SW Engineering CSC 648/848 Fall 2015

Alloca-**table**

Group Number 7- Local

Stephen Tsay – stsay@mail.sfsu.edu

Gary Ng

David Karapetian

Anish Kumaramangalam

Casey Jones

Sai Undurthi

Khanh Le

"Milestone 2"

10/22/15

History Table	
Milestone 1 Feedback	Add pagination, Say a few words about company, Add use case for manager, add web admin, add restaurant to data dictionary, group specs, add specs for host and admin, revise non-function specs, shade column, specify two latest working versions for browser

Table of Contents

I. Executive Summary
II. Use Cases
III. Data Definitions/Summary
IV. List of functional specs
V. List of non-functional specs
VI. UI Mock-ups
VII. Competitive analysis
VIII. High-level system architecture with UML Diagrams
IX. Key Risks
X. Team Roles

I. Executive Summary

Alloca-table is a group of 7 CS students in CSC 648-848 at San Francisco State University. The name Alloca-table comes from the allocate function, used to save space in memory. Similar to allocate, Alloca-table will save a table for you at a restaurant of your choice!

Alloca-table is a web-based restaurant reservation management solution that serves to significantly decrease the stress and difficulty of managing reservations and tables. It is an all-in-one system that brings together every party involved in table reservations: the customers, the hosts, and restaurant management. This system contains everything a manager, host, and customer will need all in one package, completely removing the need for any intermediaries. For this system, there are 4 different kinds of users: **patrons**, **managers**, **Hosts**, and Alloca-table **administrators**. Each of these user types will have their own interface which will limit what the respective user has access to.

Users of Alloca-table will be able to make restaurant reservations with or without a registered account. The interface is visually appealing and simple to use, so people of all ages and skill levels can use it with ease. Restaurant management/IT will be able to register a restaurant and its information into our database in order to have it display in our search results. This interface is similar to the patrons' interface so that there won't need to be excess amounts of training in order to understand how to interact with the system. Restaurant hosts will have access to the submitted information of the patron, and their table reservation. They will be able to see the basic information of a non-registered user (Name, Phone number, etc.), as well as control reservations(i.e. add/edit/remove a reservation). As for a registered user, in addition to the functionalities for a non-registered user, the host will also be able to see a basic history of the patron's visits to that particular restaurant, such as previous tab amounts, preferred table(s), etc. The host interface is focused on function over form. It will be simple and intuitive and will have large buttons, so as to make use of it on a touchscreen device simpler. Alloca-table administrators will have full control over the entire system, allowing tasks such as approving, verifying, and managing restaurant applications, managing page content on restaurant and user profiles, and general maintenance of the application. This application uses a remote database to store restaurant information that is inputted by the restaurants themselves through our restaurant registration interface. Our search query results are populated by restaurants from this database.

II. Use Cases

Use Case 1 - **Guest Patron**

Tom is a middle-aged man living in San Francisco. He wants to eat at Freddy's Steakhouse for dinner tomorrow night with his wife. He goes to Alloca-table and searches for Freddy's Steakhouse. In the menu he selects 7:00, 2 people, and tomorrow's date. He does not have an account. The timeslot he wants is available, so he selects the timeslot, and confirms the reservation by giving Alloca-table his email address. Alloca-table sends Tom an email with the reservation details. The next day, Tom goes to Freddy's Steakhouse, gives the host his name, and the host seats Tom at 7:00.

Use Case 2 - **Registered Patron**

Jason is a registered patron user of Alloca-table. He lives in San Francisco and wants to have dinner tonight with his wife at Freddy's Steakhouse. He goes to Alloca-table and searches for Freddy's Steakhouse. In the menu he selects 7:00, 2 people, and tonight's date. Being a member of Alloca-table he just has to login and select the timeslot, and confirms the reservation. Alloca-table sends Jason an email with the reservation details. Jason goes to Freddy's Steakhouse, gives the host his name, and the host seats Jason at 7:00. Jason likes the ambience and food at Freddy's Steakhouse, so he decides to favorite the restaurant for later under his account on Alloca-table. He also rates the restaurant 5/5 stars and leaves a review on the restaurant page.

Use Case 3 - **Registered Patron**

Beth is a registered patron user of Alloca-table. She is visiting San Francisco with her girlfriend and they are hungry for chicken gumbo. She remembers she saw a Food Network soul food special on Brenda's Kitchen. Using Alloca-table, she searches for Brenda's Kitchen at 7, her desired dining time. Unfortunately, Brenda's Kitchen doesn't have any timeslots available at 7. However, because she is a registered patron user of Alloca-table, Alloca-table also returns a list of restaurants that also serve soul food and are available at 7. She chooses Fried Chicken Frank's off of that list and makes the reservation there instead.

Use Case 4 - **Host**

Tyson is Freddy's Steakhouse Host. When Jason books his table on Alloca-table, Tyson gets an update on his account that the respective tables have been booked on a particular date for a particular time. When Jason shows up at the restaurant, Tyson verifies the reservation for him and seats him at the reserved table.

