JAVA PROJECT

(QUIZ MANAGER)

By

Thejus Thejus (MSc - SE)

Saurabhkumar Gajjar (MSc - SE)

EPITA

ÉCOLE D'INGÉNIEURS EN INFORMATIQUE

CONTENT

1. Subject description
2. Subject analysis
    a. Major features
    b. Application Feasibility
    c. Data description
    d. Expected results
    e. Algorithms study
    f. Scope of the application (Limits and Evolution)
3. Conception
    a. Chosen Algorithm
    b. Data Structures
    c. Global application flow
    d. Global schema and major features schema
4. Configuration instructions
5. Commented Screenshots

1. Subject description

Built using Servlets web, Spring Framework and Hibernate

The project focusses on managing quiz and is built surrounding the core operation to handle all database to manipulate the list of Questions. Javadoc will be built into the project to increase the understandability and reusability.

We have used the model, view and controller (MVC) model to structure the application. Possible operations to manipulate the are we can create, read, update and delete questions.

Users - The users are divided into two, The privileged users (Professor) and the non-privileged users (Student). The details of the both the types of users are maintained in different database tables.

Flow - The privileged user needs to login to the application to be able to access, modify, delete or create new Quiz. Hence the application is designed such that all operations are restricted only to the privileged users.
Then the user if he is a quiz manager lands to the dashboard where he can search for already created questions, view them, edit them and delete them or he can create a new quiz. After the quiz is created then the mcq options can be created as well. Then he will be redirected back to the dashboard.

In UI, we are using technologies like HTML5, CSS3, BOOTSTRAP3, jQuery and Java Script. We are connecting out front-end web with back-end web with use of Servlet web (JSP). Our UI is able interactive with all browsers.

2. Major Analysis

a. Major Features: -
The main advantage of this project is that user can perform CRUD

operations irrespective of type. The Reusable codes which creates general queries and whereclausebuilders help in generalizing the code. This makes CRUD operations independent of type.

b. Application Feasibility: -

This web application can authentication login user and then permit access to modify certain records of Question in quiz in the database with the help of Spring Framework, Hibernate and Servlet JSP.

c. Data Description: -

The data being manipulated on three objects.

1) Login – Login object contains a string userName, string password and string confirmPassword.

2) Mcq – mcq object contains a boolean validateMCQ, types of string mcqChoice, Autogenerated Id and question_Id as a foreign Id.

3) Question Object – question object has one to many relation woth mcq object.

d. Expected Results: -

Easily manipulate the list of Questions in the database without any problems. Login to the portal and check the state of the objects persisted in the database.

e. Algorithm study: -

We have not used any special algorithm in the project.

f. Scope of the application (Limits and Evolution): -

Limits – The Quiz Manager only can-do operation on quiz but he/she can not do operation with marks per question.

Evolution – To get better security and result we used Sonar product which helps to polish our code.

