# Technical Specification

# Advanced Java Project

### Shengyi ZHOU

This poject contains two parts: back-end and front-end.

# Back-end

Hibernate, Springframework and h2 database are used in the back-end to form a database server, which can communicate with the front-end by URL link. And the back-end can be seperated into two parts: quiz-core2 & quiz-rest.

### quiz-core2

Inside quiz-core2, we can find the basic datamodel and the DAO part.

**Datamodel:**

Every class has a different datamodel.

**DAOs:**

I use a GenricDAO to form the basic functions like Create, Read, Update, Delete, and let the other DAOs extend.

**#Business:**

I don’t use it any more.

### quiz-rest

It’s like the interface between front and back. Here, we form the DTO datamodel and the Resource which contains the basic function.

**DTOs:**

When the front-end send a json data to the back, it will probably turns into DTOs at first. Then we use the initilized DTO to form the corresponding datamodel, which can be used in the back-end.

**Resources:**

I put the basic methods like CRUD here and add the path URL, so that it can be used in the front through corresponding URL link.

**CorsFilter:**

This file is necessaire for the cross-domain request.

# Front-end

I use angular2 for the front-end. There are three parts: components, datamodels and services

### Components:

In order to achieve the jumps between the pages, I create two main components: edition-form & execution-form. And put them in the main component: app.component.

Each datamodel has its own component so that we can do the simple edition, so we put them in the edition-form.component, but the MCQAnswer component contains the excution of the quiz, so I put it in the execution-form component.

In the .ts file, I called some function from service (front-end), to achieve some complicated operations, such as “addMCQChoiceToQuestion”

### datamodels:

The datamodels are used to form the data which can be read by back-end. And we also transform the data get from back-end into the datamodel which can be read by front-end.

### Service:

I wrote the basic functions in each service, and use URL to connect it with the back-end.