Quiz后端-6、实现题目页的添加、删除与查询

日期：**2025.07.24** 作者：**tfzhang**

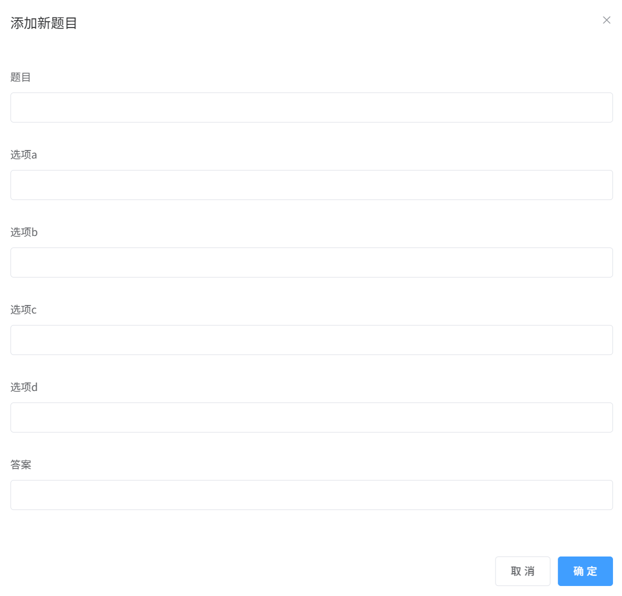
后端主要提供用户与题目的注册、查询、删除等操作。使用到的开发工具：

- IDEA2024; 关注公众号”青椒工具”，发送”IDEA”，获取windows下的IDEA安装包
- mysql 5.7；关注公众号”青椒工具”，发送”mysql”，获取windows下的mysql5.7安装包；

前情回顾：

- 用户的插入，删除，查询；

添加题目的前端页面：



1、题目的数据库表设计；

```sql
DROP TABLE IF EXISTS questions;

CREATE TABLE questions (
    id INT PRIMARY KEY AUTO_INCREMENT,
    questionText VARCHAR(255) NOT NULL,
    answer1Text VARCHAR(255) NOT NULL,
    answer1Correct BOOLEAN NOT NULL DEFAULT FALSE,
    answer2Text VARCHAR(255) NOT NULL,
    answer2Correct BOOLEAN NOT NULL DEFAULT FALSE,
    answer3Text VARCHAR(255) NOT NULL,
    answer3Correct BOOLEAN NOT NULL DEFAULT FALSE,
    answer4Text VARCHAR(255) NOT NULL,
    answer4Correct BOOLEAN NOT NULL DEFAULT FALSE,
    isDelete    tinyint  null,
    createTime   datetime default CURRENT_TIMESTAMP null,
    updateTime   datetime default CURRENT_TIMESTAMP null
);
```

此处我们的题目选项数固定为4，所以可以写死answer的数量。

如果题目的选项数不固定呢？

步骤**1**：

在quiz数据库中创建上述的questions数据库表格；

步骤**2**：

在model目录中，创建实体类：

```java
public class Question {
    private Integer id;
    private String questionText;

    private String answer1Text;
    private Boolean answer1Correct;

    private String answer2Text;
    private Boolean answer2Correct;

    private String answer3Text;
    private Boolean answer3Correct;

    private String answer4Text;
    private Boolean answer4Correct;

        /* 是否删除 */
    private Integer isDelete;

    /* 创建时间 */
    private Date createTime;

    /*更新时间 */
    private Date updateTime;
    // Getters and Setters
}