Use Case 5 – **Restaurant Manager**

Allison is Freddy's Steakhouse Manager. At the end of each day, she goes to Freddy's Steakhouses webpage on Alloca-table and checks for new reviews and pictures. She reads the reviews. Then she looks at the pictures that have been uploaded for approval. She accepts several good pictures, and rejects the inappropriate ones. The accepted ones will go onto the Alloca-table picture database for Freddy's Steakhouse.

Use Case 6 - **Web Administrator**

Chris is a Web Administrator for Alloca-table. A new restaurant Tom's Thai Cuisine has contacted Alloca-table and wishes to join the service. Chris is given Tom's Thai Cuisine's information (store hours, cuisine type, restaurant location, pictures) from Alloca-table's sales and marketing team and adds it to the Alloca-table database. Once he is done, Tom's Thai Cuisine is searchable and reservable.

III. Data Definitions/Summary

1. **Alloca-table:** Alloca-table is a group of 7 CS students in CSC 648-848 at San Francisco State University. The name Alloca-table comes from the allocate function, used to save space in memory. Similar to allocate, Alloca-table will save a table for you at a restaurant of your choice!
2. **Guest Patron**: A patron can search for and reserve tables with us, just like a guest but once they attempt to make a reservation they will be asked for, at minimum, their email address so that they may receive a receipt for their reservation.
3. **Registered Patron**: A Registered Patron can do everything that a patron can however the registered patron has gone a step further and created an account on our website. This account has a name, and picture. Allows them to leave comments, and possibly even accumulate points when they make reservations.
4. **Patron:** The term used when the user is either Guest or Registered.
5. **Host**: Clients with whom Alloca-table has tables available to reserve on their website. Hosts will have full control over what information is made available on our website. Such as menu, hours, floor plan, number of tables, expected wait time etc.
6. **Manager**: The manager can make changes to the restaurant's data page. The manager can also approve pictures that users have submitted.
7. **Web Admin**: The web administrator is an employee of Alloca-table. His function is to help the restaurant's employees set-up their account at Alloca-table. He can also remove reviews that are flagged to be not true.

IV. List of functional specs

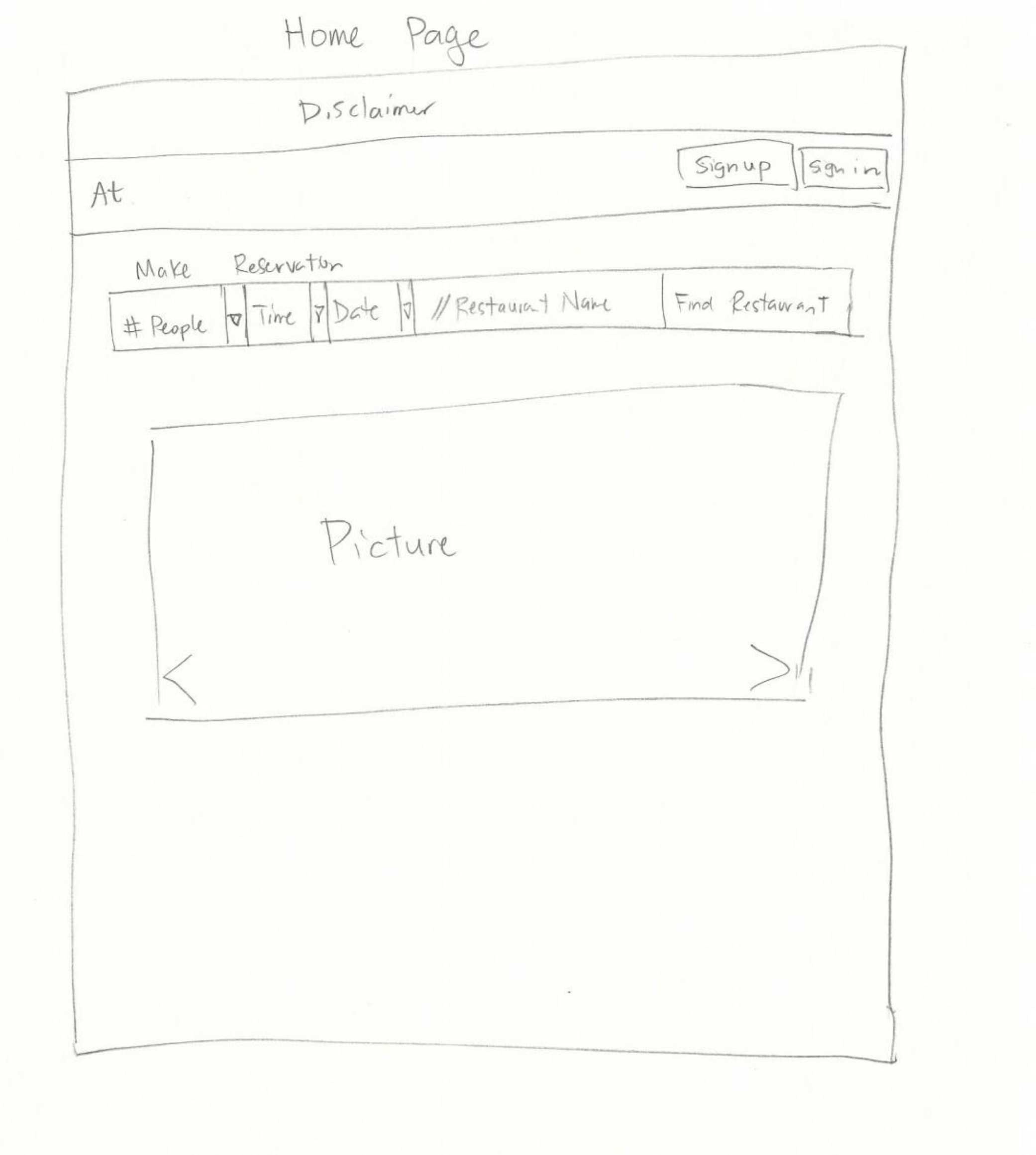
Group		Priority
1. User	1. Guest Patron shall be able to search by food category 2. Guest Patron shall be able to search by restaurant name 3. Guest Patron shall be able to specify number of people in party 4. Guest Patron shall be able to specify time of reservation 5. Guest Patron shall be able to specify date of reservation 6. Guest Patron shall be able to make a reservation with an email address 7. Guest Patron shall receive an email from the program once the reservation has been made 8. Guest Patron shall be able to upgrade to a Registered Patron. 9. Registered Patron shall have all the rights of the Guest Patron 10. Registered Patron shall receive incentives 11. Registered Patron shall be able to leave reviews for restaurants 12. Registered Patron shall be able to upload a profile picture Registered Patron shall be able to upload pictures of their experiences at a restaurant 13. Registered Patrons shall be able to save his or her information onto the Alloca-table database 14. Registered Patron shall be able to view other Restaurants of the same cuisine in the same searched timeslot if the currently searched restaurant timeslot is unavailable. 15. Registered Patrons shall be enable text alerts for last minute reservations	1 1 1 1 1 1 2 2 2 2 3 2 2 2 3 3
2. Host	1. Host shall have a list of names of people that have made reservations for that day 2. Host shall be able to mark which people have shown up for their reservation	1 1
3. Manager	1. Manager shall have all the rights as the host 2. Manager shall be able to edit the store hours, description,	2 2

