**CS 5200 Final Report**

Haoyu Wang, 002814616

**Tiger Takeaway**

Tiger Takeaway is a web-based food ordering and delivery application built using Java, HTML, Javascript, CSS and MySQL.

Use frameworks including Spring Boot (Spring MVC), Mybatis, Vue and so on (details see below).

**Build, Install, Config and Run**

**Development & debugging environment**

Due to the tight schedule, this project does not have any distribution plans. The following content is for the development and debugging environment, which facilitates reproduction.

- OS: Microsoft Windows 11, x86\_64, ver 22631

- IDE: Jetbrains IntelliJ IDEA 2024.1

- JDK: Amazon Corretto 1.8.0\_402

- Build tool: Apache Maven 3.9.6

- Database: Oracle MySQL Community Server 8.0.36

- Test browsers: Google Chrome 123.0, Safari 604.1

**Installation**

1. Unzip the code.

2. Run IntelliJ IDEA, choose “Files – Open…”, and locate the “tiger-takeaway” folder. Click “OK”.

3. After importing, you can see a configuration named "TigerApplication" above the IDE. Click the green "▶" key on the

right to run it.

4. You may need to let maven download dependencies for the application. Click "Ⓜ️" button on the right and choose

"install".

**Configurations**

1. Edit the file "application.yml" in Resources folder in Project column at left side of the IDE. (src/main/resources/application.yml)

2. Set application port: edit server.port, default 8080.

3. Set email SMTP authentication: edit spring.mail.host, spring.mail.username, spring.mail.password, spring.mail.port.

4. Import SQL into MySQL server: Execute sql/db.sql and sql/user\_tiger\_dev.sql in your MySQL server.

5. Set local MySQL authentication: edit spring.datasource.druid.username, spring.datasource.druid.password.

**Run the application**

After running the application, There should be prompt in the output showing like "Project started successfully!

[Tiger Takeaway]" and two addresses for Backend management system and frontend service page.

**Used tools and packages**

- Spring Boot 2.4.5

- Mybatis-Plus 3.4.2

- Lombok 1.18.20

- Fastjson 1.2.76

- Alibaba Druid 1.1.23

- Vue 2.6.12

**Final user flow**

The final user flow of the system. The blue bold italics are the paths used by the user or employee for a certain operation (relative to the root directory). The gray background boxes are the part we originally planned to complete but had to abandon due to progress (see “Future Work” part). 图示

描述已自动生成

**Lessons Learned**

This project allowed us to accumulate a lot of experience in database application development, and also gave us a deeper insight into the behind-the-scenes work of many database-related applications that we see every day. The reason why we chose Java Spring Boot for development is because all our team members have Java development experience, even some including Java Spring Boot internship experience. But this development has given us a deeper understanding of Spring Boot, and we have also had some contact with the Javascript toolkit Vue. Mybatis-plus is a tool that the two team members have come into contact with in their personal projects. It has greatly enhanced the Java project's ability to operate the MySQL database, simplified the development process and reduced the difficulty of development.

This exercise gave us a deeper understanding of a systematic application involving a database. In our discussions, we had imagined checking input data at three levels: front-end, back-end service, and database, but during the actual development process we realized that this situation was very redundant. This is not to say that it is unrealistic, because we believe that a robust, distributed system should require multiple verification of data, but development and design teams must make trade-offs with the resulting performance overhead and experience delays. We are also aware of the importance of positions such as product manager, development manager, and design director in a company, because we often think a lot but often hit the wall during the actual development process.

During the course we came into contact with MongoDB, which we think is a great NoSQL database. We are a little disappointed because we did not meet the prerequisites for using a NoSQL database on the final project, we think that this project can be better implemented using MongoDB, and although as mentioned above, some of our team members have used Spring Boot development experience, but none of us have experience using MongoDB for development (only one team member has been exposed to development using large JSON files as databases), so we are all very interested in using MongoDB, and It is a pity that we were unable to, but we will look to future development opportunities with MongoDB.

When we proposed, we mentioned the use case of user payment confirmation and manager confirmation of shipping orders. But we ultimately did not implement these two parts for the following reasons: obviously, we did not have any experience in designing any payment scenarios, and we did not know the starting point for simulating payments. For order confirmation, it was due to schedule issues - the previous development process had taken up a long time, so this use case could only be cut.

**Future Work**

In our future work, we will first focus on solving the two unfinished use cases mentioned earlier. One idea is that we build another server and simulate a simple payment method service API on it, such as simulating the Paypal API, so that it will be easier to transplant the program to actual operation. There will be time in the future to complete the order confirmation feature. Canceling/Rejecting orders is very important; because in actual scenarios, orders may encounter various problems, such as: dishes sold out, incorrect address filling, customer cancellation, emergencies, etc.

We hope that in the future we can migrate our database to MongoDB, not only to satisfy our curiosity: we believe MongoDB offers a completely new and different experience compared to MySQL, and we are eager to see what kind of sparks MongoDB + Spring Boot can ignite.