Project Specification – ASE Delivery

Practice makes perfect. To provide an environment for practicing the concepts that you learn from the lecture, we devise an optional project work for running a logistics business. This document contains the project descriptions, requirements, grading scheme, and suggested roadmap to communicate our expectations and recommendations for the project work.

Motivation

Logistics is an indispensable element in human society. It enables intercontinental trade, optimizes supply chains, and facilitates deliveries of goods from pizza to medicine.

With the support of modern digital technology, couriers and delivery services offer a wide range of services to their customers, to name just a few:

- Conveniently purchase items online,
- Send gifts (anonymously) to their loved ones
- Run business(es) by exporting and importing goods.

However, with great power comes great possibilities. One of the critical challenges for couriers and delivery services is to ensure reliable, timely, and accurate delivery to their customers. Therefore, in the course project, you will experience a simplified version of the challenge by developing a pick-up station delivery. The core tasks of your service are managing deliveries and verifying whether they are delivered to and collected by the right customer.

ASE Delivery

You and your team will implement a pick-up station (aka. pack station) delivery service (**ASE Delivery**). In this system, each customer who orders items is assigned a box at a pick-up station. The service then delivers the items from a central depot to the customer's box at the pick-up stations.

There are three user roles in **ASE Delivery**: dispatchers, deliverers, and customers. Dispatchers are responsible for managing pick-up boxes and deliveries. Deliverers collect each delivery by scanning the QR code attached to the delivery surface, then transport deliveries to the pick-up box assigned to a customer. Finally, Customers take their delivery from the pick-up box. Each deliverer and customer has an RFID tag to verify if the deliverer or customer can open a box. Each pick-up box has an RFID scanner to read the RFID tag from a deliverer or customer and determine if they are authorized to open the box. The Customers are informed when their delivery is created, delivered, or picked up successfully.

Due to constraints on the hardware, **we assume that each pick-up station has only one pick-up box**. Figure 1 depicts the workflow of the delivery system.

You can learn more about pick-up station models from DHL[1] or DPD[2].

[1] DHL Packstations https://www.dhl.de/en/privatkunden/pakete-empfangen/an-einem-abholort-empfangen.html
[2] DPD Parcel stations: https://www.dpd.com/de/en/pickup-paketshops/

