IS213 Enterprise Solution Development

G7T1

Assignment

Asher Chua Yee Liang

Foo Yong Long

Jaslyn Wong

Muhammad Syukri Bin Rahiman

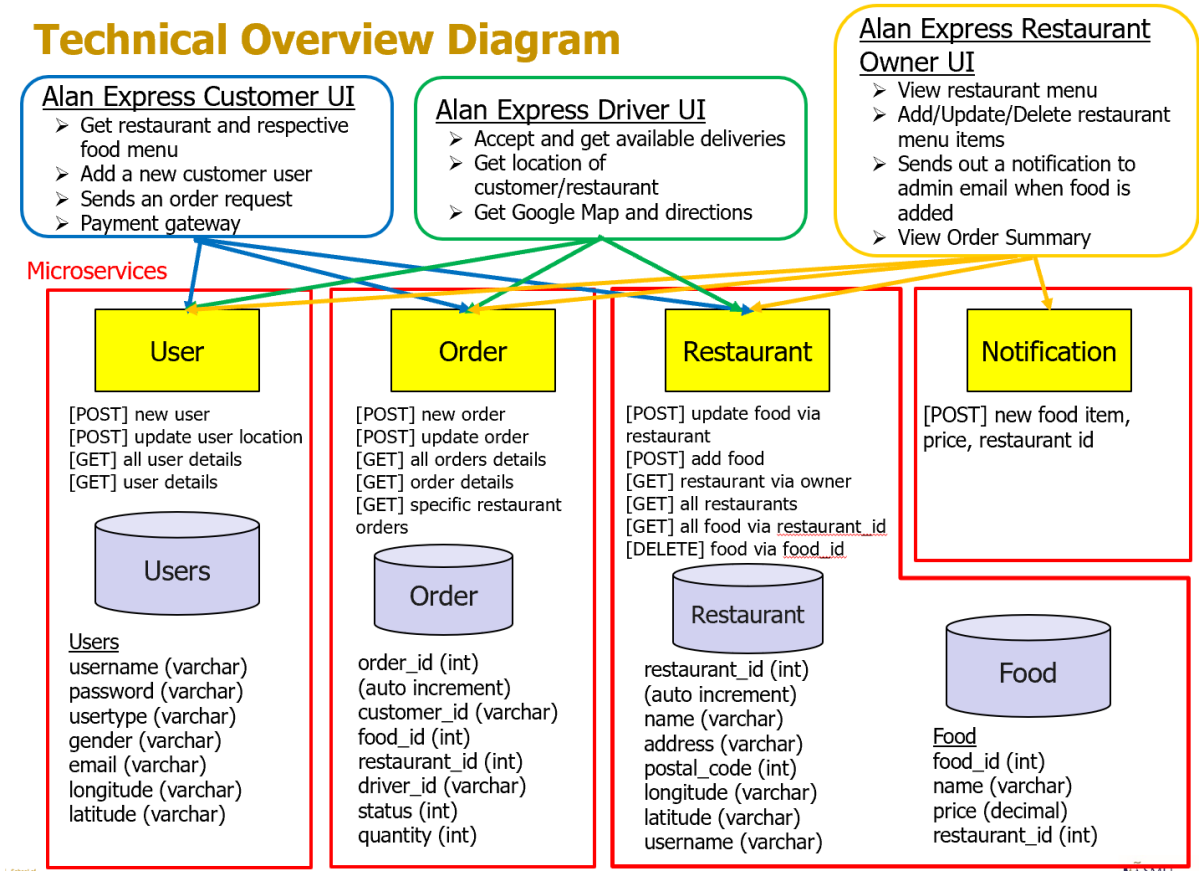
Tay Bingyuan

Tiong Nai Shi

Introduction

Alan Express is an online food ordering site, connecting food couriers, customers and restaurants into a food delivery ecosystem. Using Alan Express, customers can order food from any restaurant on our site and have them delivered by the couriers. Restaurant owners can build an online profile, manage their customers and list their food items. Couriers can access available delivery jobs and accept the orders. Our value proposition lies in integrating the various stakeholders to deliver a seamless experience. ordering site, connecting food couriers, customers and restaurants into a food delivery ecosystem.

Technical Overview Diagram



User Scenarios & Process Definition Diagrams

(Refer to Appendix)

User Scenario 1 – User views restaurants and orders food

Login and view restaurants and restaurant menu

After logging in, the UI invokes the Restaurant service via GET to Get All Restaurants operation, which does a JDBC select from the alanexpress_restaurants via JDBCQuery activity. The service returns the 200 status with all the restaurant details via the getOut activity. This activity is repeated when the user selects a restaurant, the restaurant ID would be passed through the GET operation and all the food details in the restaurant is returned.

Checkout and add order to cart

Upon submission of order, UI invokes the payment gateway via the stripe API. Payment will be received by the restaurant owner which can be tracked via Stripe admin dashboard.

After payment is completed, UI invokes the order service via POST, passing through order_id, customer_id, food_id, restaurant_id and quantity to the operation, in which status and driver_id will both be 0. The service then does a JDBC Update into alanexpress_orders database, orders table, returning a 201 status via the postOut activity, signifying that the order has been successfully added to the database.

User Scenario 2 – Driver view, select & confirm delivery

Login and select order to deliver

After logging in, the UI invokes the Orders service via GET to Get All Orders operation, which does a JDBC select from the alanexpress_order db via JDBCQuery activity. The service returns the 200 status with all the order details via the getOut activity. The UI then lists the orders with status "0". The UI also invokes the Maps service synchronously via Javascript Geolocation API, which returns the latitude & longitude of the driver's current location and prints it on the UI. The UI then invokes the Users service via POST to Add Longitude & Latitude operation by passing the userid, latitude & longitude to the operation, which then does a JDBC Update into alanexpress_user db, Users table via the JDBCUpdate activity to update the driver location. When the driver selects an order to deliver, the UI invokes the Orders service via POST to Update Order operation by passing the status, driverid & orderid to the Update Order operation, which then does a JDBC Insert into alanexpress_order db, Orders table via the JDBCUpdate activity to update the order status. The service returns the 202 status via the postOut activity.

View & complete current delivery

After selecting the order to deliver, the UI invokes the Orders service via GET to Get an Order operation, which does a JDBC select from alanexpress_order db by orderid via JDBCQuery activity. The service returns the 200 status with order details via the getOut activity. The UI also invokes the Restaurants & Users service via GET to Get A User & Get a restaurant from alanexpress_user & alanexpress_restaurant db respectively, via JDBCQuery activity. The service returns the 200 status with user and restaurant details via the getOut activity, which then passes the latitude & longitude information of both the restaurant and the driver onto the UI. The UI then invokes the Maps service via Javascript Direction API, by passing the latitude and longitude information of both the restaurant and the driver to get the route between these locations.

