COMP9900 Project Proposal

21 Sep, 2020

Srcum master/Frontend	Wanchao Huang	z5192415
Frontend	Raymond Lu	z5277884
Backend	Tao Xue	z5219280
Backend	Xiaohan Zhu	z5187021
Frontend	Ziqi Liang	z5202297

Contents

1 Background

Ziqi

1.1 Problem statement

1.2 Existing systems analysis

2 User stories

Raymond

2.1 Product backlog of user stories

2.2 Product backlog in Jira

3 Sprints

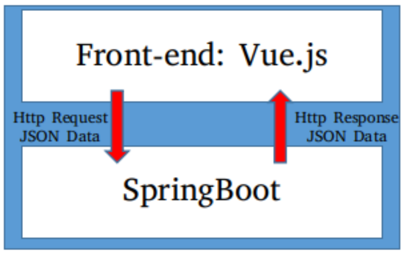
Wanchao

4 Interface and Flow Diagram

Wanchao

5 System Architecture

5.1 Separate Front-end and Back-end system

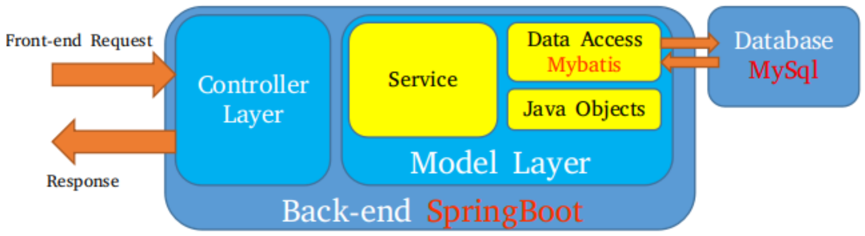


We choose to split the whole system into two subsystems: **Frontend System** which is for presentation purpose and the **Backend System** to realize the requirement. Thus the two subsystems are not dependent much on each other. So that, we can develope and test out codes independently and easily scale up and upgrade.

5.2 Front-end

We are using **Vue.js** framework for our front-end system. Comparing with traditional *HTML* and other static template, Vue framework can bring in dynamic and fancy effects. Comparing to other front-end frameworks such as Angular and React, Vue is much flatter learning curve and the higher coding speed.

5.3 Back-end



5.3.1 Basic Framework

The language we used is **Java** and basic framework chosen to realize the overall architecture is **Spring Boot** which embeds Tomcat server and gathers common java-web development frameworks. Thus, we can build up and deploy the whole system easily.

5.3.2 Back-end Architecture

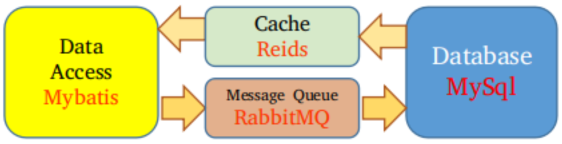
For decoupling, we adopt a MVC three-layer architecture:

- **Model Layer**: Data persistence and process requests.

  - Service Layer: Deal with specific business.
  - Data Access Object: Communicate with database.
  - Plain Java Objects

- **View Layer**: Represent the data, in our architecture is replace by the front-end system.
- **Controller Layer**: Receive HTTP request based on the url from the Front-end and return results from the Model layer.

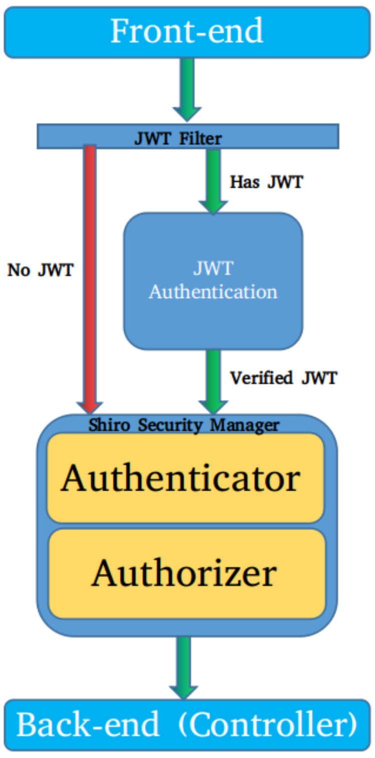
5.3.3 Data Access



In model layer, we need to fetch and update data in database. To simplify programs and improve development efficiency, we are using an ORM framework Mybatis. Besides that

Then we want to reduce the pressure of database reading and writing. Thus, we introduce **Redis** as the cache of reading data, and a message queue **(RabbitMQ)** to improve asynchronousity and concurrency.

5.3.4 Authentication and Authorization



In our system, there are three roles: Anonymous, Users and Registered Auction Bidders (RAB). Thus, we need a efficient and simple authentication and authorization framework. Thus we are using a security framework called **Shiro** along with **J**ason **W**eb **T**oken to achieve this need.

5.4 Conclusion

Our system architecture has:

- A clear description showing the presentation, business and data layers in the system, and what each layer contains
- external actors and how they interact with system
- clear description of the technologies/languages planned for use

And technologies in Use:

- Front-end
  - Vue.js
- Back-end
  - Develop Language: Java
  - Framework: Spring Boot
  - Database in USE: MySQL
  - Persistence Layer: MyBatis
  - Cache and NoSQL: Redis
  - Message Queue: RabbitMQ
  - Authentication and Authorization: Shiro + JWT
  - Cloud: AWS