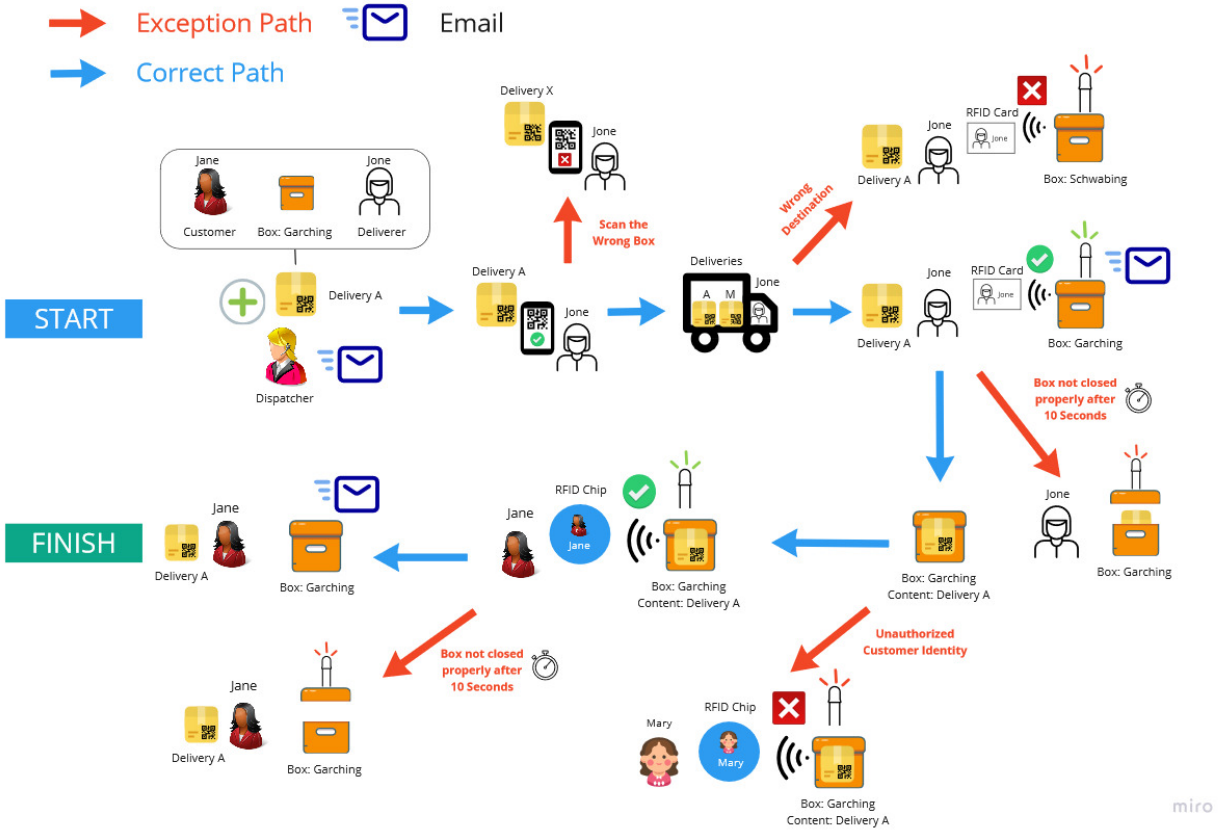

Figure 1: Big picture of the semester project, the ASE Delivery

Requirements & Grading

Requirements

The purpose of the system is to manage the deliveries from a central depot via RFID pick-up boxes to customers.

- Box Management* (*delivery* service) – **1 point**
  - The dispatcher manages pick-up boxes(CRUD). On creation, they configure the street address and ID of the Raspberry Pi.
  - Each pick-up box has a fixed street address and a unique printable name and is equipped with a Rasberry Pi device.
  - The device is connected to an RFID reader that controls the lock of the box.
  - Each pick-up box has a configuration file that will be manually defined and stored in the Raspberry Pi. The configuration file contains the ID of the box in the delivery system, name, and address of the box.
  - The pick-up box is either empty or holds arbitrary many deliveries for exactly one customer. Over time, many customers can use the same box.
  - At a given point in time, all the deliveries in a box have to be for the same customer.
- Box Unlock (*delivery* service) – **1 point**
  - There are two types of RFID tokens to open the lock of a box:

1. Deliverer-Tokens are given to the deliverer transporting deliveries. A Deliverer-Token opens only the boxes designated to its deliverer.
2. Customer-Tokens are given to Customers. They only open a box that contains the good(s) to be delivered to the customer. At a given point in time, a customer may use multiple boxes simultaneously. For example, a customer can use a box near their home or office.

A pick-up box lights a green LED if a deliverer/customer is authorized to open the lock. Otherwise, the box lights a red LED if the user deliverer/customer is unauthorized.

- After a deliverer/customer verifies their token and **closes the box properly**, the system updates the box status. A light sensor (photoresistor) is used to detect whether a box is closed properly by measuring the light intensity inside the box.
- If the box is not closed properly within 10 seconds, the pick-up box shall blink the LED light with red color.
- All the above features must be done using the necessary hardware setup, i.e., no simulation or mock-up of hardware signals is accepted.

- Delivery Management* (*delivery* service) – **1 point**
  - On the creation of delivery, the dispatcher specifies the target box and the target customer, and the responsible deliverer. The system generates a QR code for the created delivery and show it on the Dispatcher's GUI. The Dispatcher attaches the QR code onto the delivery surface.
  - The system ensures that the target box is only used for deliveries of one customer.
  - The deliverer can collect one or arbitrary many deliveries at the central depot by scanning the QR code of the box.
  - The deliverer can deposit a delivery only in the designated target pick-up box. After a successful delivery, the system records that all collected deliveries by this deliverer for this customer have been delivered.
  - The customer can use their token to open a box that contains deliveries for them. After a successful delivery collection, the system records that all deliveries in the box are delivered to the customer.
- User Management* (*delivery and authentication* service) – **1 point**
  - The dispatcher creates a customer from email and assigns an RFID token, and a password to let them log in and track deliveries.
  - The dispatcher creates a deliverer from email and assigns an RFID token, and a password to let them log in and track assigned deliveries.
  - The dispatcher can also create a new dispatcher to manage the system. The new dispatcher is created from email and assigned a password.
- Authentication/Authorization (*authentication* service) – **2 points**
  - Users can authenticate themselves via their ASE Delivery account. The web application sends the password to the server in an encrypted format, so only the server can read the password value.
  - Dispatcher requests accessing deliveries, boxes, or user information need to be authenticated and authorized.
  - Deliverer or customer requests accessing deliveries need to be authenticated and authorized.
  - Customer requests tracking of delivery need to be authenticated and authorized.
- Microservices Architecture (*api-gateway* and *service-registry* service) – **1 point**
  - All services need to be independently executable and deployable microservices written with Spring Boot.

