

# U Eat:

A Dining Concierge App

-Requirements Evaluation & Revision



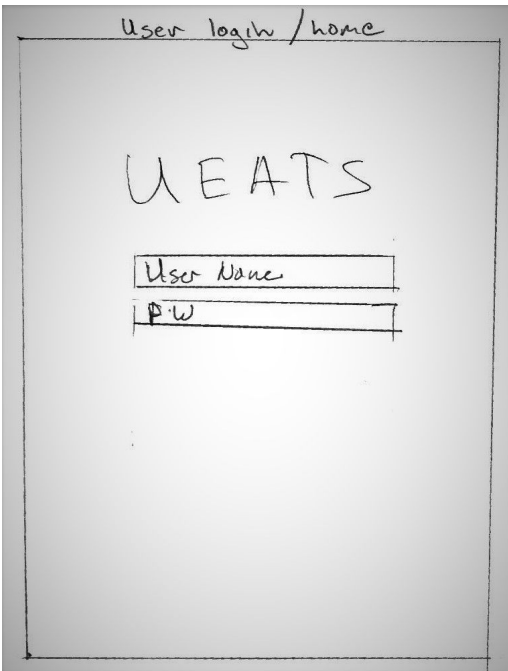
---

## Group 28

Luke Carter  
Ibrahim Mahmoud  
Miles McCoy  
Mathew McDade  
Timothy Tsen

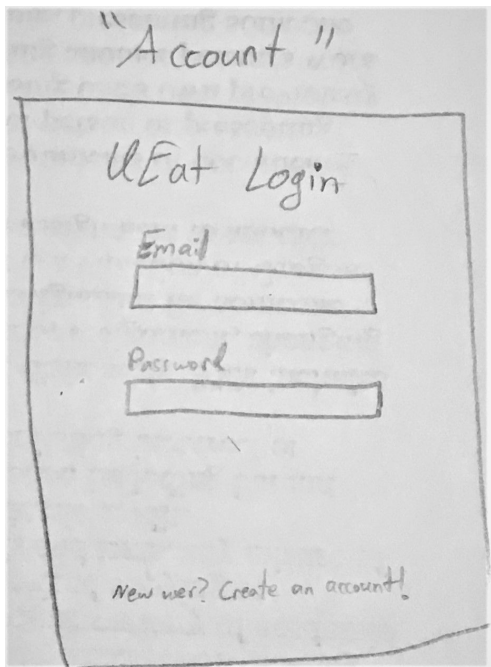
---

## Paper Prototypes:



Account Login A: Simple account login page asks for a username and password. Need to make sure "U Eat" branding is followed in future iterations. Would be improved with a login button below text fields.

\*\*Should it be "U Eat" or "UEat"?



Account Login B: The customer prefers this variation. The customer states a preference for having users login using an email address. The user also liked the inclusion of a 'New user' button at the bottom of the login page. The page would be improved by a login button below the text entry fields.

## Account Creation

First Name\*  
[ ]

Email\*  
[ ]

Password  
[ ] ⓘ

Confirm Password  
[ ] ⓘ

☐ I have read U Eat's User Agreement  
and Privacy Policy


**Create Account**

Last Name\*  
[ ]

Phone  
[ ]

Account Creation: Need to add a payment method registration page before account creation is finalized. Phone number is required, by customer suggestion, so that users can be contacted if there's a problem with their order.

## Main User Page

  
 S.O/S.O  
 Profile  
 Order Status  
 Past orders  
 P.A.Q

Localized Photo  
 Slide Show

Where to? [ ] at

Let's Eat!

Fav. Res One

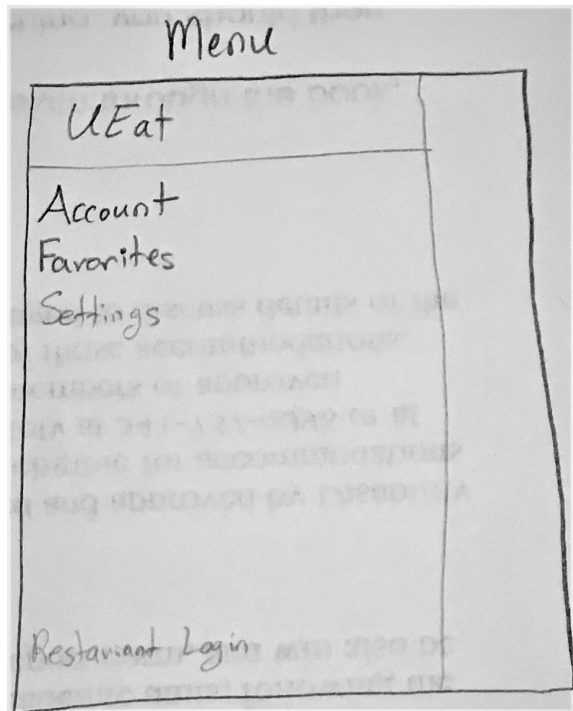
[ ]