When the driver arrives at the restaurant's location, driver clicks the 'start delivery' button on the UI. The UI invokes the Orders service via POST to Update Order operation by passing the status, driverid & orderid to the Update Order operation, which then does a JDBC Insert into alanexpress_order db, Orders table via the JDBC Update activity to update the order status. The service returns the 202 status via the postOut activity. The UI invokes the Orders service via GET to Get an Order operation, which does a JDBC select from alanexpress_order db by orderid via JDBCQuery activity. The service returns the 200 status with order details via the getOut activity. The UI then invokes the Restaurants & Users service via GET to Get A User & Get a restaurant from alanexpress_user & alanexpress_restaurant db respectively, via JDBCQuery activity. The service returns the 200 status with customer and restaurant details via the getOut activity, which then passes the latitude & longitude information of both the customer and the restaurant onto the UI. The UI then invokes the Maps service via Javascript Direction API, by passing the latitude and longitude information of both the restaurant and the driver to get the route between these locations.

When the driver completes the delivery, driver clicks the 'end deliver' button on the UI. The UI invokes the Orders service via POST to Update Order operation by passing the status, driverid & orderid to the Update Order operation, which then does a JDBC Insert into alanexpress_order db, Orders table via the JDBC Update activity to update the order status. The service returns the 202 status via the postOut activity. The UI shows a thank you message.

User Scenario 3 – Restaurant owner edits restaurant food and views order history of his/her restaurant

Restaurant owner view order summary

When restaurant owner clicks on 'Order summary' on the UI after login, they will be able to view the order history. When clicking on it, the UI invokes the restaurant1 service via GET to get the restaurant_id so that it can invoke the orders4 service via GET to retrieve all orders from the restaurant and the delivery status. Both services will return 200 status with the orders shown on the UI via getOut activity.

Restaurant owners edit, delete or add new food item and price from/to their restaurant

When owners click on 'Restaurant Listing' and then 'edit food listing' on the UI, they can choose to edit, delete or add new food item and price from/to their restaurant.

To edit a food item and price, owner clicks on the edit button on the UI, the UI then invokes the restaurants12 service via POST to update the food name and price. When restaurant owners click on the delete button on the UI, the UI then invokes the restaurants3 service which deletes the food item from the list. The operation does a JDBC update and delete respectively into the restaurant's food database. The service then returns 201 and 203 status respectively with the update and delete shown on the UI via the postOut activity.

To add a new food item and its price, owners can click on the add item button on the UI, the UI then invokes the restaurants2 service via POST to Update new food items by passing the name of the food, price and restaurant_id. The operation does a JDBC insert into the restaurant's food database and a new food will be shown on the UI via the postOut activity. Upon updating the database, the same information is converted to json and passed to RabbitMQ via AMQPSendMessage and received in the notification service via AMQPReceiveMessage. The received message is then sent via email(gmail) to the admin of the website. This is to allow admin to know which restaurants are active and for further analysis.

Web Services

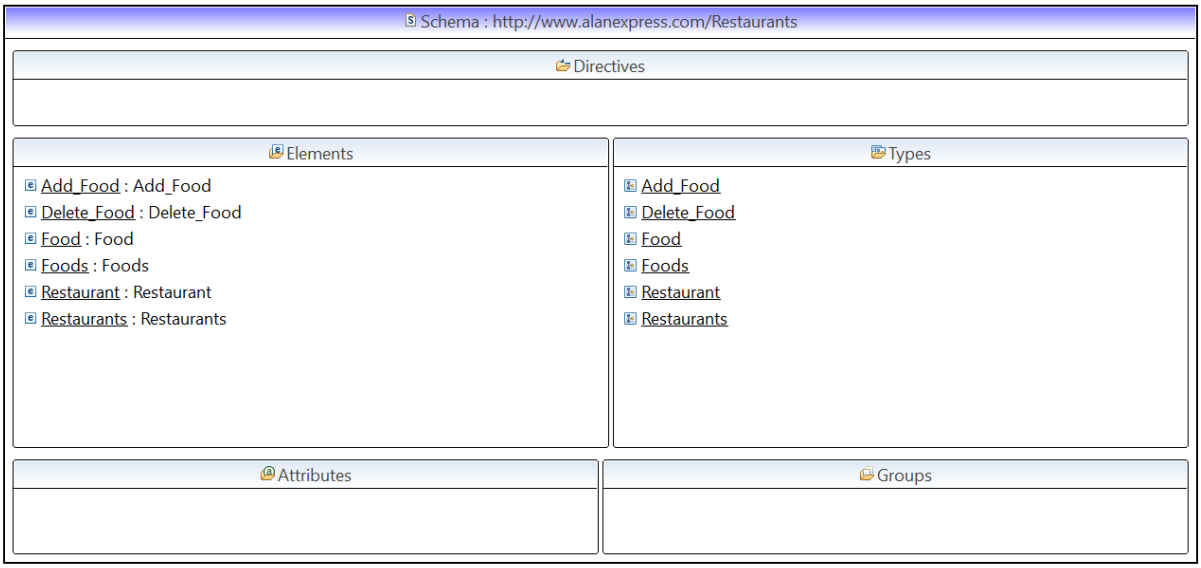
Service	Operation	Description	Input	Output
Users service	Get all users	Getting all the users in user table GET/users	nil	Array of users
	Get a user	Get a user based on the username GET /users1/{username}	username	User
	Add a new user	Adding of a new user POST /users1	username, password, usertype, gender, email, latitude, longitude	NIL
	Add latitude and longitude	Adding latitude and longitude to a user based on the username POST /users1	Username, latitude, longitude	NIL
Alan_Restaurant service	Get all restaurants	Getting all the restaurants in restaurant table GET/ restaurants	nil	Array of restaurants
	Get a restaurant	Get a restaurant based on the username GET /restaurants1/{username}	username	Restaurant

	Get food from restaurant	Get food from the restaurant based on the restaurant id GET /restaurants/{restaurant_id}	Restaurant_id	Array of food
	Add new food	Add a new food to a restaurant Send new food name and price over to notification service via AMQPsendmessage POST /restaurants2	Name, price, restaurant_id	NIL
	Edit food	Editing of a food details inside a particular restaurant POST /restaurants1	Food_id, name, price, restaurant_id	NIL
	Delete food	Deleting of a food based on the food id DELETE /restaurants3/{food_id}	Food_id	NIL
Order service	Get all orders	Getting all the orders in order table GET/orders	nil	Array of orders
	Get an order	Get an order based on the order_id GET /orders2/{order_id}	Order_id	Order
	Add a new order	Adding of a new order POST /orders1	Order_id, customer_id, food_id, restaurant_id, driver_id, status, quantity	NIL
	Update Order	Updating the status of an order based on the order id and driver id which is a username POST/orders3	Order_id, driver_id, status	NIL
	Get orders based on a restaurant id	Get all the orders based on the restaurant_id given GET /orders4/{restaurant_id}	Restaurant_id	An array of Orders
Notification service	Get message via RabbitMQ (using AMQP message)	Received json string over from restaurant2 service using AMQPreceivemessage from AMQPsendmessage	food_id, food name, price and restaurant_id	Json string of food_id, food name, price and restaurant_id
	Send Mail (Gmail SMTP)	Send email to website admin's email account when restaurant owner's adds a new food item and its price.	food_id, food name, price and restaurant_id	Email on updates (Json string of food_id, food name, price and restaurant_id)
Facebook Login API	N/A	Instead of having an account directly for the website, users can use their Facebook account to login	Email, password	To be continued as Facebook username in the list-view