  - An API gateway serves as a single entry point for all clients and proxies/routes requests to the appropriate services.
  - Service discovery is performed by a central service registry entity.
  - The system enforces a Cross-Origin Resource Sharing (CORS) policy to process **only requests from its web application**.
  - The system applies the Cross-Site Request Forgery (CSRF) protection to prevent CSRF attacks.
- Email Notification (*delivery* service) – **1 point**
  - The system sends an email notification to a customer in the following cases:
    1. A dispatcher creates a new delivery for a customer
    2. A deliverer successfully places a delivery inside the pick-up box of the customer
    3. The customer successfully collects all deliveries in their pick-up box
- Delivery Management* (*UI* service) – **2 points**
  - A dispatcher can create, list, update and delete pick-up boxes through the web GUI.
  - A dispatcher can create, list, update and delete deliveries through the web GUI. On the creation of delivery, the dispatcher specifies the target box, the target customer, and the responsible deliverer. The system ensures that the target box is only used for deliveries of one customer.
  - A dispatcher can create, list, update and delete other users through the web GUI.
  - A deliverer can list boxes to which they are assigned. Afterward, they can change the status of their assigned deliveries from Ordered to Picked-up by scanning the QR code attached to each delivery.
  - A customer can list active deliveries and their corresponding pick-up box through the GUI.
  - A customer can see past deliveries and their corresponding pick-up boxes through the GUI.
  - A customer can track their active delivery by inputting the tracking code in the GUI.
- Continuous Integration & Deployment – **2 points**
  - Each microservice lives in its own (Docker) container, to simplify scaling of specific services.
  - Containers of microservices and databases can be orchestrated and scaled independently.
  - The whole ASE Delivery microservice orchestration is continuously deployed and publicly accessible by a web browser. To achieve this, you can use a Cloud Computing Platform (like Amazon AWS[3]), or tunneling of localhost (like ngrok[4]).

Generally, we will guide you through all these requirements in the weekly project exercises.

The required technology stack of the project comprises of React (frond-end), Python (front-end hardware), Spring with Java (back-end), and MongoDB (Database).

[3] Amazon AWS: https://aws.amazon.com/
[4] ngrok: https://ngrok.com/

Grading Scheme

The requirements are structured by the components (which can often be directly mapped to a microservice) and give the amount of points indicated behind them (e.g., 2 points for *Authentication/Authorization*). These points are given **proportionally** to the requirements that you fulfill for a component. That means for each component you get either full points, half of the points, or none. Overall, this leads to a total of 12 possible points. The grading scheme is as follows:

- If you reach at least 6 points, **with Delivery Management, Box Management, User Management and Delivery UI implemented** (5 points), your final grade will improve by one grade step (0.3/0.4, e.g., 1.7 to 1.3).
- If you reach at least 9 points, your final grade will improve by two grade steps (0.6/0.7, e.g., 2.3 to 1.7).
- If you reach 12 points, your final grade will improve by three grade steps (1.0, e.g., 2.3 to 1.3).
- You cannot get a better final grade than 1.0.
- The bonus is only applicable if you pass the final or retake exam.
- We will check the contribution of each team member individually during the final technical demo (see below). If the contribution is questionable or differs significantly across team members, we will adjust the bonus for each individual (i.e., some team members might get less bonus than others).

If we cannot build (i.e., compile) or execute your project, we will not further consider it.

Cheating

We will review every team's code and detect any attempt of cheating or copying code. If we find any code parts to be copied from other project teams, **both** project teams will be disqualified for the bonus.

Deliverables

- Project Gitlab repository containing all source code and documentation necessary to run and deploy the ASE Delivery. We will create a repository for your team in LRZ Gitlab and invite every member to join your team's repository.
- Short document or slide (PDF) containing information about team members and their individual contributions (not more than 1-2 pages). Place this document under a folder with name **contribution** at the root location of your project repository.
- Short technical demo including the deployed ASE Delivery and short code walk-through (no slides)

Organization

- You should form groups of 5 students that work together.
- If any team member quits the project, **inform us immediately and cc the member in the email**.
- If any team member is not reachable, inactive, or does not sufficiently contribute to the project, **you shall handle the matter internally first**. **If the situation does not improve after two weeks, contact us and describe the matter in details so we can make the necessary intervention**.
- **The deadline for project development is around 3 weeks before the exam date.** Project teams can work on their repositories only until then, everything that is added afterwards will not be considered.

Technology Roadmap / Project Schedule

- **Exercise Week 1**
  - Setup Raspberry Pi and Working with RFID Scanner and LED
- **Exercise Week 2**
  - Requirement Engineering and System Design
- **Exercise Week 3**
  - Spring Boot, Maven and Spring Data MongoDB Introduction
- **Exercise Week 4**
  - Introduction to React
- **Exercise Week 5**
  - Redux Slice and Calling REST Services
- **Exercise Week 6**
  - Authentication and Authorization with Spring Security and Java Web Token (Java Web Signature and Java Web Encryption)
- **Exercise Week 7**
  - Secure REST Services with CSRF, CORS protections and HttpOnly Cookies
- **Exercise Week 8**
  - Hardware-Server Communication and Working with Photosensor
- **Exercise Week 9**
  - Integration and Unit Testing
- **Exercise Week 10**
  - Generating and Processing QR Code, Sending Email and Querying MongoDB with Mongo Query and Aggregation
- **Exercise Week 11**
  - API Gateway (Spring Cloud Gateway) and Service Discovery (Netflix Eureka) pattern with Spring Cloud microservices
- **Exercise Week 12**
  - Microservice deployment with Container-per-Service pattern with Docker, Docker Compose and GitLab CI