Fav. Res Two

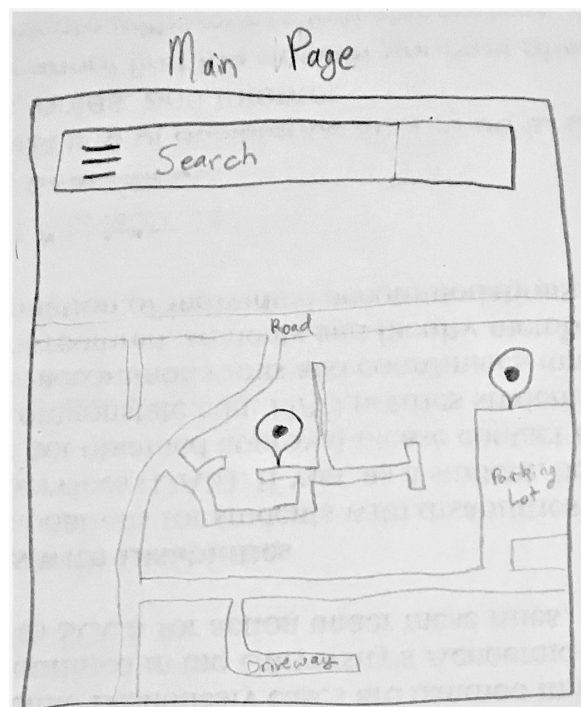
[ ]

↑  
 Float out menu  
 when scrolled over

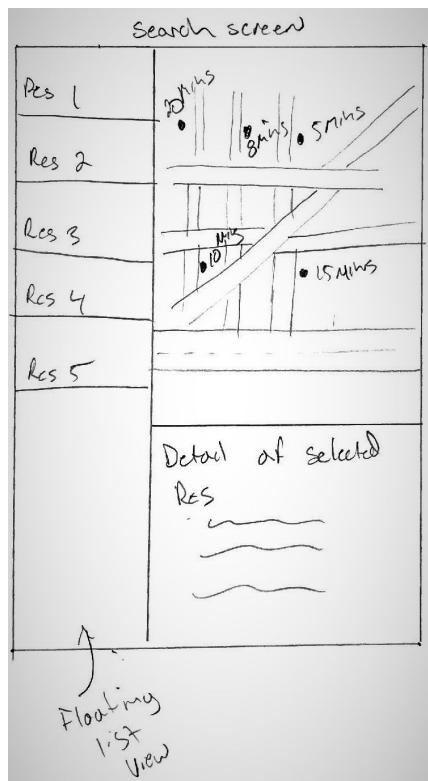
User Main Menu: The customer likes the main user page presented here. This screen combines the most frequent use cases on one screen. Should there be a setting field in the float out menu?



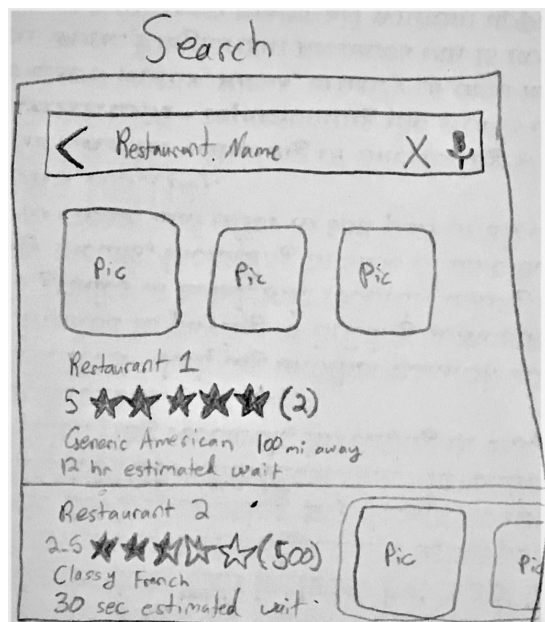
Top Menu: Expand each of these submenus. Is restaurant login part of the user app? The customer preferred the previous main menu page.



Search Page Combined: The customer prefers the clean interface of this search page. The search bar should search by restaurant name. The map should load local restaurant locations. \*\*We need to decide on the number/distance of restaurants to load for this initial search screen.



Search Screen: The customer liked this page for a search results page where selecting one of the results will focus its map position and distance and display basic restaurant information in the details pane.



Search Name: This is a good base for the slide down when name search is selected from the search page.

Menu

My table

(Res controlled here  
photo)

Res selected

Menu	Overview	Reviews
------	----------	---------

Apps

Food item one	Price	Add to table
---------------	-------	--------------

Zents

Food item two	Price	Add to table
---------------	-------	--------------

Restaurant Menu: The customer liked this page a lot. The informational organization got great feedback. Does 'Overview' have restaurant details like hours, contact info, and average wait times?

