**Assignment 4: Grocery Express Project - Architecture & Design (v1)**

**Date:** March 27, 2023

**Group Number**: 25

**Group Members**:

Xi Chen: xchen911@gatech.edu

Zhipeng Liu zliu352@gatech.edu

Weijun Sun: wsun345@gatech.edu

Chenyu Yan: cyan47@gatech.edu

Weijia Zhao: wzhao307@gatech.edu

## ADR1. ROBUSTNESS

### STATUS

Proposed

### CONTEXT

Enhancement of System Robustness, where robustness is defined as the ability to handle error and boundary conditions while running if the internet connection goes down or if there is a power outage or hardware failure.

There is no error handling regarding system crash (internet connection failure, power outage or hardware fair) yet. Any crash will cause a loss of all system information and the administrator will have to re-run all the commands from the beginning.

All the information must be stored in either relational database or non-relational database to resume the system after the program crashes. The status of the system must be rolled back in the case of the program crashing in the middle of an operation by using either @Transactional annotation in Spring Boot or logging all the commands to run when the program is resumed. The system must be recovered with

minimal delay after the program crashes by using either Java Persistence API (JPA) to directly read the data from the database or transferring data into the system when the program is resumed.

**DECISION**

All the information will be stored in the relational database to resume the system after the program crashes. All the operations interacting with the database will be built using @Transactional annotation in Spring Boot to roll back the status of the system when the program crashes. JPA will be used in this application to directly read the data from the database to minimize the delay from the system recovery.

**CONSEQUENCES**

This design will ensure that data could not be lost due to the system crash and system could not be recovered with minimal delay.

In addition, the status of the system will return to the previous status if the system crashes in the middle of an operation.

In general, this improvement will benefit all users by making the system more robust.

**ESTIMATION OF THE COMPLEXITY**

The effort required to implement a modification decision is influenced by the familiarity with the technologies involved, such as Spring Boot and MySQL, and the integration of the new architecture with the existing Docker environment.

**COMPLIANCE**

Additional test cases will be developed by the group to test the outcome of the system and TA (Sarah Kidd) will validate the system upon submission. In particular, we will test (1) whether data can be retrieved after a crash; (2) the time length for the system

to recover from a crash; (3) whether half executed commands will be rolled back after a crash

**NOTES**

Last modified data: March 26, 2023, by group 25

**ADR2. SOLAR-POWERED DRONES**

**STATUS**

Proposed

**CONTEXT**

The current system treats all deliveries the same in the sense that the fuel capacity only tracks the count of delivery instead of the distance to travel for each delivery. There is no concept of time in the current system and delivery/refuel is done instantaneously.

We modify the current system to introduce the concept of solar-powered drones, which consumes fuel when delivering orders and is refueled automatically by the energy of the sun if not enough fuel to move to another valid location.

**DECISION**

In terms of additional variables: we will introduce current time as the new system variable that is accessible for all the entities in the system. We will add the location information for stores, customers (or orders) and drones. We will also introduce the concept of airspeed, charging speed (day/night based on current time), fuel consumption speed, available time for the drones.

When a consumer places an order, the system will attach this order to a drone that is currently available. Based on the location of the store and the destination of the order or the customer, the drone will calculate the time to delivery, taking into account both

the travel time and refueling time (if applicable). We allow the drone to refuel multiple times along the way if the distance between the store and order destination is long enough. The drone will be unavailable to this store until it comes back, which is defined in the way that the time when it starts the delivery plus the period/time length of the delivery exceeds the current time.

**CONSEQUENCE**

Overall, this solar-powered drone feature will make the system more realistic. It will introduce the concept of time/distance and help the system administrator/stores to better track the availability of drones and the efficiency of the process.

UPDATE: Upon review at the March 26th, 2023 meeting, we determine this feature should be implemented though the estimated time of the solar-powered drone feature will be around 5 hours according to some preliminary exercises.

**ESTIMATION OF THE COMPLEXITY**

After some preliminary exercises, we estimate the time it takes to add the solar-powered drone feature to be around 5 hours.

**COMPLIANCE**

Additional test cases will be developed by the group to test the outcome of the system and TA (Sarah Kidd) will validate the system upon submission. In particular, we will test that the system can calculate the time to deliver an order (and thus allocate the available drones) correctly, under different cases when: (1) no need to refuel the drone along the way (2) need to refuel the drone in daytime or at night or both.

**Notes**

Last modified data: March 26, 2023, by group 25

**ADR3 TIME-SENSITIVE COUPONS**

**STATUS**

Proposed

**CONTEXT**

We modify the current system to introduce a new capability for customers to receive digital coupons randomly based on their ratings, which can be used to reduce the cost of their orders within a specified time frame.

There is no concept of coupons in the current system and consumers face the same price for a given item.

**DECISION**

The functional improvement will offer system users four configuration options:

(1) Users can set the length of the coupon code, and the system will generate a unique code consisting of random numbers and letters for each coupon;

(2) Users can sort customers based on their rating and select one or multiple groups to receive coupons;

(3) Users are allowed to define the number of customers selected randomly in each group;

(4) Users can define the frequency of customers receiving coupons in each group.

In order to ensure fairness, customers' eligibility will be verified when they are selected to receive a coupon. For instance, if the system is on its tenth distribution round, and the frequency for this group of customers is set to 30%, a customer who has received two coupons so far is eligible for another coupon, while a customer who has received three coupons is not eligible. The system will continue selecting customers until the predetermined number of customers is reached, or all customers in the group have been checked. To ensure customer satisfaction, the system mandates that for a redeemed coupon, the order must be delivered before its expiration date. To

determine this, the delivery time for the order fulfilled by solar-powered drone will be compared against the coupon's deadline. If the delivery time surpasses the coupon's deadline, the store must refund 10% of the order cost to the customer.

**CONSEQUENCE**

Overall, this coupon feature will make the system more flexible. It will benefit customers and stores by promoting customer satisfaction and driving stores' revenue growth. Although the risk for the store is that they must refund 10% of the order cost to the customer if an order purchased with a coupon cannot be delivered before the coupon's expiration date.

**ESTIMATION OF THE COMPLEXITY**

After some preliminary exercises, we estimate the time it takes to add the time-sensitive coupons feature to be around 5 hours.

**COMPLIANCE**

Additional test cases will be developed by the group to test the outcome of the system and TA (Sarah Kidd) will validate the system upon submission. In particular, we will test that (1) the system can distribute coupons to customers randomly (possibly based on ratings); (2) customers can only use unexpired coupons; (3) applying coupon will cause a reduction on the overall price of an order; (4) stores failing to deliver orders on time will refund 10% of the order costs to customers.

**Notes**

Last modified data: March 26, 2023, by group 25

**TITLE: PLANNED TECHNICAL DEPLOYMENT**

The application will be written in Java and built using the Spring Boot Framework. MySQL is chosen as the relational database management system for this application.

JPA will be used to map Java objects to relational databases and perform database operation: querying, inserting, updating, and deleting data. In order to maintain the portability, we will continue to wrap up the entire implementation process. This application will also be built and run in docker by using docker compose. To verify the implementation results, we will develop additional testing cases covering the new features of the system and also provide a short video demo.

**CONSEQUENCES**

Following the deployment plan, we will successfully implement the Grocery Express Project with newly added features.