3. Conception

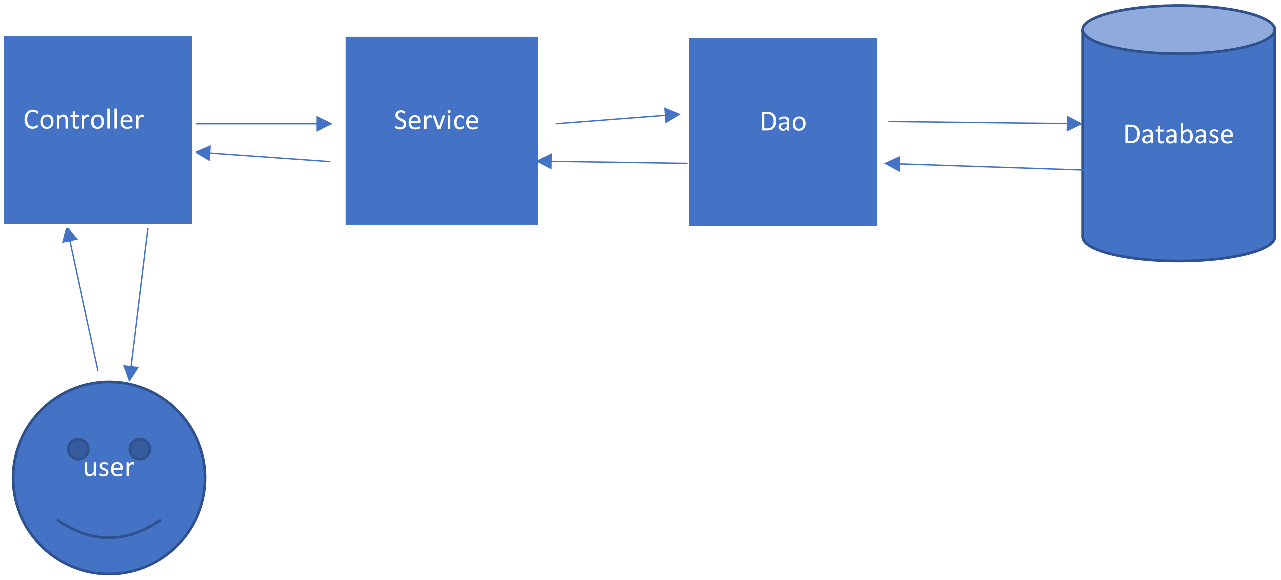
a. Chosen Algorithm: -

We have Login Module which logs in the users and once the login successful we have prepared a documented Question Services class with all the operation related to the Question and Quiz. #how dao fetch value?

b. Data Structures: -

A The data structures used are List's when the search operation is hit by the controller. And a list of MCQ Choices and Questions will be returned by the dao method to the controller via the service class. We also use Hash Maps to map questions to mcq choices.

c. Global Application Flow: -



d. Global Schema: -

The application has three tables, The LOGIN ,the QUESTIONS table and the MCQChoice Table. The login table contains the list of users with priveliged users and the MCQChoice table contains mcq choices linked to questions via Foreign key id of question and the QUESTION contains the list of all the questions.

4. Configuration instructions: -

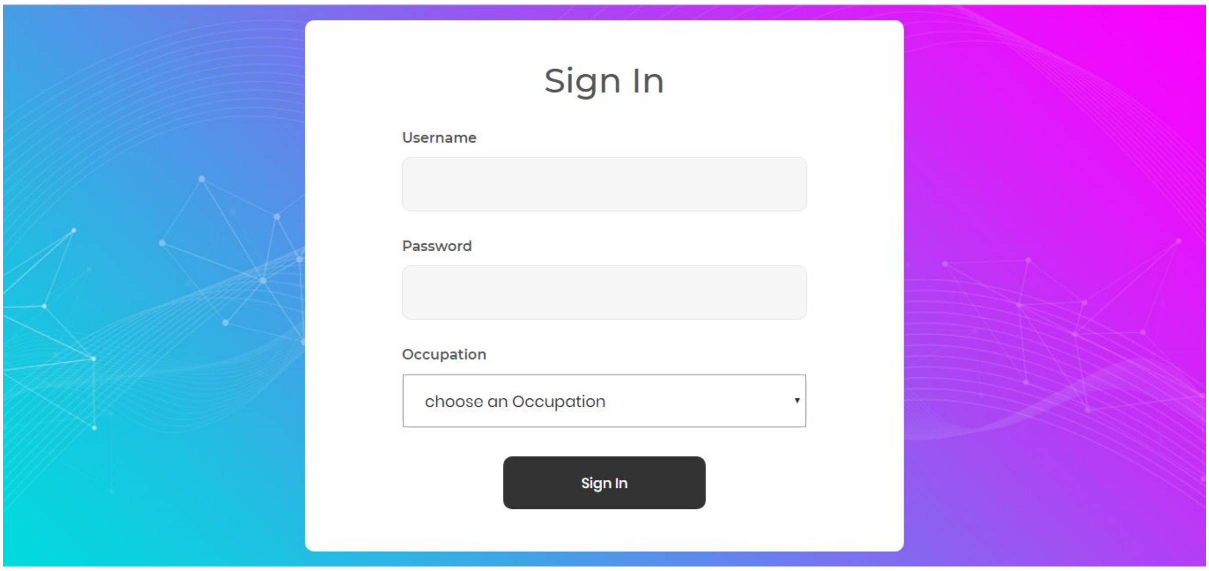
1. Quiz Manager : This is a dependency for the quizmanagerweb application.
2. Quiz Manager Web : We have used servlets and JSPs.
3. You will need Tomcat server for your eclipse.
4. We have used DERBY database. You would need to modify the properties of the database in the applicationcontext file under the header datasource.
5. There is also a need to add quizmanager as a librarby dependency to this project.
6. You would need to update maven dependencies for both the project.