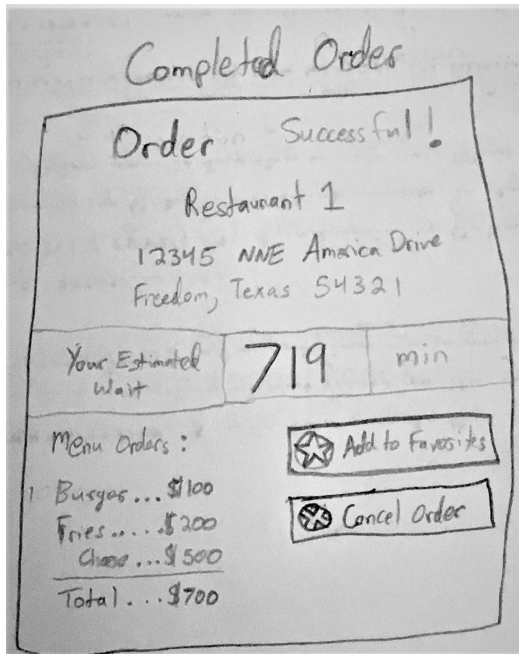
My table

Item one	Price
Item two	Price
Total	\$\$\$

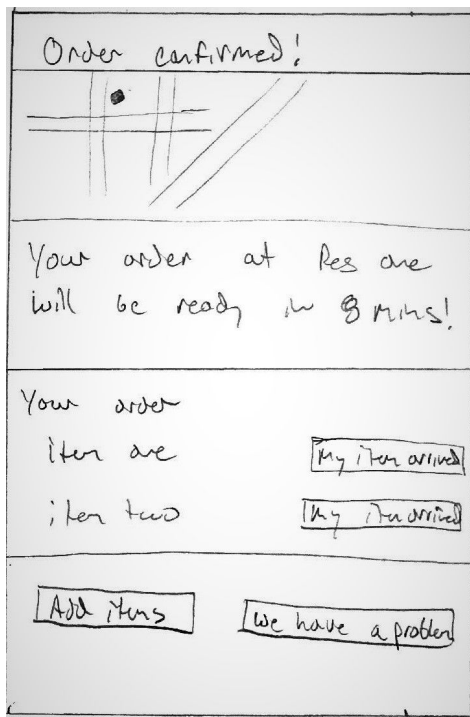
Checkout.

Table Order: The customer thinks this page is a good start for order summary, but it needs to be developed to meet the requirements of being able to specify the number of guests and move items to individual guests. \*\*Should the table reservation process be confirmed before the order is placed?





Order Successful: The customer recommends making this an order confirmation screen so that the customer can make any last second modifications before placing the order




Order Confirmed: This should be our order complete screen with some small changes. The customer doesn't care for the "We have a problem" button heading--can we come up with a more descriptive label?

# Requirements Definition:


## 1. Functional Requirements Definitions:

- 1.1. The customer can create an account that includes a password, phone number, name, and a preferred payment method.
- 1.2. The system can display a list of participating restaurants within a particular area.
- 1.3. The system can display a list of participating restaurants from a name search.
- 1.4. The user can filter search results based on distance from user, rating, and cuisine type.
- 1.5. The system can display a selected restaurant's menu.
- 1.6. The customer can select the number of guests who will be dining.
- 1.7. The customer can add items from the menu to an order and assign that item to an individual guest or for the entire table.
- 1.8. The customer can request a table for a selected number of guests at a specific time.
- 1.9. The system can display the soonest available table for a requested time and number of guests.
- 1.10. The customer can accept or reject an available table offer.
- 1.11. The customer can access a help menu at any time.
- 1.12. The system can process payment for the meal using the customers preferred payment method.
- 1.13. The system can display a confirmation that an order has been completed.
- 1.14. The system can display information about a completed order, including table reservation time and number of guests.
- 1.15. The customer can add items to an order after arriving at the restaurant.
- 1.16. The restaurant must verify diner age prior to delivery of alcoholic beverages.
- 1.17. The restaurant can add and remove items from the menu.
- 1.18. The restaurant can change menu item prices.
- 1.19. The restaurant can set the maximum number of diners.



- 
- 1.20. The restaurant can set available reservation hours limits.
  - 1.21. The system can notify the restaurant of a new table request.
  - 1.22. The restaurant can send the user available table times for the requested number of guests.
  - 1.23. The system can display an order that has been placed to the restaurant, including the table reservation information and the food order.
  - 1.24. The system must display confirmation to the restaurant that payment for an order has been accepted.
  - 1.25. The restaurant can begin a customer visit when the customer is seated.
  - 1.26. The restaurant can end a visit when the customer leaves.

## **2. Non-functional Requirements Definitions:**

- 2.1. Users will be able to log in within 5 seconds.
  - 2.2. The system will display a map of nearby restaurants within 10 seconds.
  - 2.3. The system will display a map of participating restaurants within a 5 mile radius as the default map search view.
  - 2.4. The system will display restaurant search results within 5 seconds.
  - 2.5. The system will display the selected restaurant's menu within 5 seconds.
  - 2.6. The system will allow the customer to select a number of diners between 1 and an upper limit set by the restaurant.
  - 2.7. The system will display a menu item added to an order within 1 second.
  - 2.8. The system will allow the user to select a desired dining time, from the current time to an upper limit set by the selected restaurant on the current day.
  - 2.9. The system will process and confirm payment within 15 seconds.
  - 2.10. The system will confirm customer order and dining time within 15 seconds.
  - 2.11. The system will display an order to the restaurant within 15 seconds of order confirmation
- 

# Use Cases:

## ❖ Case One - Customer places an order and pays.

### ➤ Actors

- Customer
- Restaurant Staff

### ➤ Preconditions

- User downloaded app and created user account
- Restaurant is a “provider” member of UEats
- Restaurant provider uploaded menu and prices correctly
- User has access to network
- Restaurant near user has a table available within a reasonable time period

### ➤ Postconditions

- Customer received food, as ordered
- Customer paid restaurant
- Restaurant received payment

### ➤ Flow of Events

- Customer opens UEats App
- Customer navigates to restaurant, either through search or map view
- Customer reviews wait time to determine if the time is acceptable
- Customer reviews menu items
- Customer selects menu items and place them in their “table”
- Customer reviews their table items
- Customer finalizes order
- Customer pays for order using Apple Pay API
- Restaurant is notified of new UEats order and required time
- Restaurant staff prepare meal and place items on table within required time
- Customer arrives at the time indicated by the app
- Restaurant staff seats customer at prepared table
- Customer eats meal
- Customer confirms receipt of meal
- Customer allocates tip for service
- Customer approves final payment for meal
- Customer leaves restaurant
- Restaurant notified of final payment