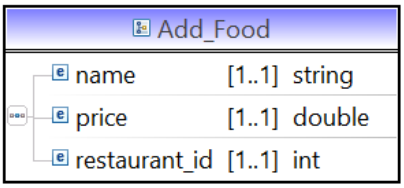
Stripe payment API	N/A	Making payment to check-out orders using credit card details	User card details	Order payment success
Google Maps API	Directions	Obtain the directions between two points for the use of price calculation for a delivery (to restaurant, then to customer).	origins, destinations, key	Route
	Geolocation	Obtain the current location of the user.	key	Current Location

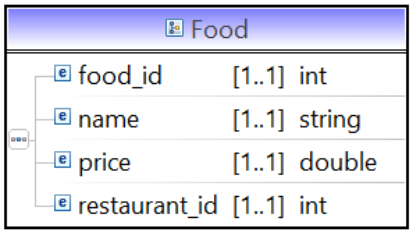
Usage of XML schema

The XML Schema is used in 3 of the microservices. (Orders, Users and Alan_Restaurant). We used the XML Schema mainly for different operations. This could be in the areas of GET, UPDATE or ADD. These operations require data to be passed in and based on the operation, the data passed in could be different. The XML Schema is used mainly to check whether the data passed in is valid. One example of a schema can be seen below.(Orders, Users and Alan_Restaurant). We used the XML Schema mainly for different operations. This could be in the areas of GET, UPDATE or ADD. These operations require data to be passed in and based on the operation, the data passed in could be different. The XML Schema is used mainly to check whether the data passed in is valid. One example of a schema can be seen below.



As seen above, this is an example of the schema used in Alan_Restaurant named Restaurant.xsd. Mainly, we use XML for ADD, DELETE and GET Operations.an example of the schema used in Alan_Restaurant named Restaurant.xsd. Mainly, we use XML for ADD, DELETE and GET Operations.





As seen above, adding of a new food requires the name, price and the restaurant that it is from. Delete food simply requires the food id and for retrieval of a food, we require a food id, name, price and restaurant id. Therefore, we can see how schemas are formed based on the requirements of the microservices and what operations we need.adding of a new food requires the name, price and the restaurant that it is from. Delete food simply requires

the food id and for retrieval of a food, we require a food id, name, price and restaurant id. Therefore, we can see how schemas are formed based on the requirements of the microservices and what operations we need.

Graphical User Interface

HTML5, CSS, Bootstrap, JQuery

Beyond the Labs

Stripe

Stripe developer is a platform to enable payment services. Stripe APIs can be added to enable credit card transactions and payment management from the admin console. This enables business to provide a convenient payment option for users.

This api is invoked when users check out of their shopping cart. The user card details are stated below. Upon successful payment, restaurant can check payment received by logging into their Stripe account. (Refer to restaurant owner account details at 'README.md').

User Visa Details:

Card Number: 424242424242424242 Expiry Date: 12/34 Password: <Autofilled>

Facebook

Facebook developer is a platform for Facebook integration. There are various services, tools and products for third party developers to invoke and create products that requires accessing data from Facebook. This enables convenience for the customers.

This api is invoked when users choose to login Alan Express via Facebook. A Facebook account is required to login to Alan Express. However, the api that we are using is still in development status, hence only the following Facebook account is allowed – email: esd.g7t1@gmail.com password: esdg7t1esd

Google Maps Javascript API

Google Maps Javascript API lets us customize maps with your own content and imagery for display on web pages and mobile devices using their various events, services and libraries.

Our application uses the Geolocation service to capture the users and driver's current location after they log into the application, which will then be updated onto AlanExpress' User database. The Directions service is then used to plot a route, depending on the mode of travel selected (driving, walking, bicycle & transit), based on the locations captured from the Geolocation service and updated onto the database.

Send Mail Activity

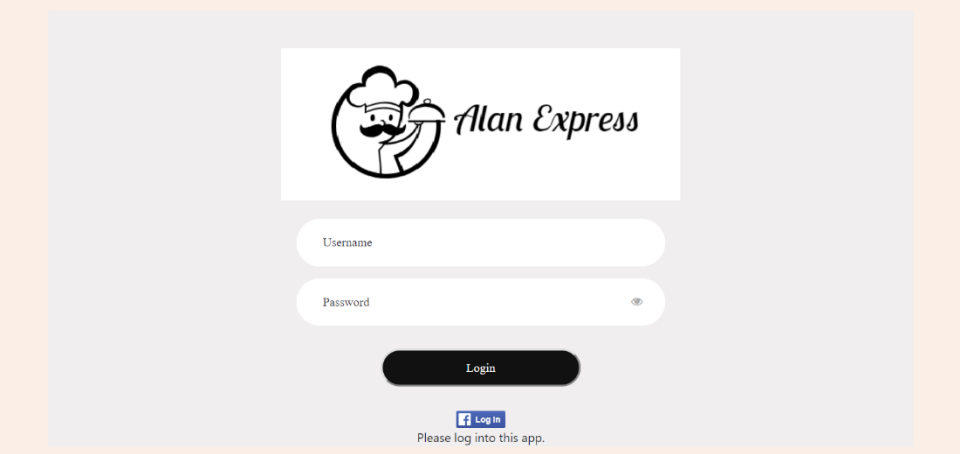
Send Mail is a synchronous activity that sends an email by way of an SMTP server in Tibco BW. Email is sent to specific email address of the Alan Express's admin so that the admin know which restaurants are active and for further analysis.

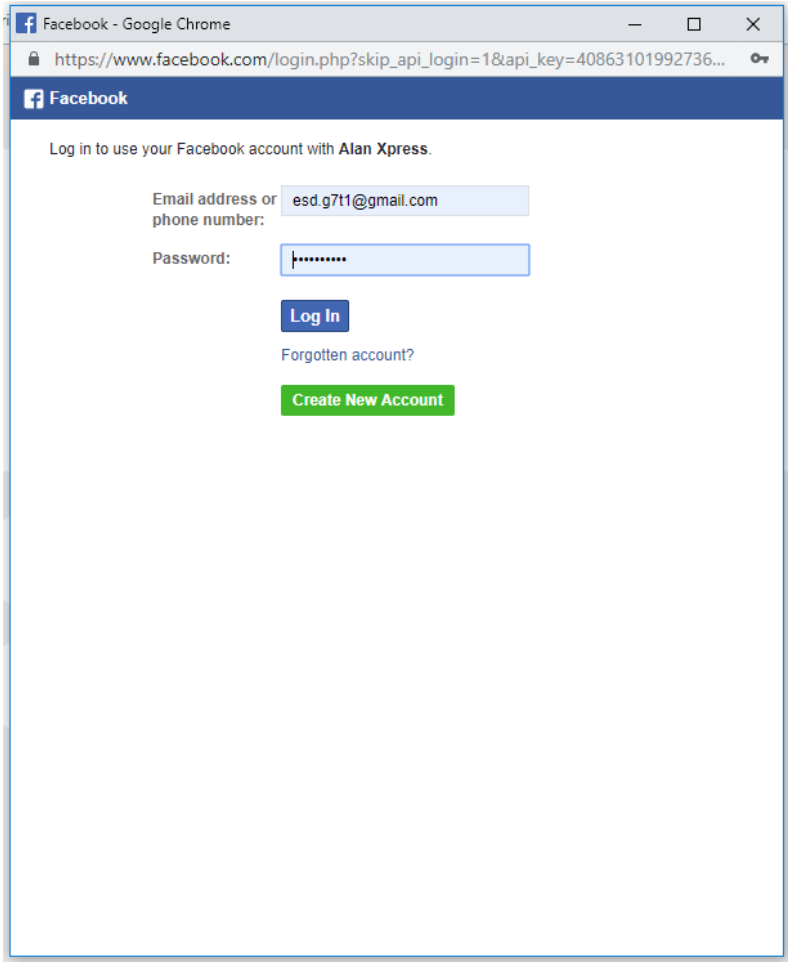
The email is sent via smtp when RabbitMQ is not empty. A username, password and certificate is required to access the smtp host (smtp.gmail.com). To import the certificate into tibco, please refer to the readme file.

