by autoincrement by hibernate in the database. It can be used to identify questions, search questions and create tests relating to a theme.

User

Contains user information for the database. Its most important attributes are email and password which are used as credentials to gain access to the application. It also has a Boolean attribute called is_admin which is used to determine the privileges and actions the user can have. It is identified by its id in the database.

Services

DAO<T>

Abstract class that contains the methods to save, delete, update, and search for entities in the database. It is implemented by each class DAO which contains the implementations for the search query and parameters.

AnswerDAO

Implements DAO for answer entities in the database.

ExamDAO

Implements DAO for exam entities in the database. Allows search by title of the exam.

OptionDAO

Implements DAO for option entities in the database. Search method is used to get the options belonging to a question.

QuestionDAO

Implements DAO for question entities in the database. Allows search by content and contains a method to obtain 10 pseudorandom questions for creation of an exam.

SubmissionDAO

Implements DAO for submission entities in the database. It allows search by user id. Primarily used to display submissions to signed in user.

TopicDAO

Implements DAO for topic entities. It allows search by name of the topic.

UserDAO

Implements DAO for user entities. It allows search by first name and last name for the searching method. It also has a method to validate user credentials.

ExamService

Contains methods for more high level and business process actions. It contains methods to create an Exam with questions, a method to delete an exam with its corresponding relations to question and a method to update the many-to-many relationship with questions.

QuestionService

Contains methods that allow creation of a question with options, creation of question with a topic, removal of options from a question (and their subsequent deletion), deletion of a question along with its options and update method for the options and content of a question.

SubmissionService

Contains method for scoring a user submission after they have answered an exam, as well as the method that allows a submission to be created along with the answers of the user. It also contains a method that facilitates adding options to an Answer for later update in the database.

Quiz-Rest-Api

CorsFilter

This class allows requests coming from another port to be processed by the server. The implementation allows for all requests from all sites to be processed. This is not best practice and should be changed to a finer implementation that allows only recognized sources to be processed.

ExamResource, OptionResource, QuestionResource, SubmissionResource, TopicResource, UserResource

Each of these classes contains the methods that respond to http requests to the API using DELETE, PUT, POST, and GET for their respective resources in the database. All of them contain methods for creation, deletion, search, and update. Each with their corresponding http request type. User resource contains method to validate credentials for a user.

Wrappers

ExamQuestionWrapper, QuestionOptionWrapper, SubmissionCreationWrapper

The classes included in this type of objects are used to receive various class types in the ResourceBody parameter. They are used in QuestionResource to obtain question with options, in Exam Resource to get exam with questions, and in Submission with exam, user, and answers.