```

2、题目的添加；

步骤**1**：创建**QuestionMapper**类

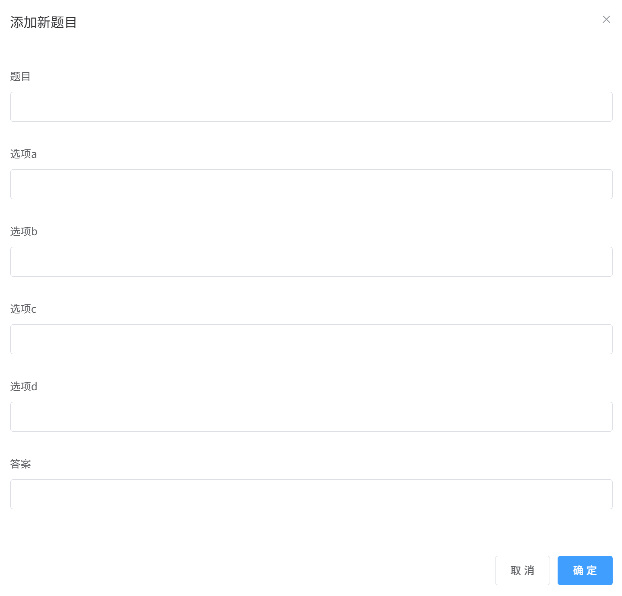
```java
@Mapper
public interface QuestionMapper {
    @Insert("INSERT INTO questions (question_text,answer1_text,answer1_correct,answer2_text, answer2_correct, answer3_text,answer3_correct,answer4_text, answer4_correct, isDelete, createTime, updateTime)" +
            "VALUES (#{questionText},#{answer1Text}, #{answer1Correct}, #{answer2Text}, #{answer2Correct},#{answer3Text}, #{answer3Correct},#{answer4Text}, #{answer4Correct}, #{isDelete},#{createTime},#{updateTime})")
    @Options(useGeneratedKeys = true, keyProperty = "id")
    int insertQuestion(Question question);
}
```

步骤**2**：测试**QuestionMapper**类

```java
    @Autowired
    private QuestionMapper questionMapper;

    @Test
    public void testInsertQuestion(){
        Question question = new Question();
        question.setQuestionText("What is the capital of France?");

        question.setAnswer1Text("Paris");
        question.setAnswer1Correct(true);

        question.setAnswer2Text("London");
        question.setAnswer2Correct(false);

        question.setAnswer3Text("Berlin");
        question.setAnswer3Correct(false);

        question.setAnswer4Text("Madrid");
        question.setAnswer4Correct(false);

        question.setIsDelete(0);
        question.setCreateTime(new Date());
        question.setUpdateTime(new Date());

        // 执行插入操作
        int result = questionMapper.insertQuestion(question);
        System.out.println(result);
        System.out.println(question.getId());
    }
```

步骤**3**：**Controller**类使用到的**model**

假设来自前端的插入题目的页面：


添加新题目

题目

选项a

选项b

选项c

选项d

答案

根据这个前端页面，在model中书写相应的实体类：

```java
public class QSBean {
    private String question;
    private String optiona;
    private String optionb;
    private String optionc;
    private String optiond;
    private String answer; // "a", "b", "c", or "d"
}
```

步骤4：在Service层创建QuestionService，并增加接口和实现

```java
//接口代码：
public int insertQuestion(QuestionSubmitBean qsBean);

//实现类：
public int insertQuestion(QuestionSubmitBean qsBean){

    String ans = qsBean.getAnswer();
    if (!List.of("a", "b", "c", "d").contains(ans.toLowerCase())) {
        throw new IllegalArgumentException("Answer must be one of: a, b, c, or d");
    }

    Question q = new Question();
    q.setQuestionText(qsBean.getQuestion());

    q.setAnswer1Text(qsBean.getOptiona());
    q.setAnswer1Correct("a".equalsIgnoreCase(ans));

    q.setAnswer2Text(qsBean.getOptionb());
    q.setAnswer2Correct("b".equalsIgnoreCase(ans));

    q.setAnswer3Text(qsBean.getOptionc());
    q.setAnswer3Correct("c".equalsIgnoreCase(ans));

    q.setAnswer4Text(qsBean.getOptiond());
    q.setAnswer4Correct("d".equalsIgnoreCase(ans));

    q.setIsDelete(0);
    q.setCreateTime(new Date());
    q.setUpdateTime(new Date());

    int result = questionMapper.insertQuestion(q);
    return result;
}
```

步骤5：创建QuestionControler，并添加代码：

```java
    @Autowired
    private QuestionService questionService;

    @PostMapping("/addQuestion")
    public Result addQuestion(QSBean question) {
        int result = questionService.insertQuestion(question);
        if (result > 0){
            return Result.success("插入新问题成功");
        }else{
            return Result.error("插入失败");
        }
    }
```

步骤**6**：**apifox**测试

测试数据：

```
question: which of the following is not a programming language?
optiona: Java
optionb: Apple
optionc: Python
optiond: Javascript
answer: b
```

3、题目的删除；

- 根据id删除用户；
- 主要采用逻辑删除的方式；

参考删除用户的代码来实现。

4、题目的查询；

- 与用户的查询类似，提供：分页查询与按关键词查询两种方式；

参考用户查询的代码来实现。