Scenario Walkthrough

User Scenario

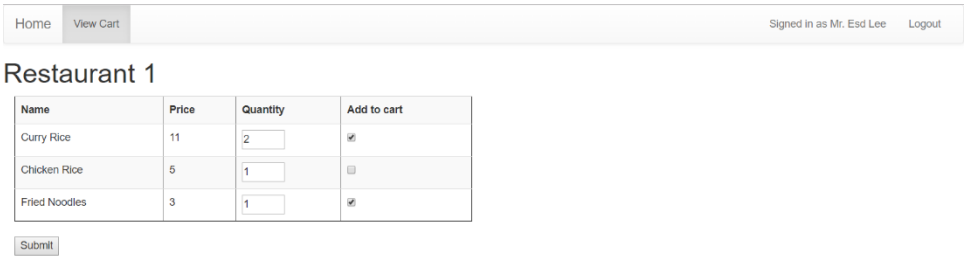
Login with their username or Facebook account via Facebook API



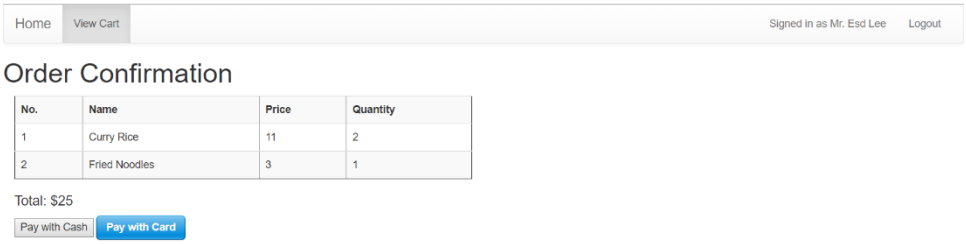


View Restaurant Listing to pick a restaurant to order food

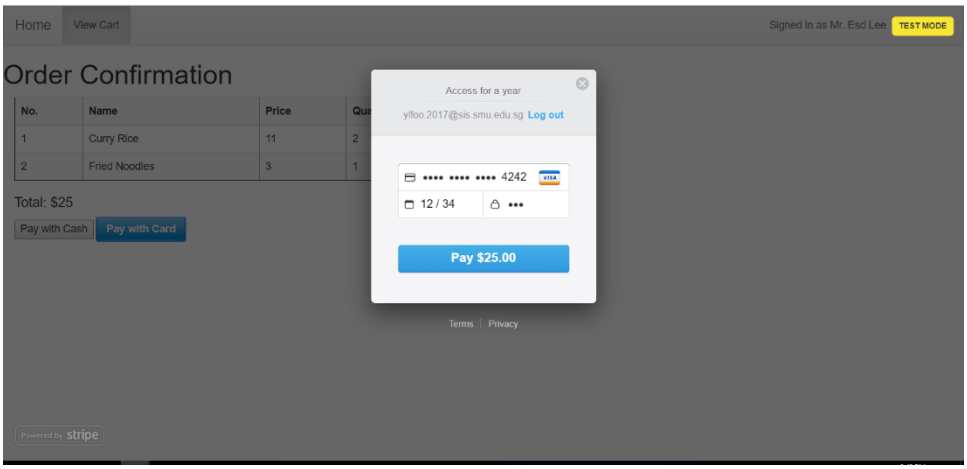
View Food Listing of the restaurant picked to order food



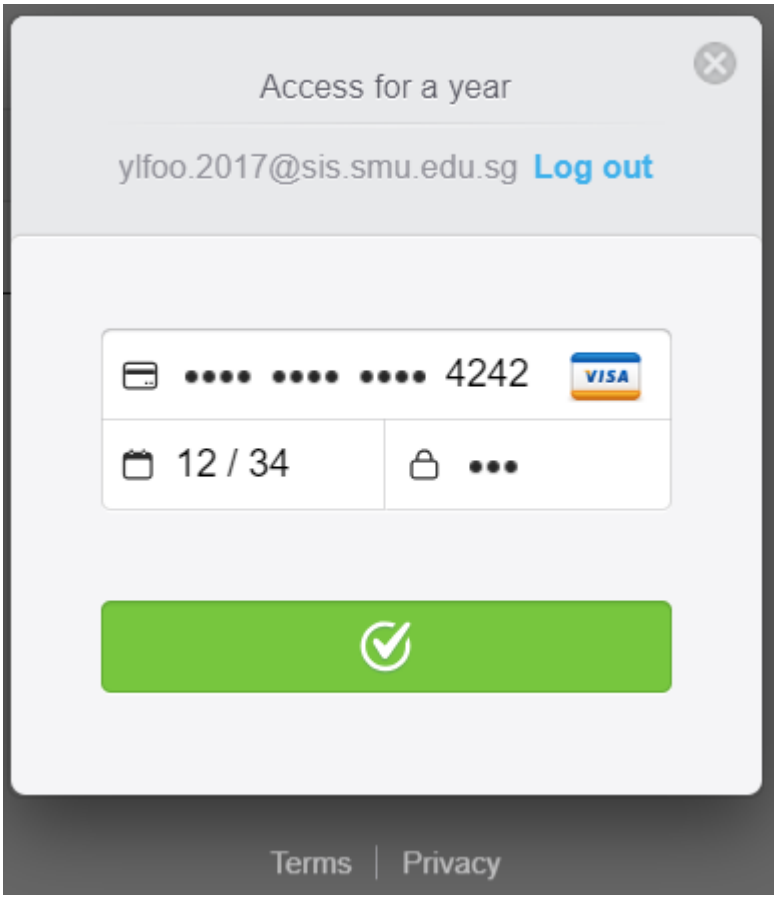
View cart to confirm order and pay with cash or Pay with Card via Stripe API