	and location 3. Manager shall be able to approve pictures added by Registered Patrons	3
4. Web Administrator	1. Web Administrator shall be able to add restaurants to the database 2. Web Administrator shall be able to remove reviews 3. Web Administrator shall have all the rights of the Manager	1 1 1
5. Restaurant	1. Potential restaurant matches will be listed in a drop down menu while the patron is typing 2. The food category search shall return a list of restaurants that have that type of food 3. The restaurant category search shall return the restaurant's data page 4. If the restaurant is not in the database, the restaurant category search shall return a list of similar sounding restaurants 5. The restaurant data page shall list whether the selected time is available 6. The restaurant data page shall list other times if the selected time is not available 7. The restaurant data page shall list other restaurants of the same cuisine if the same time slot is available 8. The restaurant data page shall contain a brief description of the restaurant 9. The restaurant data page shall have a map of the surrounding location 10. The restaurant shall be able to list what times are available 11. The restaurant shall be able to list how many tables are available at each time slot.	3 1 1 3 1 2 3 1 1 1 1
6. Alloca-table	1. The program shall provide real-time feedback on restaurant availability	1

V. List of non-functional specs

1. Application shall be developed using class provided LAMP stack
2. Application shall be developed using pre-approved set of SW development and collaborative tools provided in the class.
3. Application shall be hosted and deployed on Amazon Web Services as specified in the class
4. Application shall be viewable in standard desktop/laptop/mobile browsers, and shall render correctly on the two latest versions, as of this writing, of the following major browsers:
    a. Mozilla Firefox on Windows 10 and OSX 10.11.1
        i. 41.0.2
        ii. 40.0.3
    b. Google Chrome on Windows 10 and OSX 10.11.1
        i. 46.0.2490.80
        ii. 45.0.2454.104
5. Data shall be stored in the database on the class server in the team's account
6. Application shall be served from the team's account
7. No more than 50 concurrent users shall be accessing the application at any time
8. Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.
9. The language used shall be English.
10. Application shall be very easy to use and intuitive. No prior training shall be required to use the website.
11. Google analytics shall be added for major site functions.
12. The website shall prominently display the following text on all pages "SFSU/ Software Engineering Project For Demonstration Only". (Important so as to not confuse this with a real application).
13. Modern SE processes and practices must be used as specified in the class, including collaborative and continuous SW development, using the tools approved by instructors