---

## ❖ Case Two - Account Creation

### ➤ Actors

- Customer


### ➤ Preconditions


- Customer has downloaded the UEat app on a smartphone
- Customer has a preexisting email account
- Customer has a preexisting phone number
- Customer has a preexisting online payment method
- Customer has access to network

### ➤ Postconditions

- Customer created a UEat account they have access to
- Customer's email contains a confirmation email from UEat about a new account being created
- UEat created a new entry in its database of accounts

### ➤ Flow of Events

- Customer opens UEats App
  - Customer navigates to "Log In"
  - Customer navigates to "Create an Account"
  - Customer inputs their first name in the "First Name" textbox
  - Customer inputs their first name in the "Last Name" textbox
  - Customer inputs their email address in the "Email Address" textbox
  - Customer inputs their phone number in the "Phone Number" numeric textbox
  - Customer inputs a password of their choice in the "Password" textbox, with the minimum requirements of at least one lowercase letter, uppercase letter, number, and special character
  - Custom inputs the same password in the "Confirm Password" textbox, which bars the customer from continuing until the passwords are the same
  - Customer reads and checks that they have read the UEat's User Agreement and Privacy Policy in the checkbox next to "I have read and agree to UEat's User Agreement and Privacy Policy", which bars the customer from continuing until the checkbox is ticked
  - Customer clicks "Sign Up"
- 

- 
- UEat's database of accounts gets updated with the new customer's information
  - Customer selects preferred third party payment option from the available options and confirms UEat app access for charges.
  - Upon completion, the customer will receive a confirmation email with the option to undo their action of creating their account
- 

## ❖ **Case Three – Restaurant adds seasonal items to menu**

### ➤ **Actors**

- Restaurant staff.


### ➤ **Preconditions**

- Restaurant staff have UEat app installed
- Restaurant staff have access to change menu and look of menu.
- Restaurant have pictures, prices, and description of seasonal items.

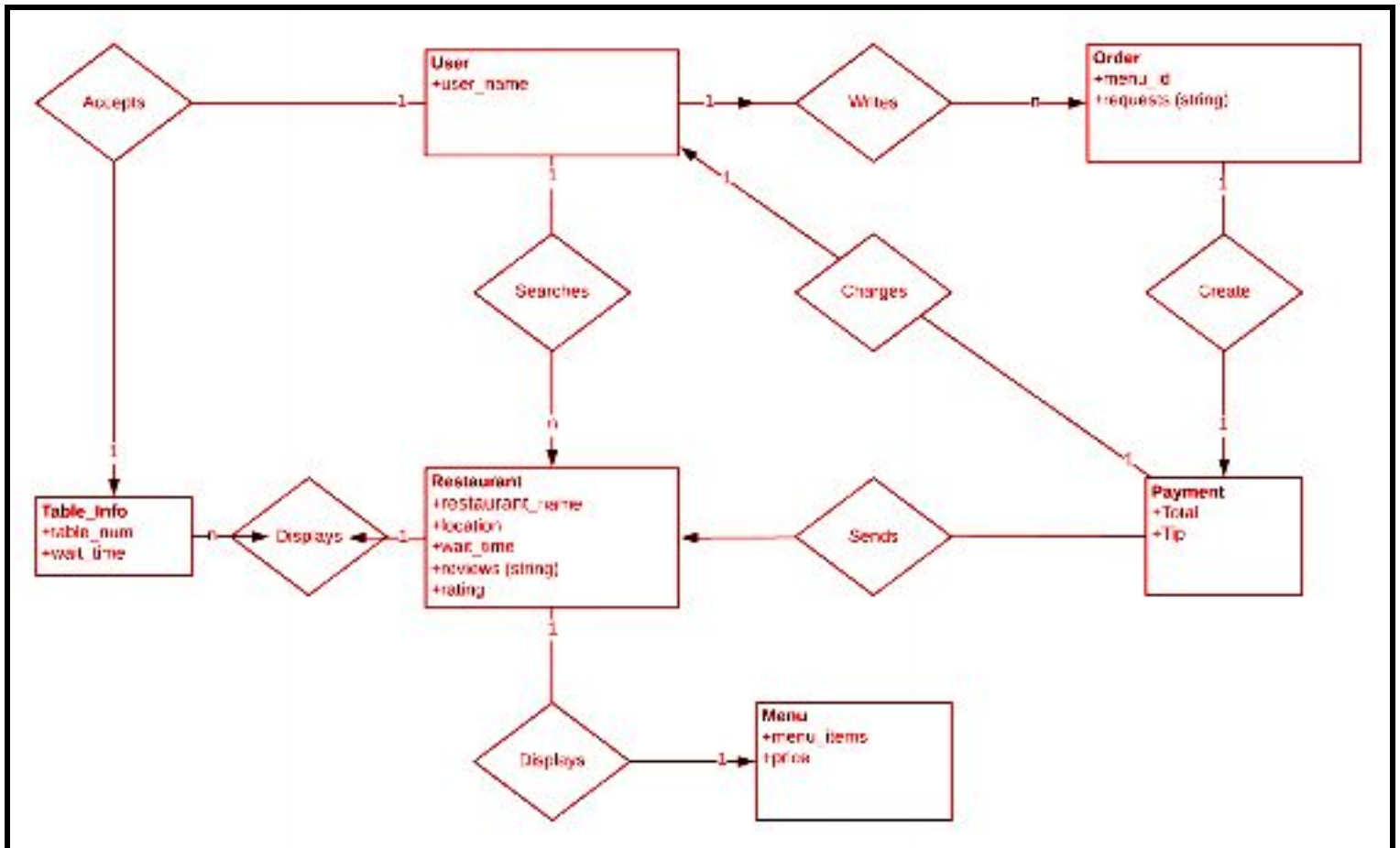
### ➤ **Post-Conditions**

- Restaurant staff member removes seasonal items after season.