Payment with Stripe

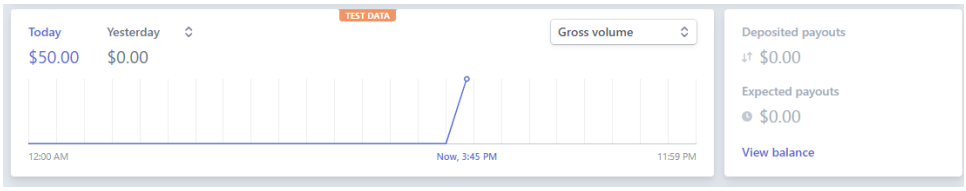


Stripe Paid

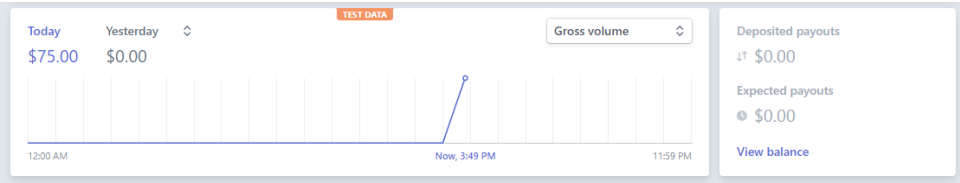


It will reflect in the stripe's admin account after the customer has paid

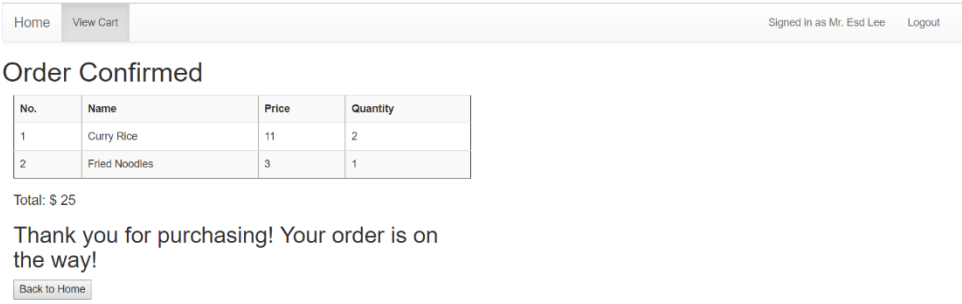
Before



After

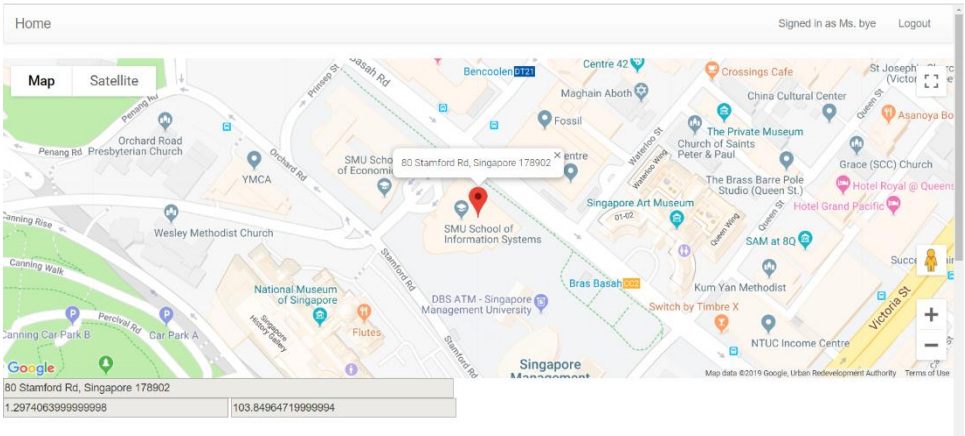


Order confirmed



Driver

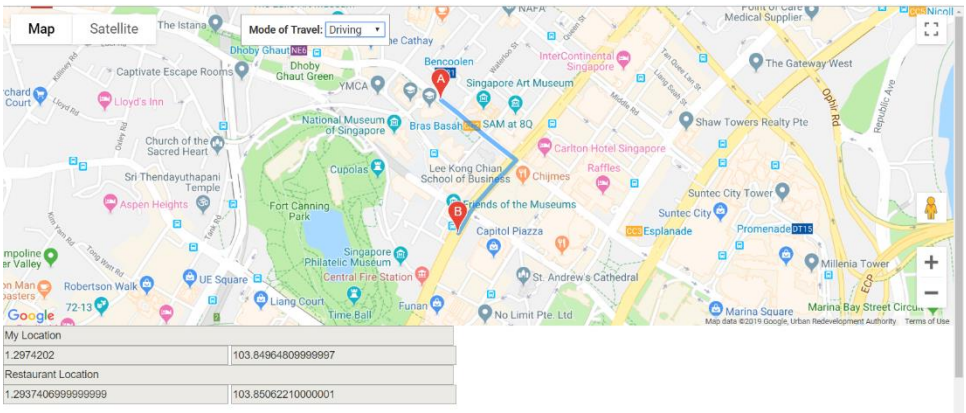
Driver home page: it will show the driver's current location using Javascript Geolocation API and the order listing available



Orders Listing

Order ID	Customer	Food	Qty	Restaurant	
2	Esd Lee	3	1	3	Deliver
2	Esd Lee	4	1	3	Deliver
3	Esd Lee	2	1	2	Deliver
4	hello	2	1	2	Deliver
5	hello	2	1	2	Deliver
6	hello	2	1	2	Deliver
7	hello	2	1	2	Deliver
10	Esd Lee	1	2	1	Deliver
10	Esd Lee	7	1	1	Deliver
11	Esd Lee	4	1	3	Deliver

Order details page – Once the driver clicked ‘Deliver’ on an order, it will lead them to this page. This page will show the direction route to the restaurant from the driver’s location using the Javascript Directions API and the order details.

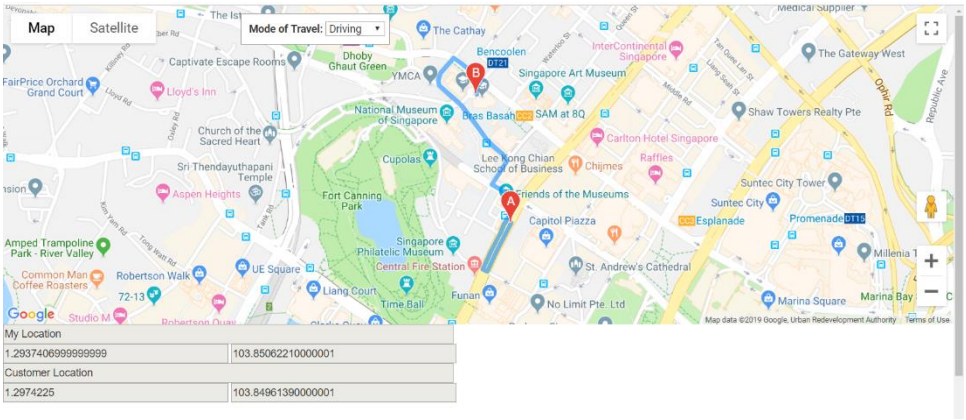


My Location	
1.2974202	103.84964809999997
Restaurant Location	
1.2937406999999999	103.85062210000001

Order Details

Order ID	Customer	Food	Qty	Restaurant	Arrived?
10	Esd Lee	1	2	1	Start Delivery
10	Esd Lee	7	1	1	Start Delivery

Once the driver clicked ‘Start Delivery’, it will change the order status to 1 (means that the order has been picked up and is on the way). It will also lead them to this page where it shows the direction route to the restaurant from the driver’s location using the Javascript Directions API.