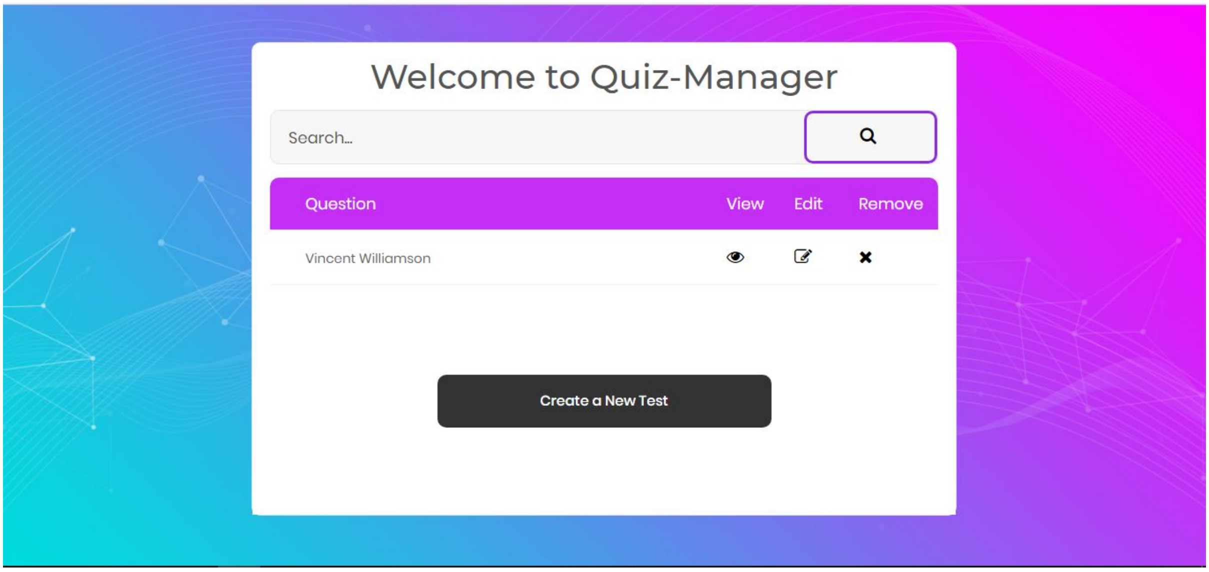
REFRENCE: BINOY CHERIAN

https://github.com/BinoyCherian/Reusables

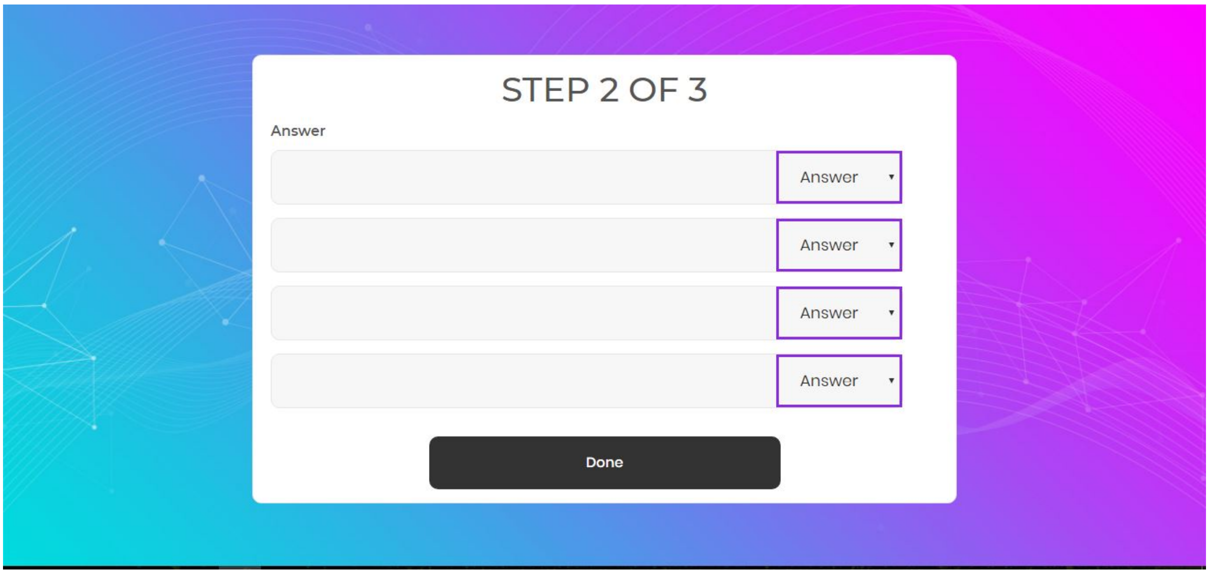