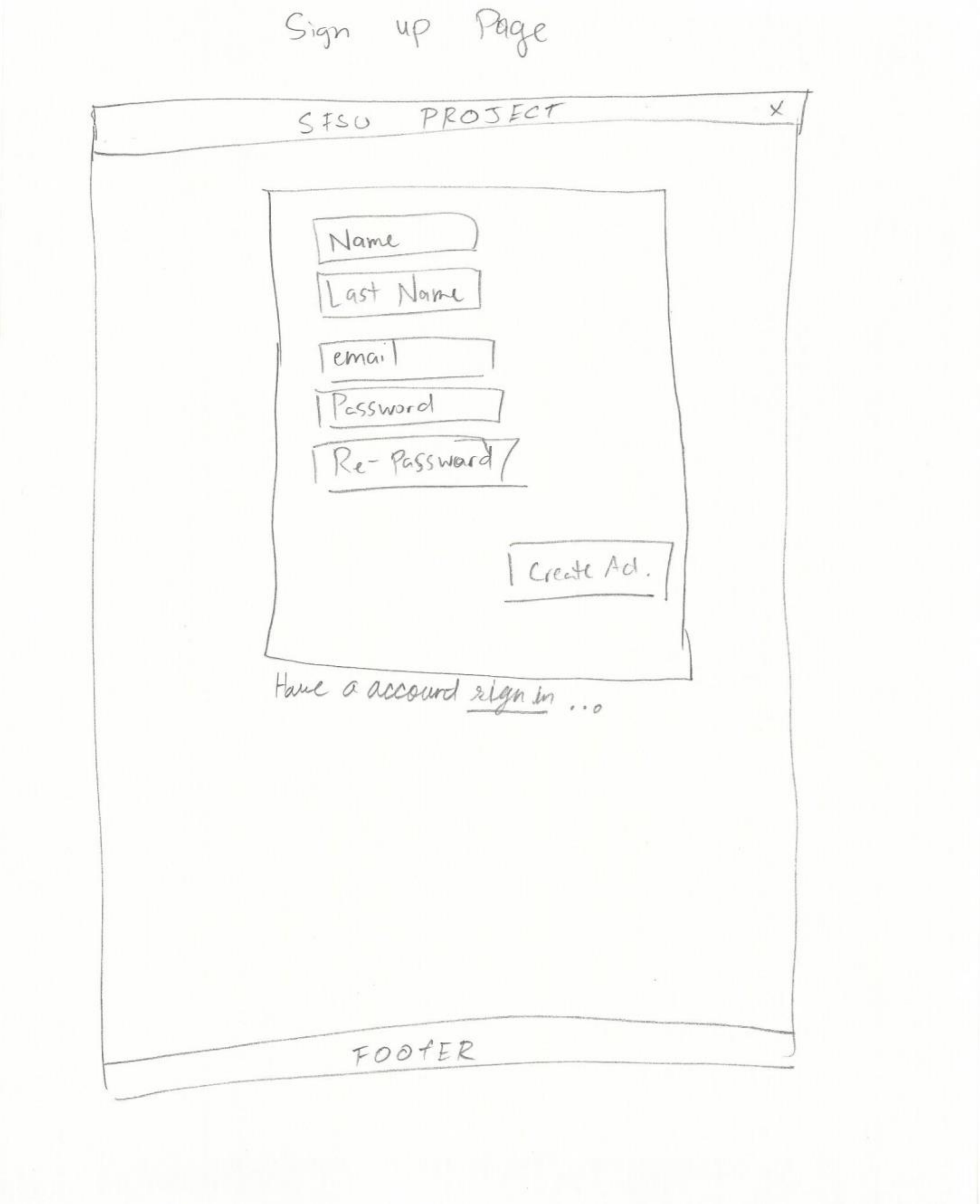
VI. Mock-ups and Storyboard



Page 1

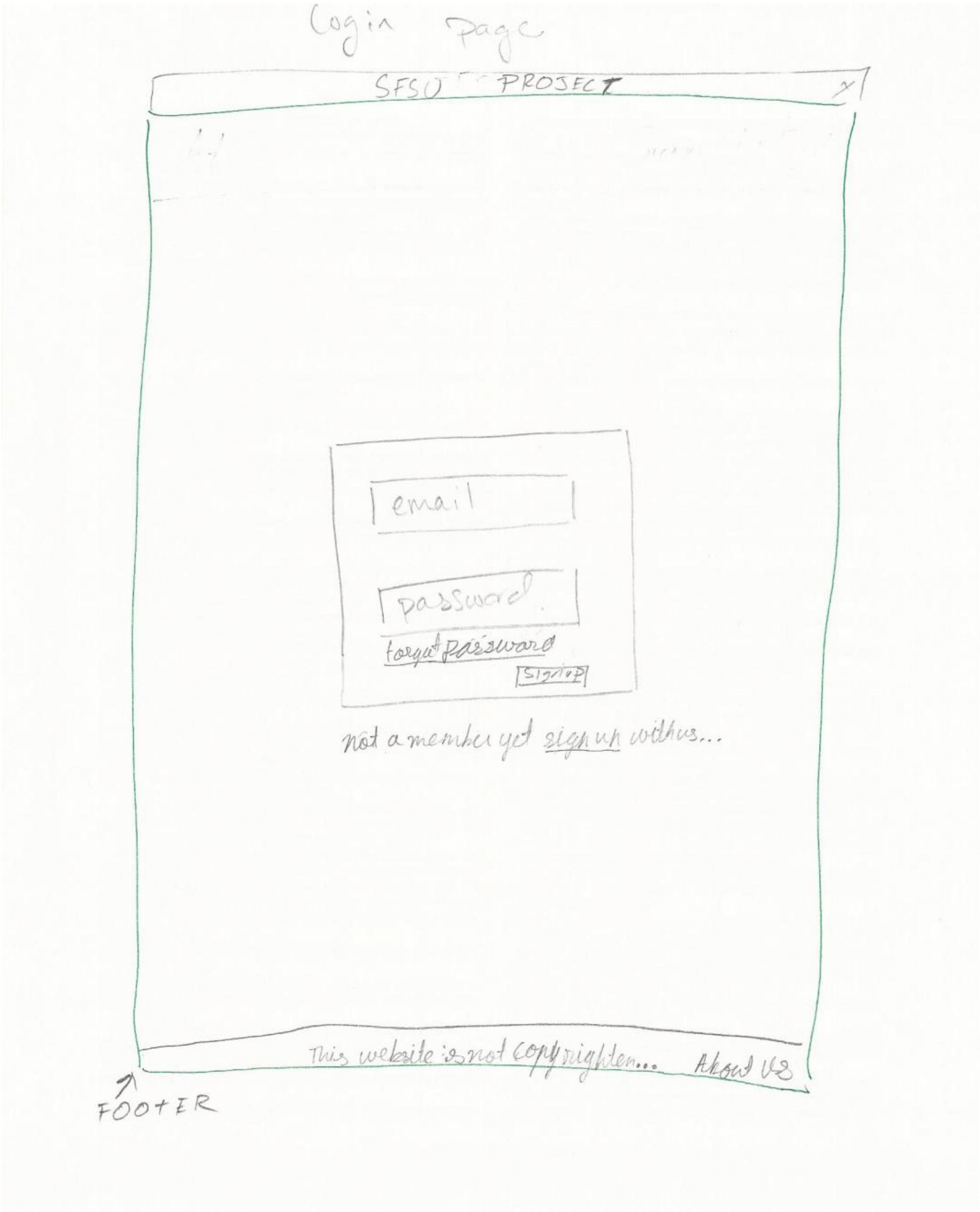
Home page for Alloca-table. The "Restaurant Name" area is grayed out. After hitting the Find Restaurant, the **patron** will be directed to one of several results depending on if his restaurant search was successful(Page 4) or not, and whether the desired time is available (Page 4) or not(Page 5, 6).

The **patron** can also sign up(Page 2) or log in(Page 3).