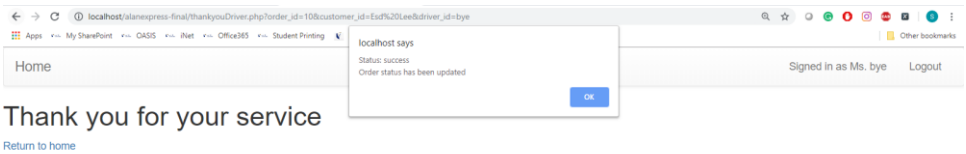


My Location	
1.2937406999999999	103.85062210000001
Customer Location	
1.2974225	103.84961390000001

Order Details

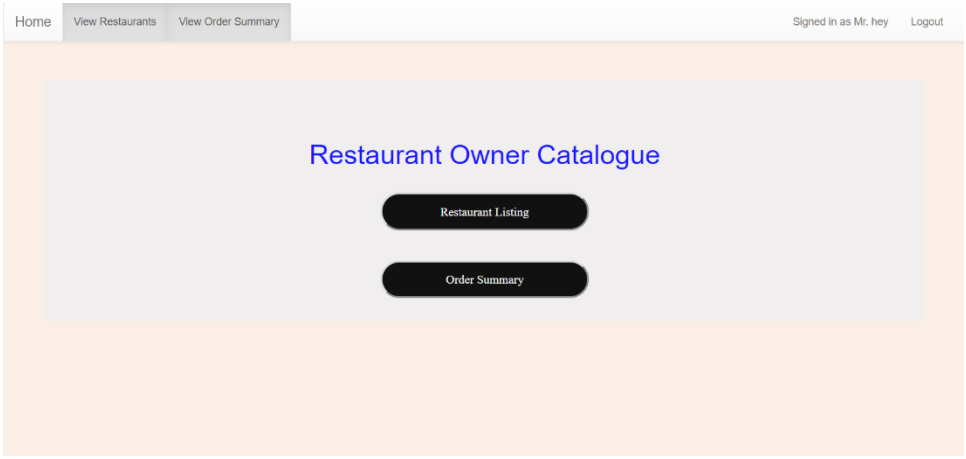
Order ID	Customer	Food	Qty	Restaurant	
10	Esd Lee	1	2	1	End Delivery
10	Esd Lee	7	1	1	End Delivery

Once the driver clicked on ‘End Delivery’, it will lead to this page and update the order status to 3 (which means the order has been delivered) .

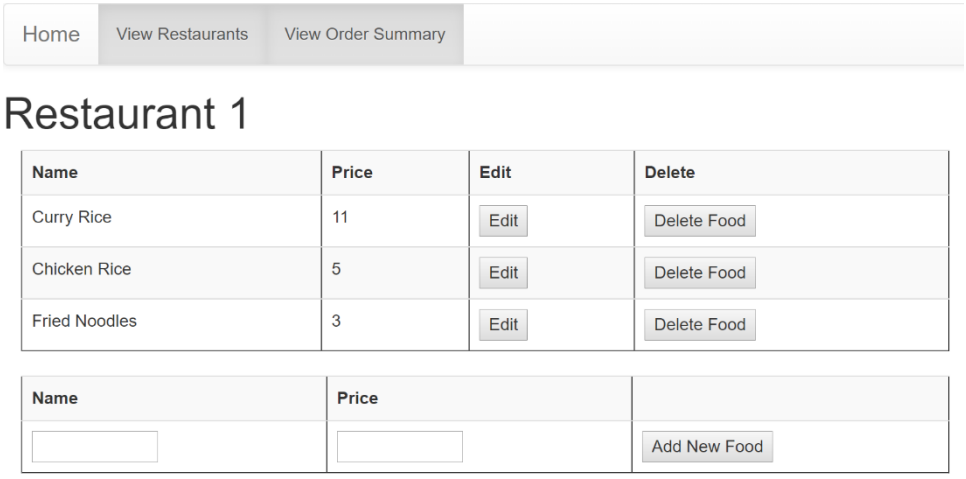


Restaurant Owner

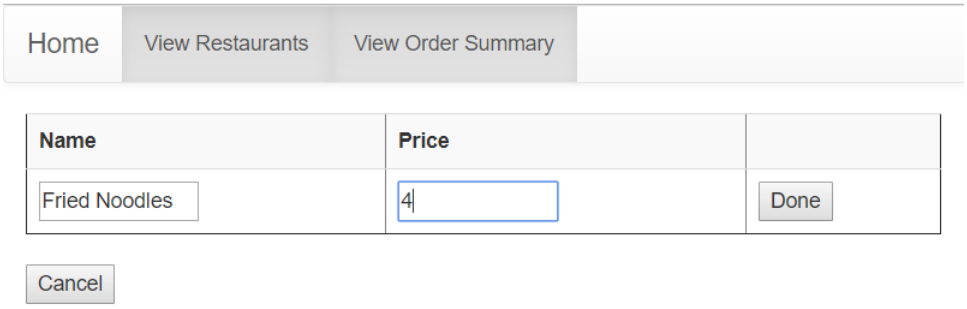
Once Restaurant owner logged in, it will lead them to the Restaurant Owner Catalogue page.

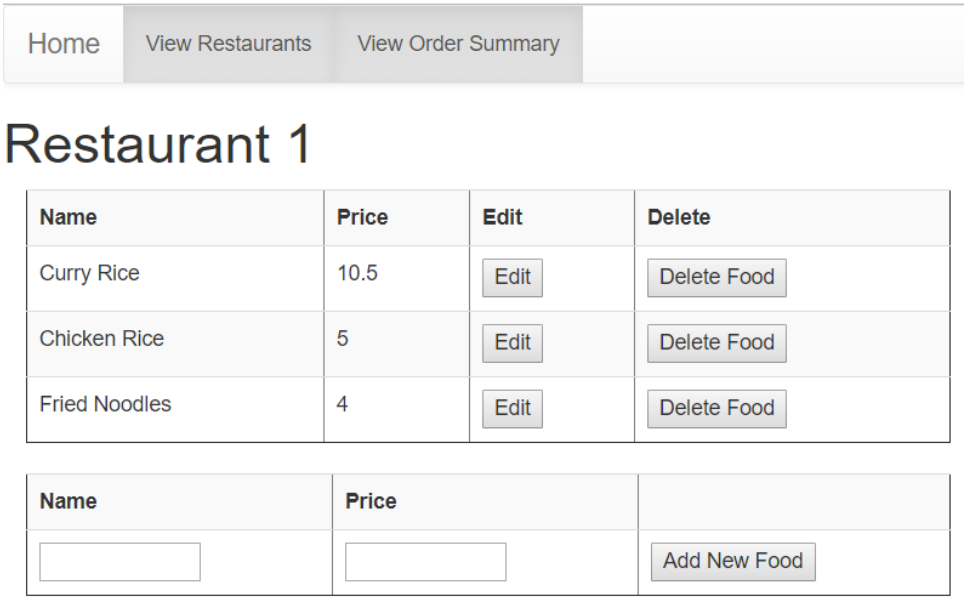


Once the owner clicked on the Restaurant Listing button, they have the option to add, edit or delete a food.



In this example, the owner edit Fried Noodles





In this example, the owner adds pizza into the restaurant.