5. Screen Shorts: -

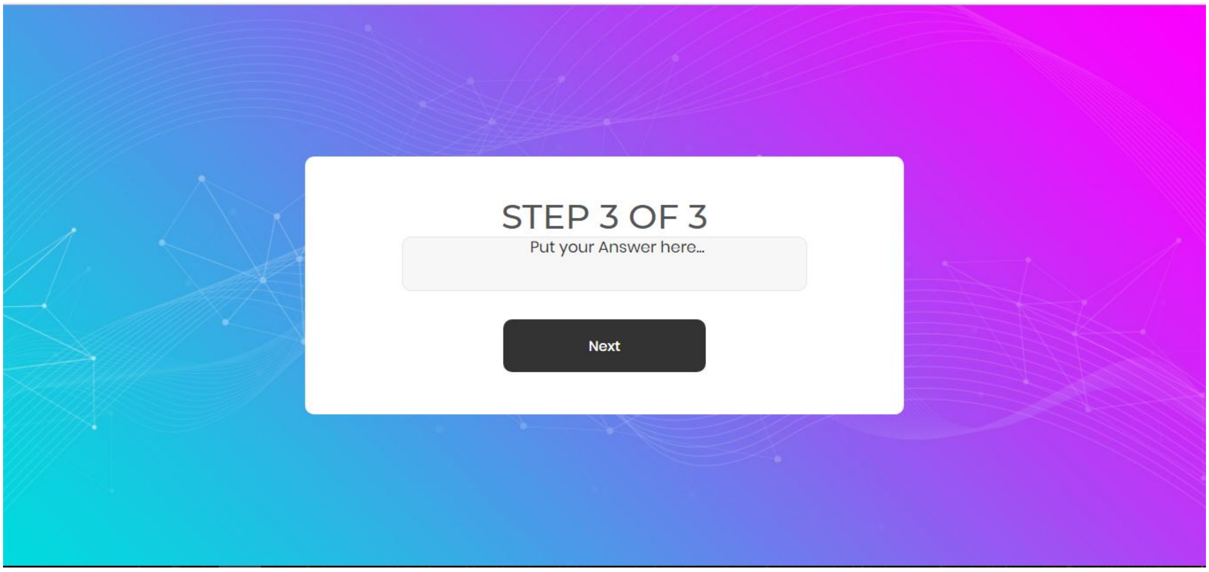
UI screen shorts are here, you can see whole application when you Run a code.

