Computer Engineering Program

Temporary e-commerce platform

By LO, Chi Fung
Supervised by Prof. Wei WANG

Overview

The COVID-19 pandemic has significantly changed online shopping behavior, prompting some tech companies to develop new digital solutions for online shopping.

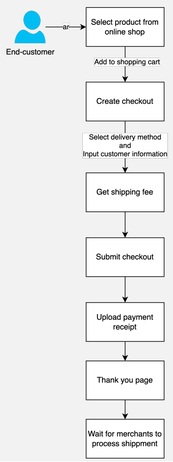
In response, **Appcider Limited** offers **ShipAny**, a software service designed to assist merchants in managing orders from different online platforms and arranging shipping using various courier services.

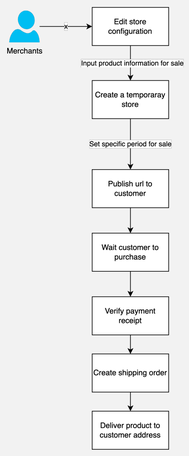
In this project, we aim to build a temporary e-commerce platform within **ShipAny**. This will allow merchants to sell their products in temporary online stores for specific periods.

Objectives

The objectives are divided into two perspectives:
Customer:
1. Receive a hyperlink from merchants and gain access to the stores,
2. Purchase products from online stores and have the products delivered to the shipping address.

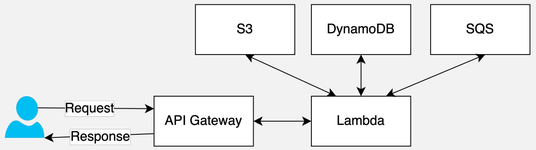
Merchant:
1. Have the ability to publish the hyperlink and update any product information in the store.
2. Check customer order information and verify payment in ShipAny.
3. Have the ability to use all courier services or locations from ShipAny.





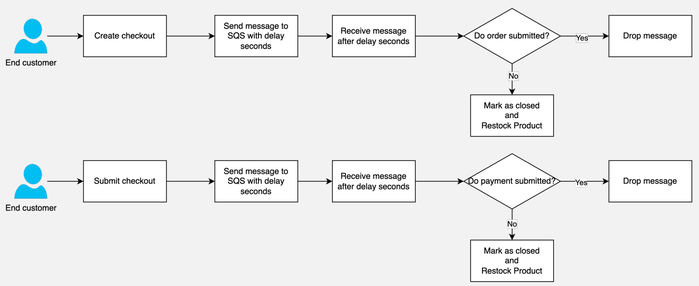
Methodology

Backend system:



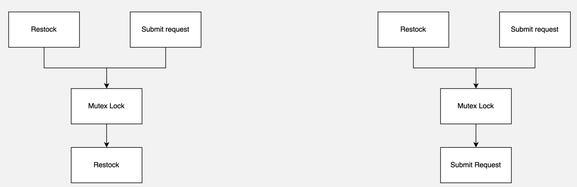
The system was implemented using **AWS** Services. The backend server is deployed to **AWS Lambda**, while **API Gateway** handles endpoint routing. **DynamoDB** and **S3** are used for data storage, and **SQS** is used to control concurrency limits.

Restock system:



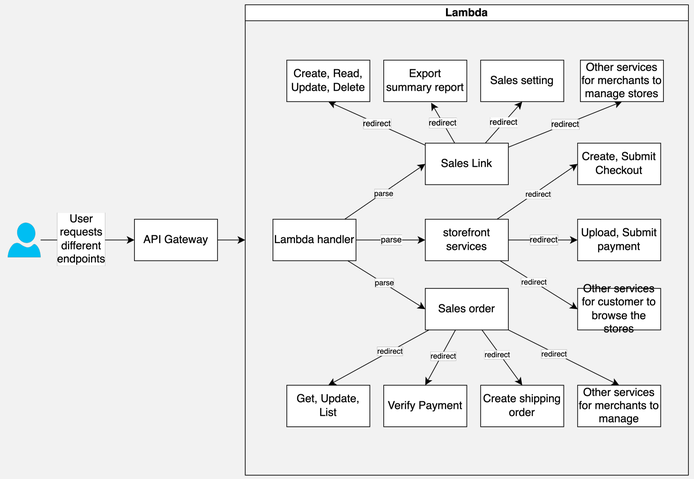
The restock system utilizes **AWS SQS** to check the expiration time of checkout creation and payment submission. However, a **race condition** may occur if the restock system is triggered while the customer is submitting the order or payment simultaneously.

To address this, we can add a **mutex lock** to prevent such **race condition** problems.



Result

API Services



Since the system depends on **AWS Lambda**, which is a serverless platform, we deployed the project as a function and allowed **API Gateways** to directly trigger the event.

Conclusion

Overall, the goals of this project have been successfully completed and it is ready for testing and deployment in a production environment. We can simulate a customer's experience of purchasing a product and having it delivered to the address.

Further improvements could include:
- Online payment
- Email notification
- User Interface
- Refund and Return order handling
- Deployment using Docker