Home | View Restaurants | View Order Summary

Restaurant 1

Name	Price	Edit	Delete
Curry Rice	11	Edit	Delete Food
Chicken Rice	5	Edit	Delete Food
Fried Noodles	3	Edit	Delete Food

Name	Price	
Pizza	12	Add New Food

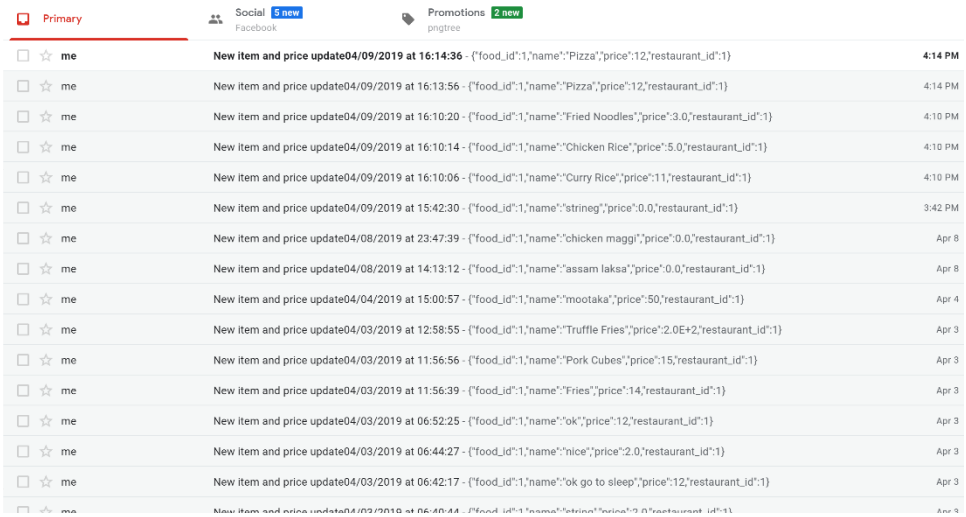
Home | View Restaurants | View Order Summary

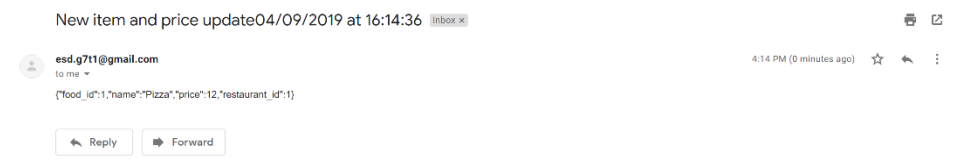
Restaurant 1

Name	Price	Edit	Delete
Curry Rice	11	Edit	Delete Food
Chicken Rice	5	Edit	Delete Food
Fried Noodles	3	Edit	Delete Food
Pizza	12	Edit	Delete Food

Name	Price	
		Add New Food

Once the food has been added, it will send an email to the admin of the application to notify the admin that the owner has added a food





In this example, the owner deletes the food ‘Curry Rice’

Home | View Restaurants | View Order Summary

Restaurant 1

Name	Price	Edit	Delete
Chicken Rice	5	Edit	Delete Food
Fried Noodles	4	Edit	Delete Food
Pizza	12	Edit	Delete Food

Name	Price	
		Add New Food

The owner also can check the history of orders that have been made in their restaurant

Home | View Restaurants | View Order Summary | Signed in as Mr. hey | Logout

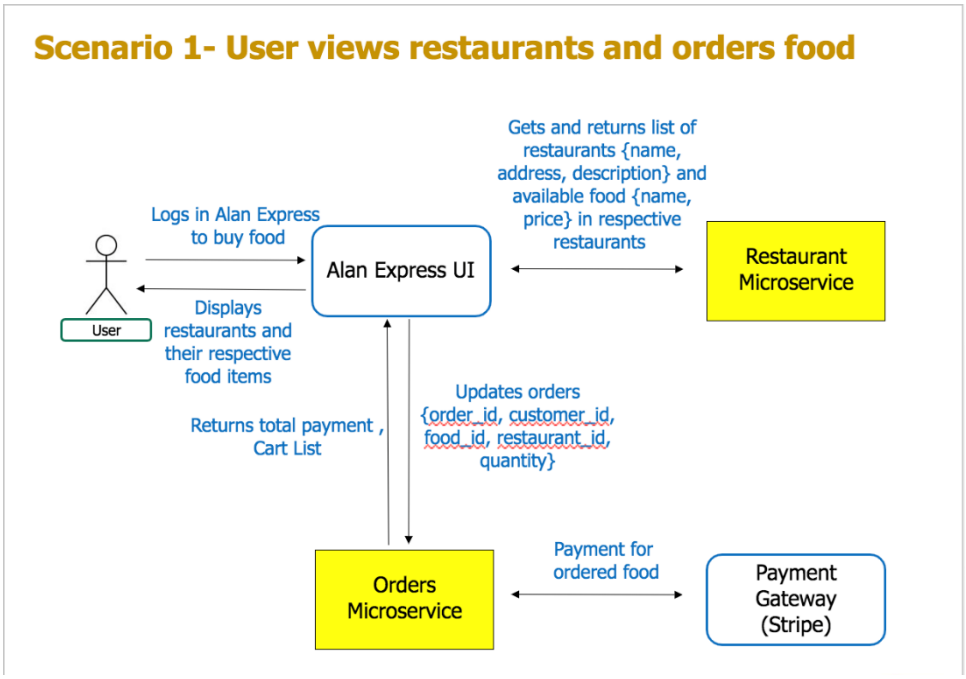
Order History

No.	Customer	Food ID	Quantity	Driver	Status
1	hello	1	3	bye	Order has been delivered!
2	Esd Lee	1	1	string	Someone picked up the order and is on the way!
3	Esd Lee	1	2	bye	Order has been delivered!
4	Esd Lee	7	1	bye	Order has been delivered!
5	hello	6	1	NOT ASSIGNED YET	Order has not been picked up