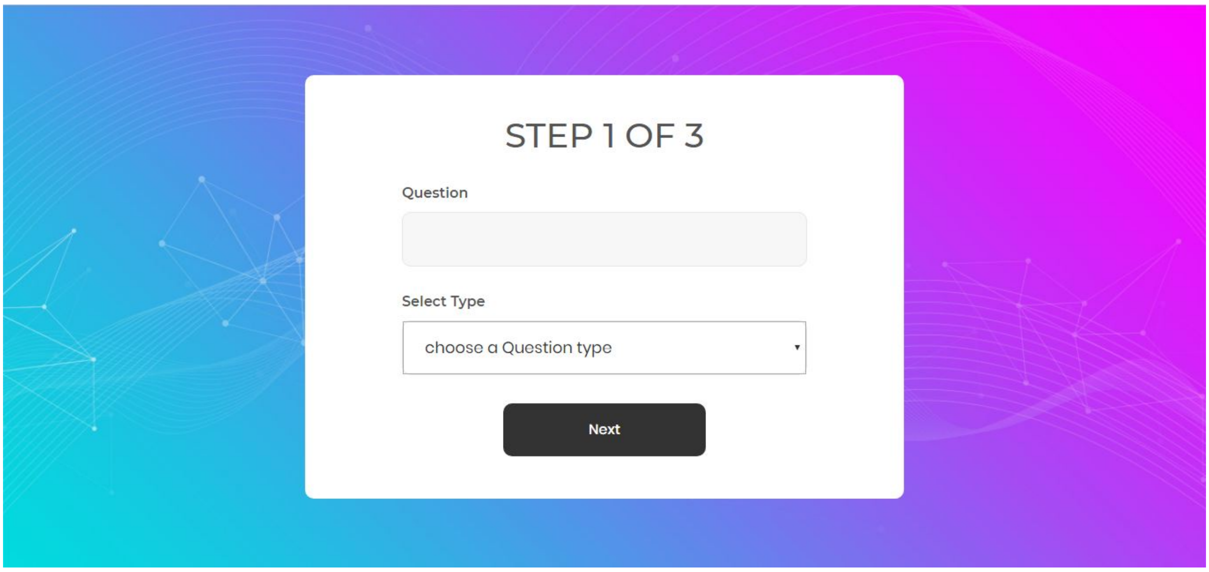


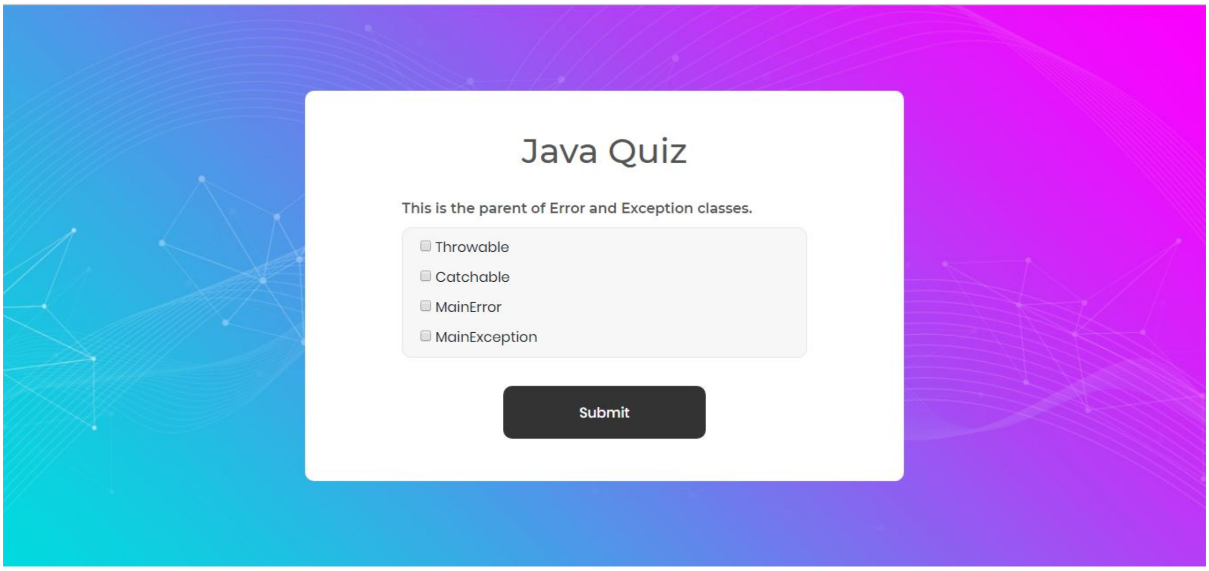


Question	View	Edit	Remove
Vincent Williamson	👁	✎	✖


STEP 2 OF 3
Answer
Answer
Answer
Answer
Answer
Done


STEP 3 OF 3
Put your Answer here...
Next

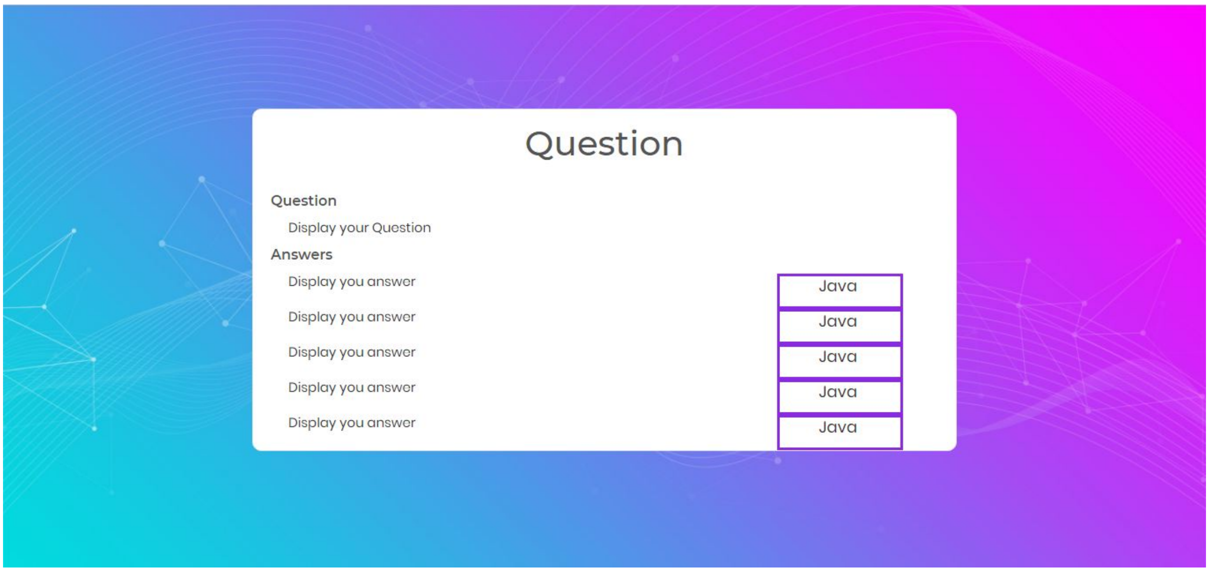

STEP 1 OF 3
Question
Select Type
choose a Question type
Next



Java Quiz

This is the parent of Error and Exception classes.

- □ Throwable
- □ Catchable
- □ MainError
- □ MainException



Question

Question

Display your Question

Answers

Display you answer

Display you answer

Display you answer

Display you answer

Display you answer

THNAK YOU!