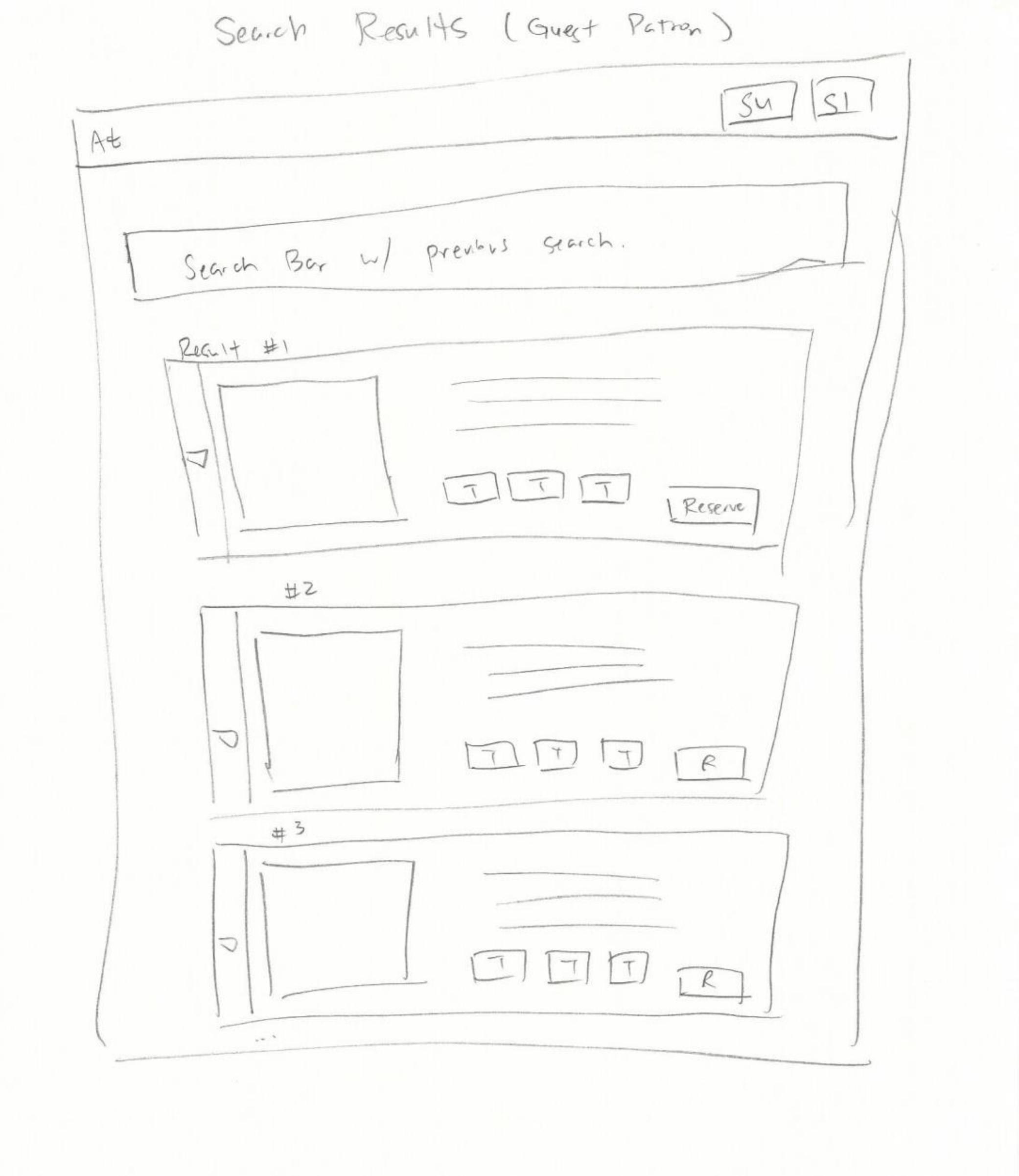
Page 2

Guest Patrons can create an account from this page. Once the account is created, they will be redirected to the homepage(Page 1) automatically.


Login Page
SFSU PROJECT
x
email
password
forget password
SIGNUP
not a member yet signup with us...
This website is not copyrighten... About US
FOOTER