Appendix

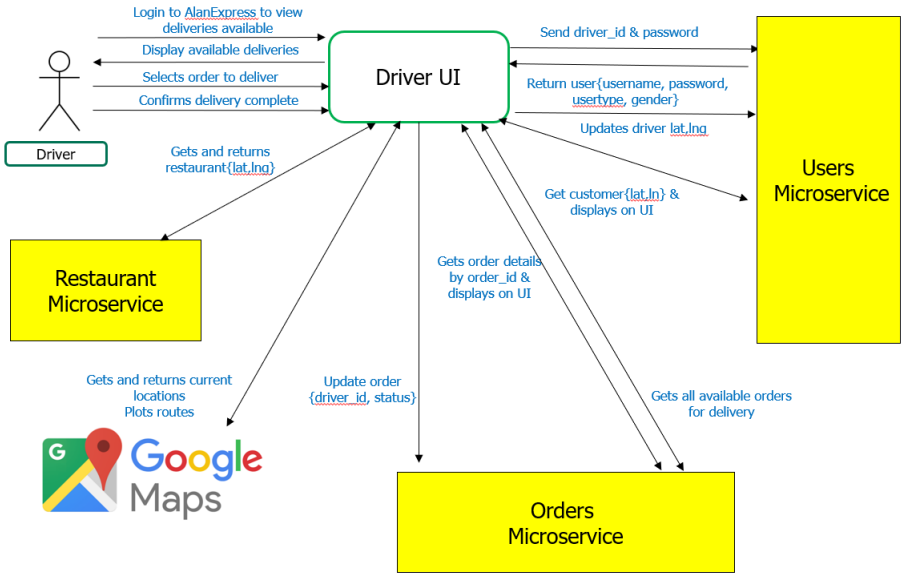
User Scenario Diagrams

User Scenario 1 – User views restaurants and orders food



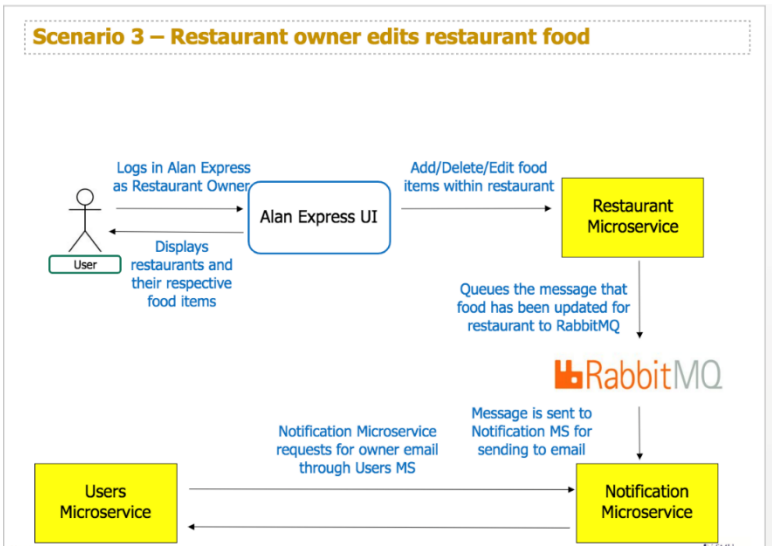
User Scenario 1 diagram

User Scenario 2 – Driver view, select & confirm delivery



User Scenario 2 diagram

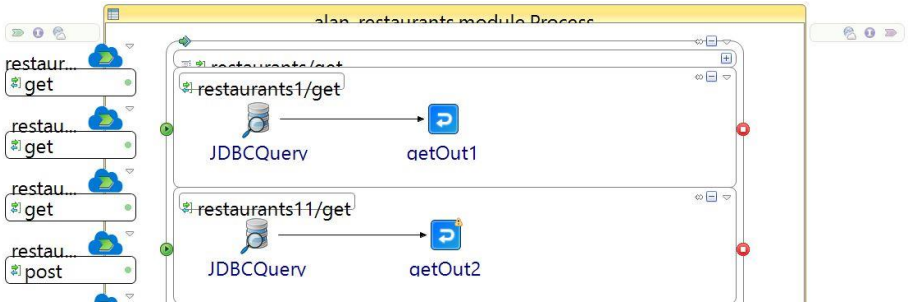
User Scenario 3 – Restaurant owner edits restaurant food

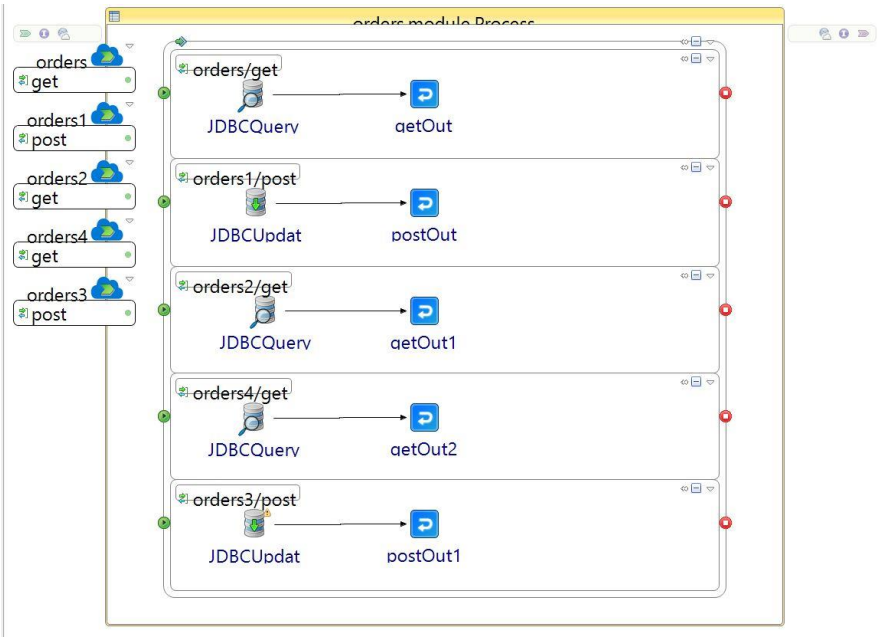


User Scenario 3 Diagram

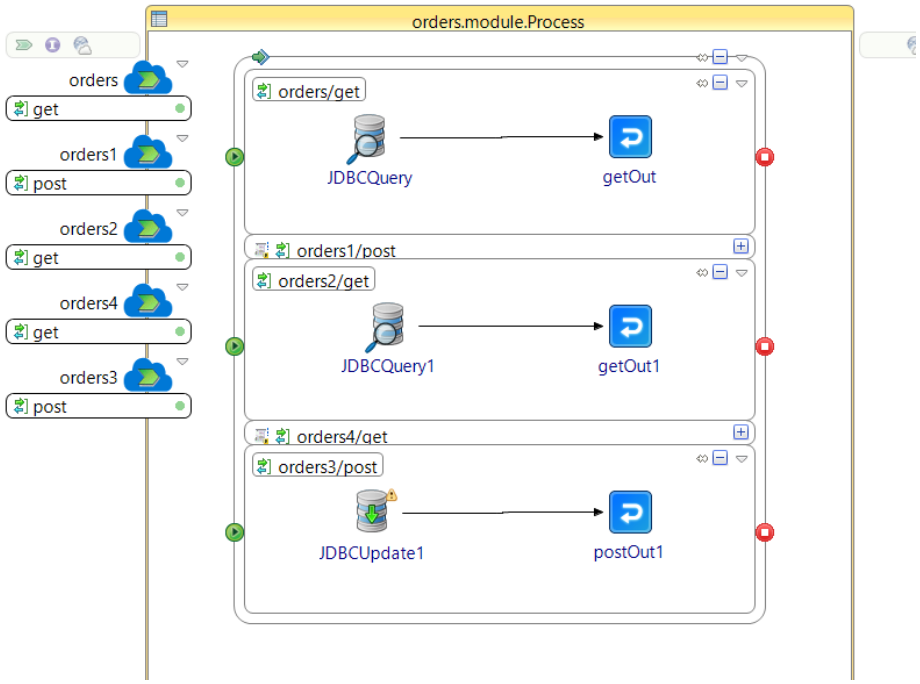
Process Definition Diagrams

User Scenario 1 – User views restaurants and orders food





User Scenario 2 – Driver view, select & confirm delivery



Update Order, Get All Order, Get an Order Process Definition diagram

Get a restaurant Definition diagram

Update User Definition diagram

User Scenario 3 – Restaurant owner edits restaurant food