
# ADVANCE JAVA PROJECT TECHNICAL DOCUMENT

-Venkata Lokesh ANNE

## Aim:

This project's goal is to develop a program (API-oriented, Web-based) that helps with quiz evaluations.

## Technologies Used:

- Backend→ Java 8, Spring-Hibernate, Apache Tomcat 9, JPA
- Frontend→Angular8
- Database→H2

## Application Program Interface(API):

## Base URL→ http://localhost:8080/quizzery--rest-api/rest/questions

## Add Questions:

```
@POST
@Path("/addQuestion")
@Consumes(MediaType.APPLICATION_JSON)
@Produces(MediaType.APPLICATION_JSON)
public Response createQuestion(@RequestBody Questions question) throws URISyntaxException {
    //create a question
    questionDAO.create(question);
    return Response.ok(question).build();
}
```

API for generating add question→ http://localhost:8080/quizzery--rest-api/rest/questions/addQuestion

## Add Choices:

```
@GET
@Path("/getChoices")
@Produces(MediaType.APPLICATION_JSON)
public Response searchQuestions(@QueryParam("questionRef") Integer questionRef) {
    MCQChoice mcq = new MCQChoice();
    mcq.setQuestionRef(questionRef);
    //create a question
    System.out.println("questionRef::"+questionRef);
    List<MCQChoice> searchList = mcqDAO.search(mcq);

    return Response.ok(searchList).build();
}
```

```
@POST
@Path("/mcq")
@Produces(MediaType.APPLICATION_JSON)
public Response addMCQOptions(@RequestBody MCQChoice mcqChoices) throws URISyntaxException {
    System.out.println("mcqChoices::"+mcqChoices);
    mcqDAO.create(mcqChoices);
    return Response.ok(mcqChoices).build();
}
```

API for generating Choice→ http://localhost:8080/quizzery--rest-api/rest/questions/getChoices?questionRef=

API for generating Mcq→ http://localhost:8080/quizzery--rest-api/rest/questions/mcq

**Search Question:**

```
@GET
@Path("/search")
@Produces(MediaType.APPLICATION_JSON)
public Response searchQuestions(@QueryParam("qContent") String questionContent) {
    Questions questions = new Questions();
    questions.setQuestionContent(questionContent);
    //create a question
    List<Questions> searchList = questionDAO.search(questions);
    System.out.println("searchList::"+searchList);
    return Response.ok(searchList).build();
}
```

API for generating Search → http://localhost:8080/quizzery--rest-api/rest/questions/search?qcontent=

**List Questions:**

```
@GET
@Path("/questionList")
@Produces(MediaType.APPLICATION_JSON)
public Response questionList() {
    Questions questions = new Questions();
    List<Questions> questionList= questionDAO.getQuestionList();
    return Response.ok(questionList).build();

}
```

API for generating List of Questions→ http://localhost:8080/quizzery--rest-api/rest/questions/questionList

**Delete Question:**

```
@DELETE
  @Path("delete/{id}")
  public Response deleteOrderById(@PathParam("id") int id) {
    System.out.println(id);
    Questions question = new Questions();
    question.setId(id);
    questionDAO.delete(question);
    return Response.ok().build();
  }
```

API for Deleting Questions→ http://localhost:8080/quizzery--rest-api/rest/questions/delete/1

Refer User Manual For the Execution of The Project "QUIZZERY"

**Bibliography:**

https://thomas-broussard.fr/work/java/courses/project/advanced.xhtml