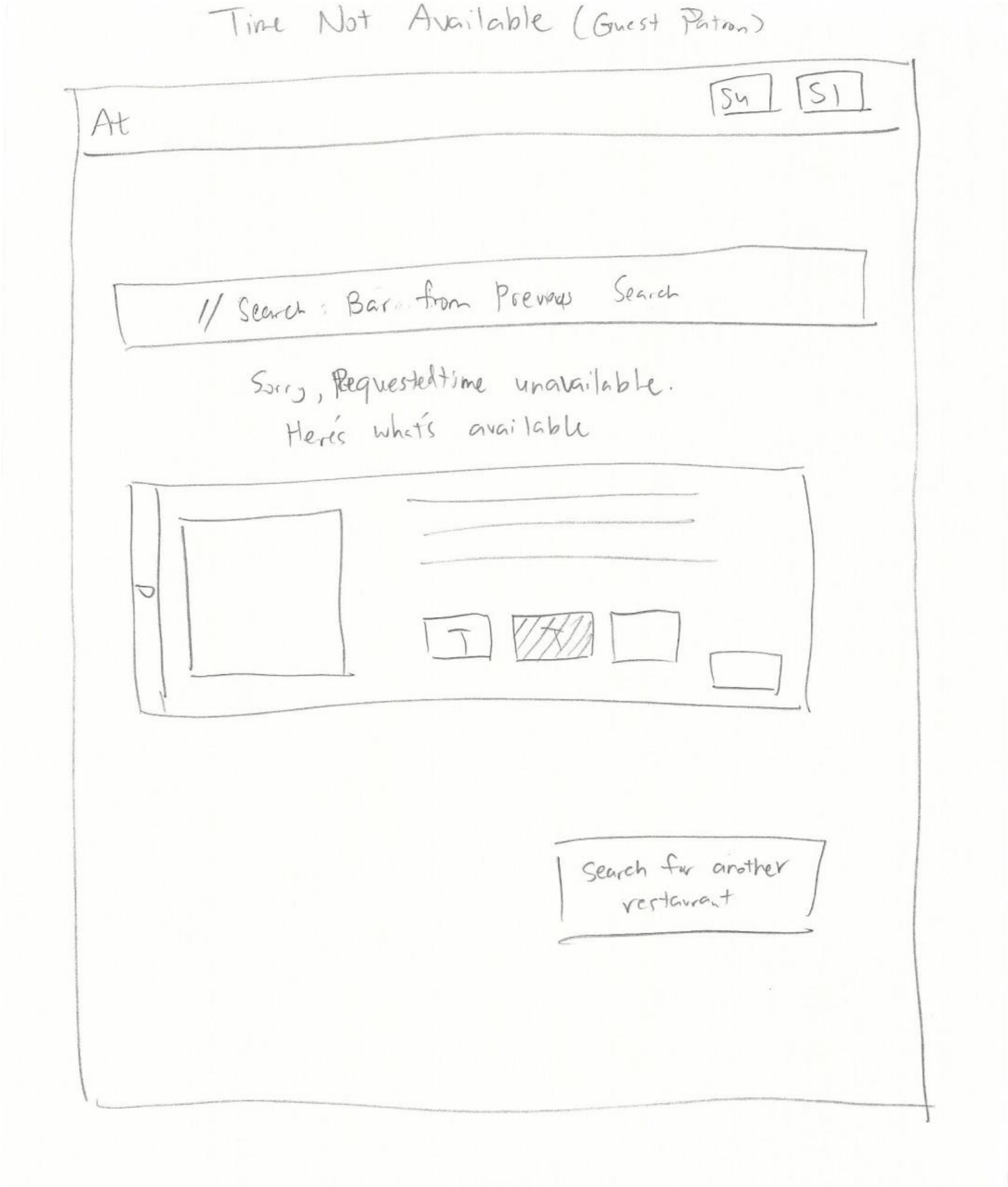
Page 3

Registered Patrons can log in from here. The log in page is a small pop-up box.



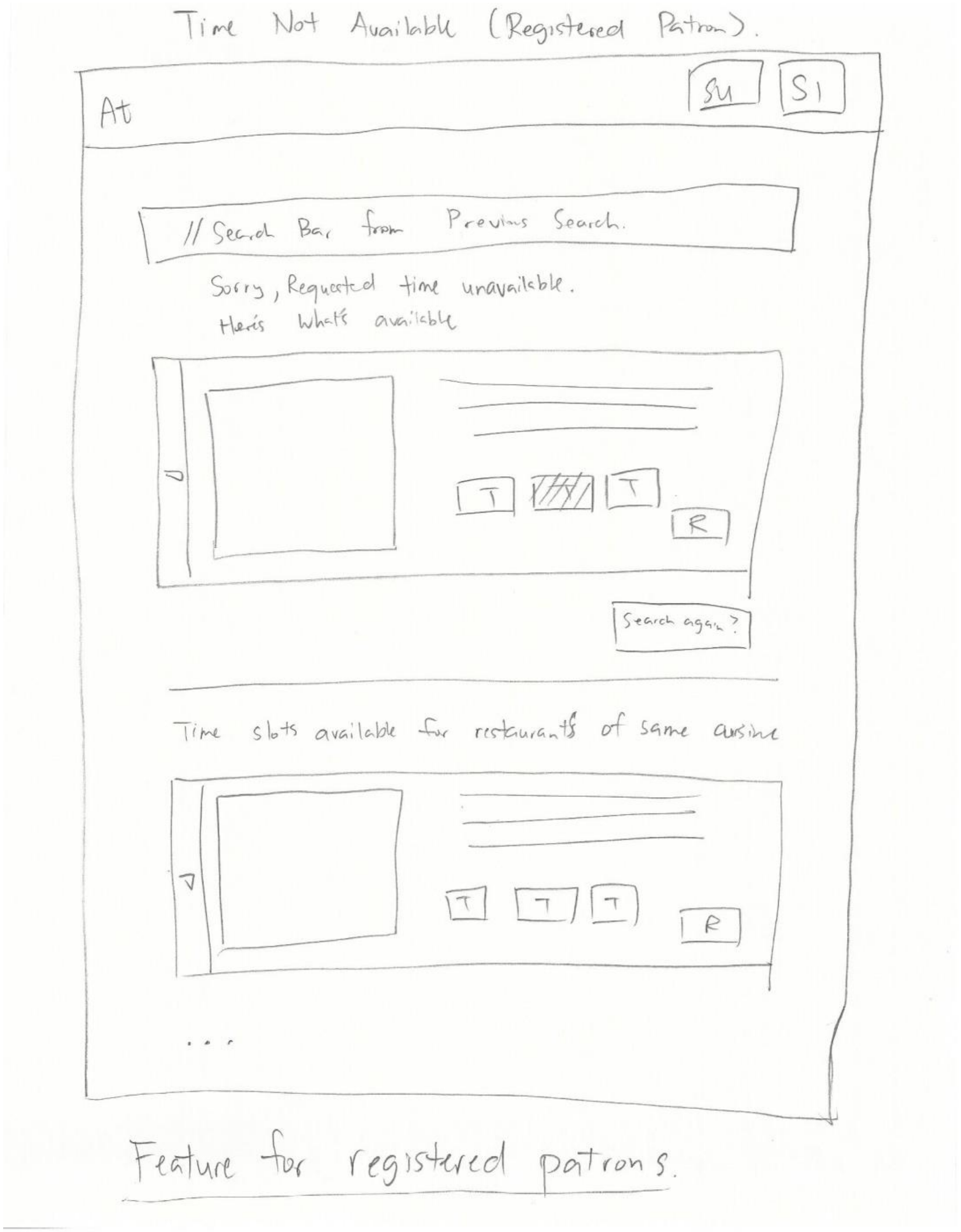
Page 4

This is the result for a successful search for **patrons**. The indicated time will be indented, but other available times will be shown in case the **patron** changes his or her mind. The **patron** can reserve right away by clicking the reserve button or hit the drop down arrow and go to the restaurant information page(Page 8).