### ➤ **Flow of Events**

- Restaurant staff member opens UEats app.
  - Restaurant staff member navigates to change menu.
  - Restaurant staff member Uploads new item's names, descriptions, pictures, seasonal tag, and prices to collection of items for sale
  - Restaurant staff Member Saves new seasonal items.
  - Restaurant staff member adds new items to active menu.
  - Restaurant staff member logs into customer side.
  - Restaurant staff member checks the active menu and sees new items with correct pictures, names, descriptions and prices, and the seasonal tag
- 

# UML Diagram:








# Requirements Specification:

## 1. Functional Requirements Specifications:

- 1.1. Customer account creation will add a new entry to the database user table.
- 1.2. Restaurant account creation will add a new entry to the database restaurant table.
- 1.3. User login credentials will be verified against the database user table.
- 1.4. User location will be obtained via location services.
- 1.5. Nearby participating restaurants will be loaded from the database restaurant table.
- 1.6. Nearby restaurants will be inserted into a map view.
- 1.7. A restaurant name search will query the database for matches or near matches and return the result.
- 1.8. Customer selection of a restaurant will retrieve the restaurant's menu, max table size, and seating hours from the database.
- 1.9. The system can interact with third party payment systems to complete order payment.
- 1.10. After payment transaction is complete, an order is added to the database order table.
- 1.11. When a customer adds items to a completed order, a new suborder is added to the database.
- 1.12. When a restaurant updates the menu, seating options, or hours, the database entries are updated.
- 1.13. The system will close any unclosed visits at the end of the day.

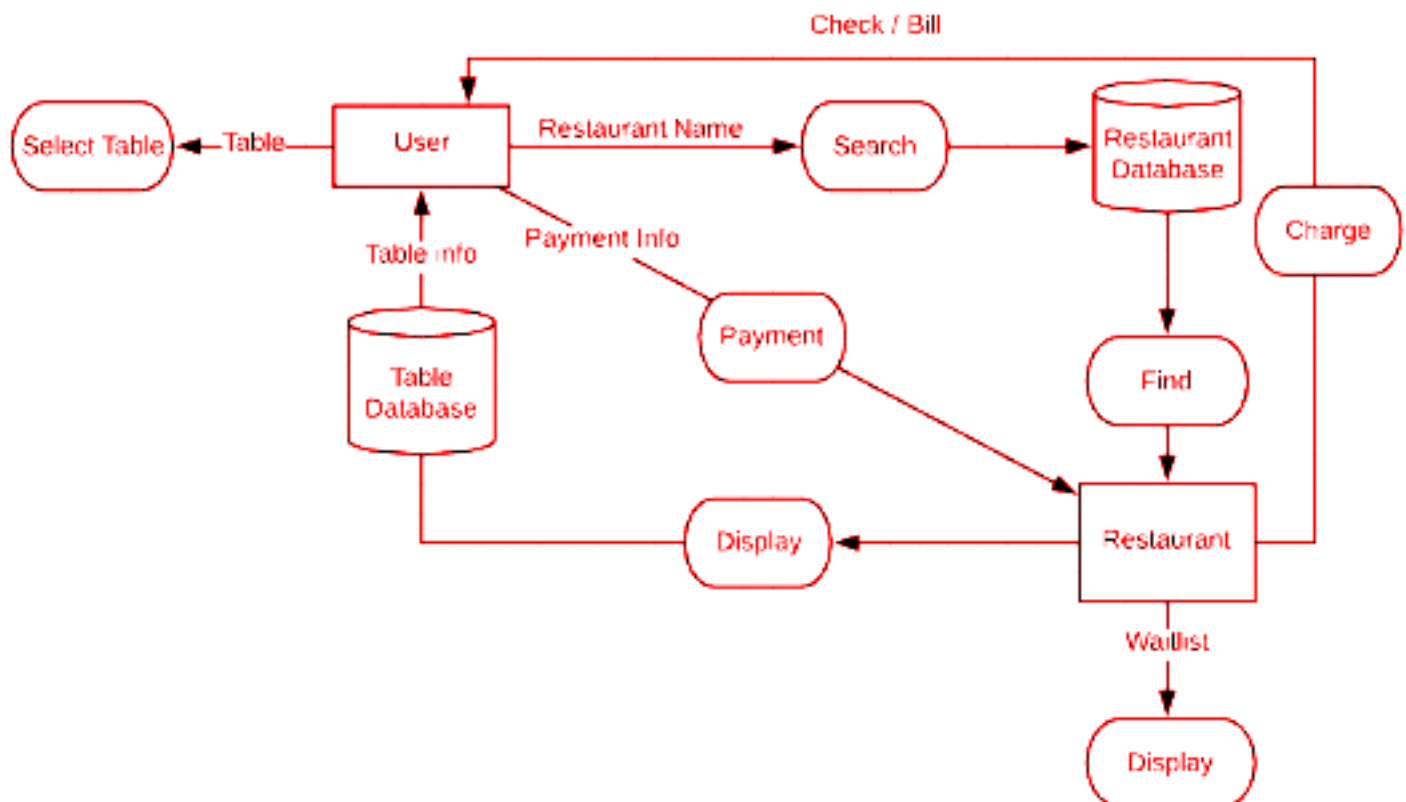
## 2. Non-functional Requirements Specifications:

- 2.1. Customer creation will add an entry to the user table within 5 seconds.
  - 2.2. Restaurant creation will add an entry to the restaurant table within 15 seconds.
- 

- 
- 2.3. Updating restaurant information in the database will update and return the updated restaurant information within 10 seconds.
  - 2.4. User login queries to the database will return within 5 seconds.
  - 2.5. Restaurant search queries will return within 5 seconds.
  - 2.6. The system will limit customer active visits to a maximum of 1 restaurant at a given time.
  - 2.7. The system will limit customer active visits at a given restaurant to 1 at a given time.
  - 2.8. The system will end all of a restaurant's active, not closed, visits 1 hour after the end of a day's business hours
- 

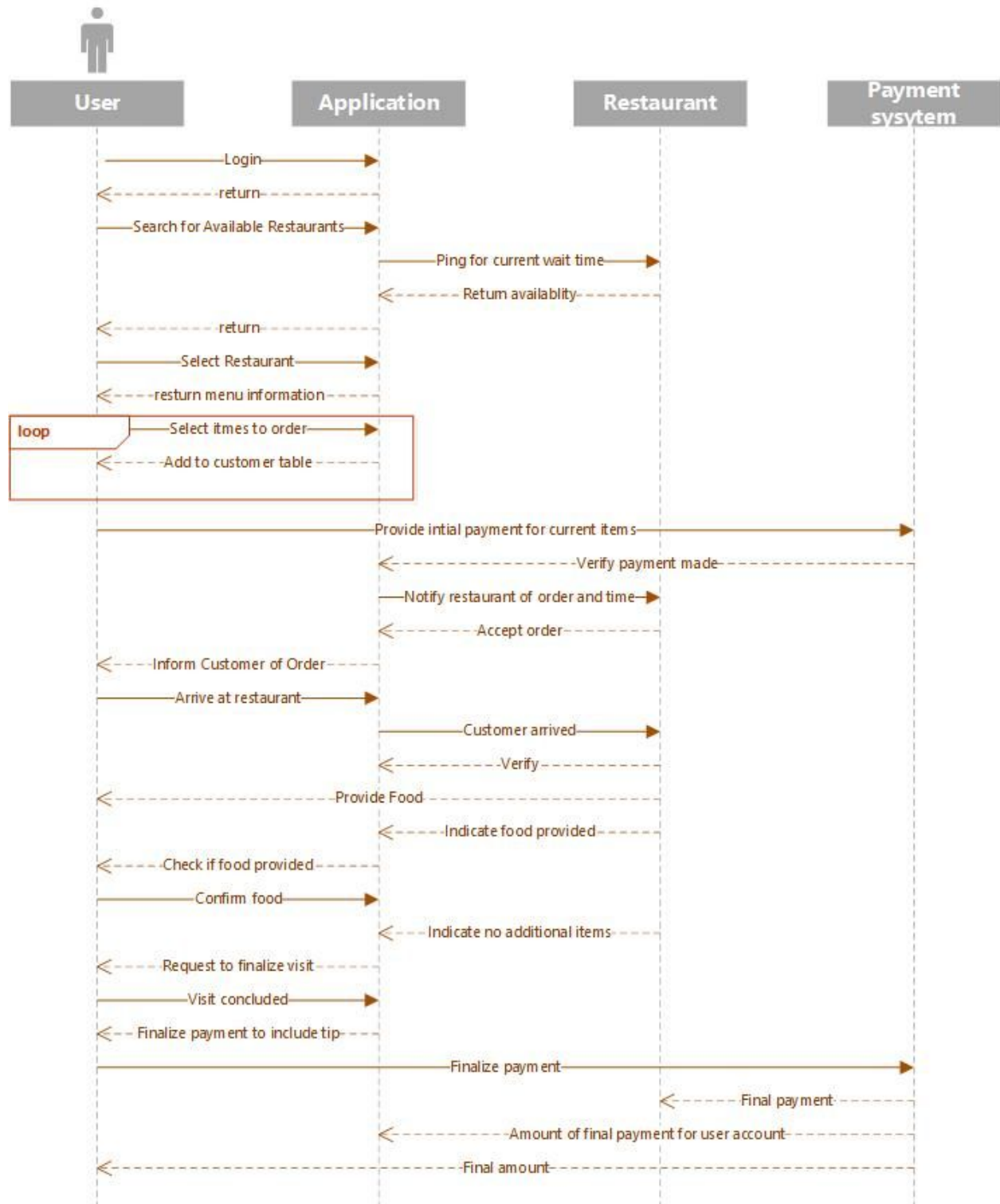


# Dataflow Diagram:

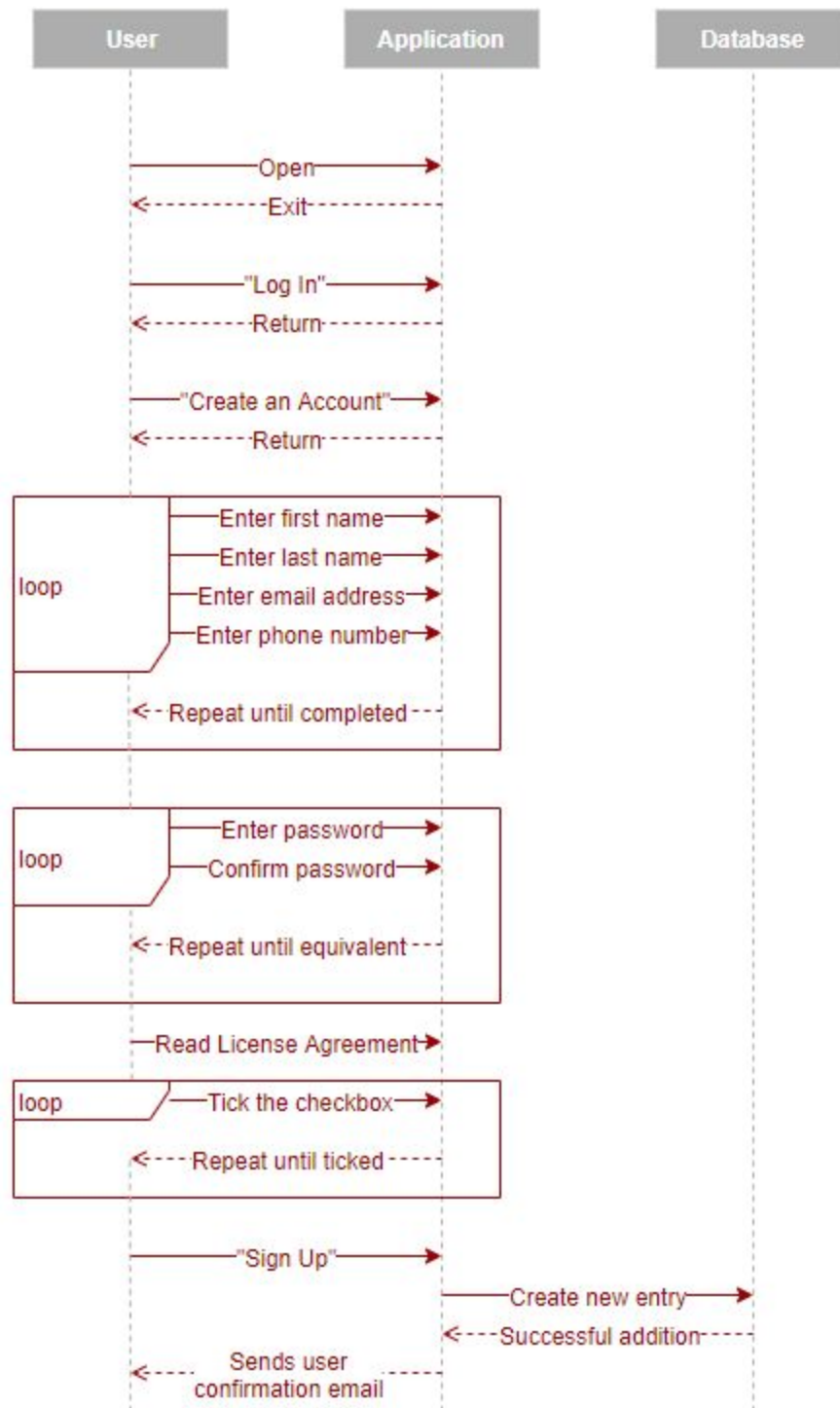


# Use Case State Charts:

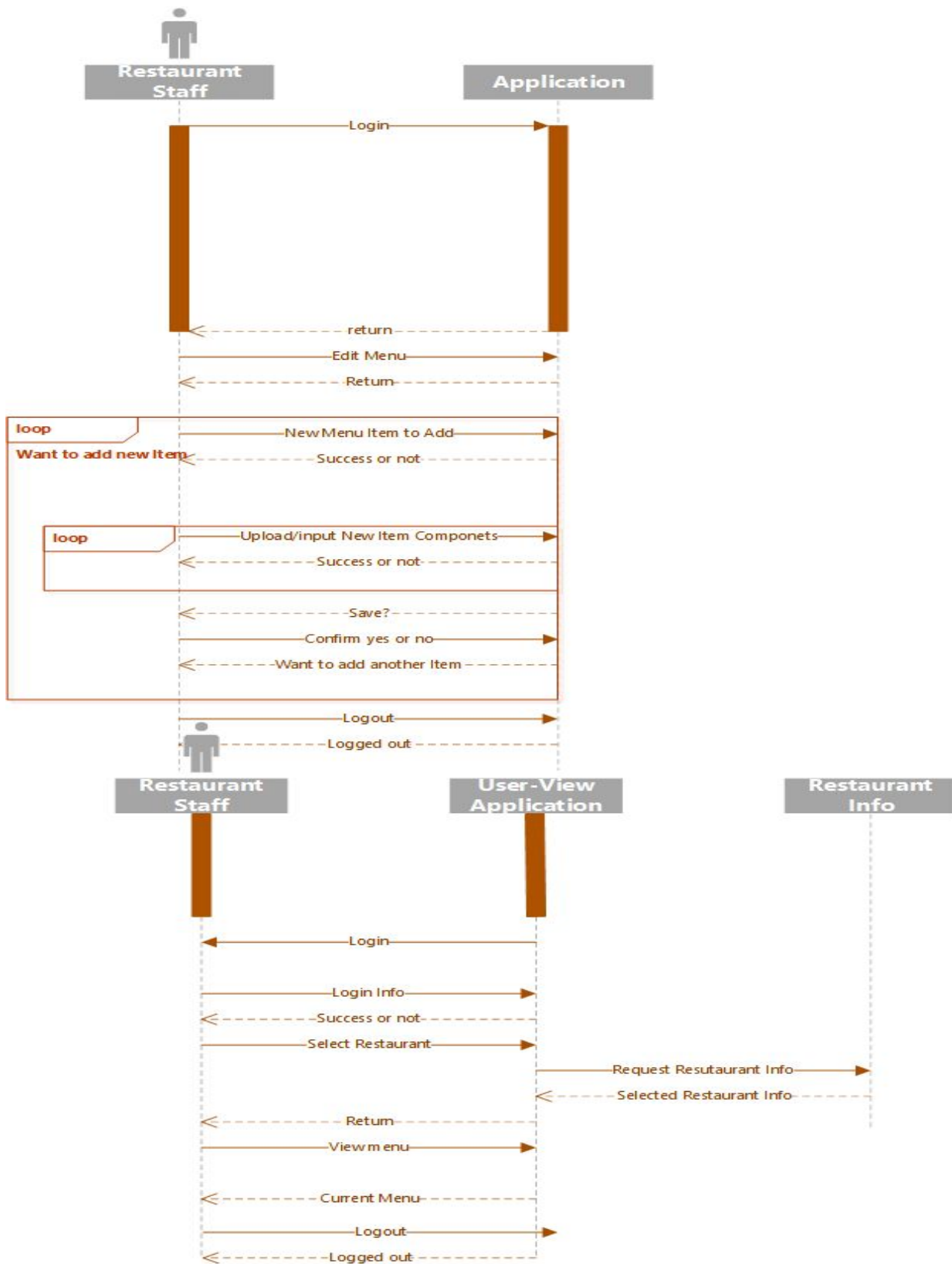
## Use Case One: Standard Customer Order



## Use Case Two: Account Creation





### Case Three - Luke Carter: Adding New Menu Items







## Change Journal:

- While not a requirements change, we decided to change our requirements list bullet format in this version of our development document. The idea is to allow us to more easily refer to a specific definition or specification based on a numbering format. This will be especially useful when we are having group and customer meetings. It saves the time spent with everyone searching for the exact requirement being referred to by description with an exact reference. This is especially useful since we work in a collaborative document format.
  - We updated our account creation requirements based on direct feedback from the project customer after reviewing the requirements and paper prototypes. The project customer stated that they preferred login information to be linked to an email as opposed to a username, in part because it's an extra step trying to remember an extra unique user name whereas most people already know their email by memory. We also added a valid phone number to the user account creation requirement based on the recommendation from the project customer. This addresses a perspective we hadn't considered where there is some need for the restaurant to contact the user directly, eg. the restaurant is out of a certain dish that was ordered. This change in particular showed us the value of getting direct feedback from the project customer viewing our prototypes as a user.
  - The customer also requested that we add an interface to coordinate the payment system between the user, app, and restaurant. This is probably the most unfamiliar technical area to our group, so the requirements specifications regarding payments systems are still relatively sparse, or black-boxed. Leaving the payment system abstract will however allow us to develop the system around it with a predictable input and output state and dataflow. One point of concern here is that leaving the payment system as a black box might not fit well in our current waterfall based development cycle. Being able to develop around this
- 



system and implement it with a concerted push would be very useful for our development cycle as it has proceeded so far.

- Based on our customer's original vision of being a GrubHub-like product, we decided to add functionality to our requirements to reflect a similarly flexible search capability. The search related requirement should reflect the need to search and/or sort for distance from user, rating, and cuisine type.
  - We added language to the requirements to clarify the fact that once an order has been placed only restaurant staff can edit the order, users can only add to an order. The impetus behind this decision was a discussion about the difficulties faced not necessarily by the system, but by the restaurant user, of having orders changed while they are in process up until the time of customer arrival. Frequent modifications up until arrival would make the app a point of conflict, which is the opposite of the project customer's vision and the development teams goals.
  - Drinks orders can only be fulfilled once ID has been verified in person at the restaurant. This requirement needed to be added as a legal obligation. We will likely need to include that in the restaurant's terms of service. It differs from typical practice where ID is checked prior to ordering alcoholic beverages. We should do further research into the legal requirements of this interface.
  - We added several abilities to the restaurant side application of the requirements. The restaurant should be able to set the maximum number of diners it will accept for an app reservation. We also added a requirement that the restaurant be able to set a limit on the time that reservations can be requested. This makes sense from the restaurant side as extremely large parties often require things like exceptional staffing. I would acknowledge our project customer and my wife here for providing extremely valuable insight and consideration of the silent user in our system, which is the restaurant staff. If we make the system amazing for customers but awful for restaurant staff, it will not succeed.
  - The requirements were also modified to reflect the restaurant's ability to modify an order based on the availability of items or customer feedback. This again
- 



refers to the agency of the restaurant as a user. At the time of marketing, we should emphasize these qualities as being conducive to restaurant-side adoption, i.e. we aren't building a system only for users, but also for vendors behind the scenes. This is where the app gets monetized and should get more attention in late stage development.

- The app will provide a new user tutorial that walks them through finding a restaurant, getting a table, placing an order, and payment. This will be presented to new users after their account creation and first login.





## ❖ Customer Input:

- The customer was able to meet with us Thursday evening for a productive meeting reviewing our paper prototypes and requirements, providing great feedback and insight into our design process.

## ❖ Team Contributions:

- All: Communication via private Slack channel and collaborative Google Doc.
  - Timothy Tseng - UML and Dataflow diagram review, revision, and updates, customer feedback review.
  - Miles McCoy - Paper prototypes, requirements revision.
  - Ibrahim Mahmoud - Paper prototypes.
  - Luke Carter - Requirements evaluation, requirements revision, revision documentation, document preparation.
  - Mathew McDade - Document setup, customer requirements and paper prototype feedback meeting, requirements revision, revision documentation, document preparation and submission.
- 