Time Not Available (Guest Patron)
Su
SI
At
// Search : Bar from Prevous Search
Sorry, Requestedtime unavailable.
Here's what's available
Search for another restaurant

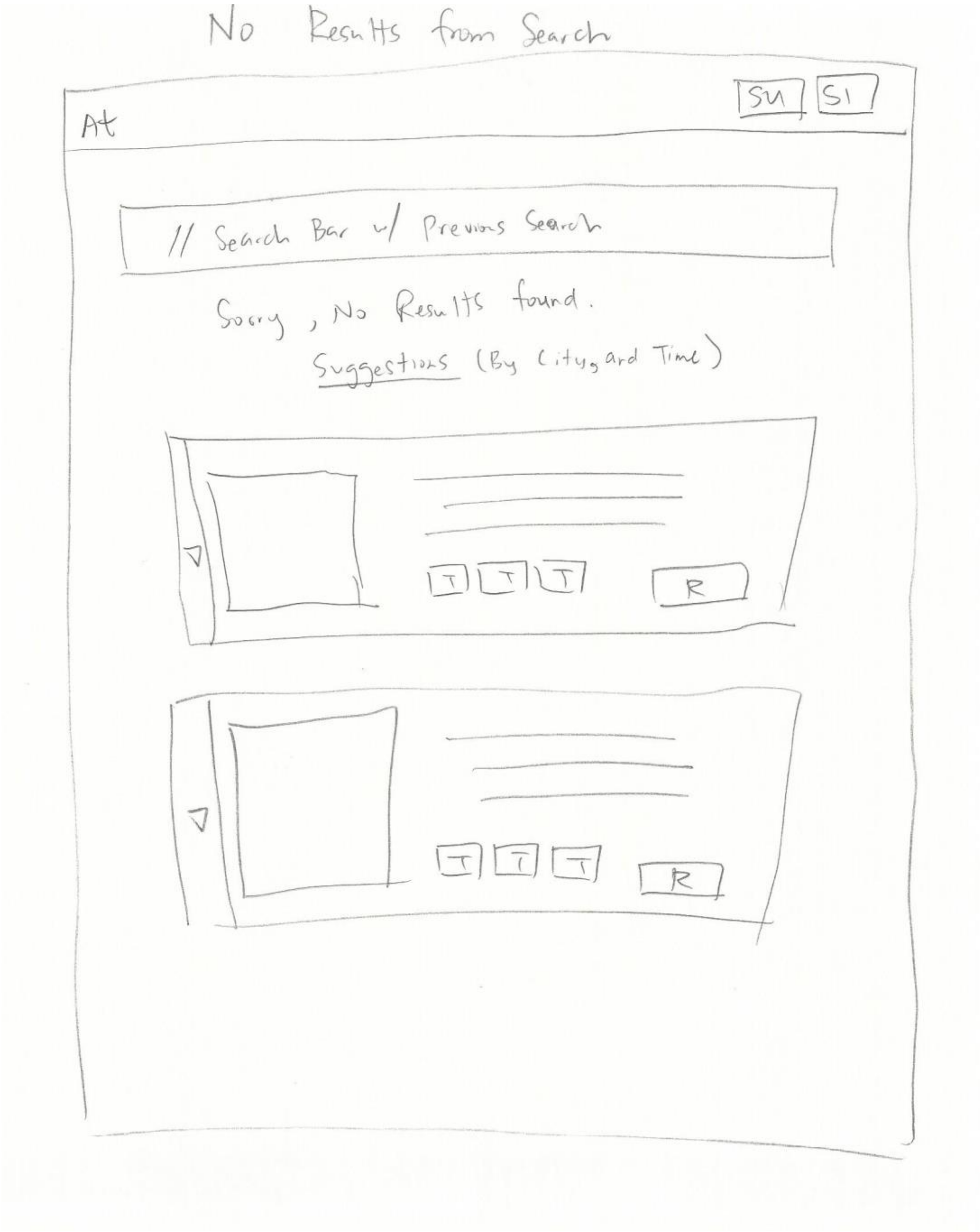
Page 5

If the time is unavailable, the **guest patron** will see what the current times are available. The **guest patron** can then reserve from this screen and go to the reservation page(Page 9) or return to the homepage (Page 1) and search for another restaurant. Another search can also be performed from this screen.


Time Not Available (Registered Patron).
At
SU
SI
// Search Bar from Previous Search.
Sorry, Requested time unavailable.
Here's What's available
T
T
R
Search again?
Time slots available for restaurants of same cuisine
T
T
T
R
. . .
Feature for registered patrons.

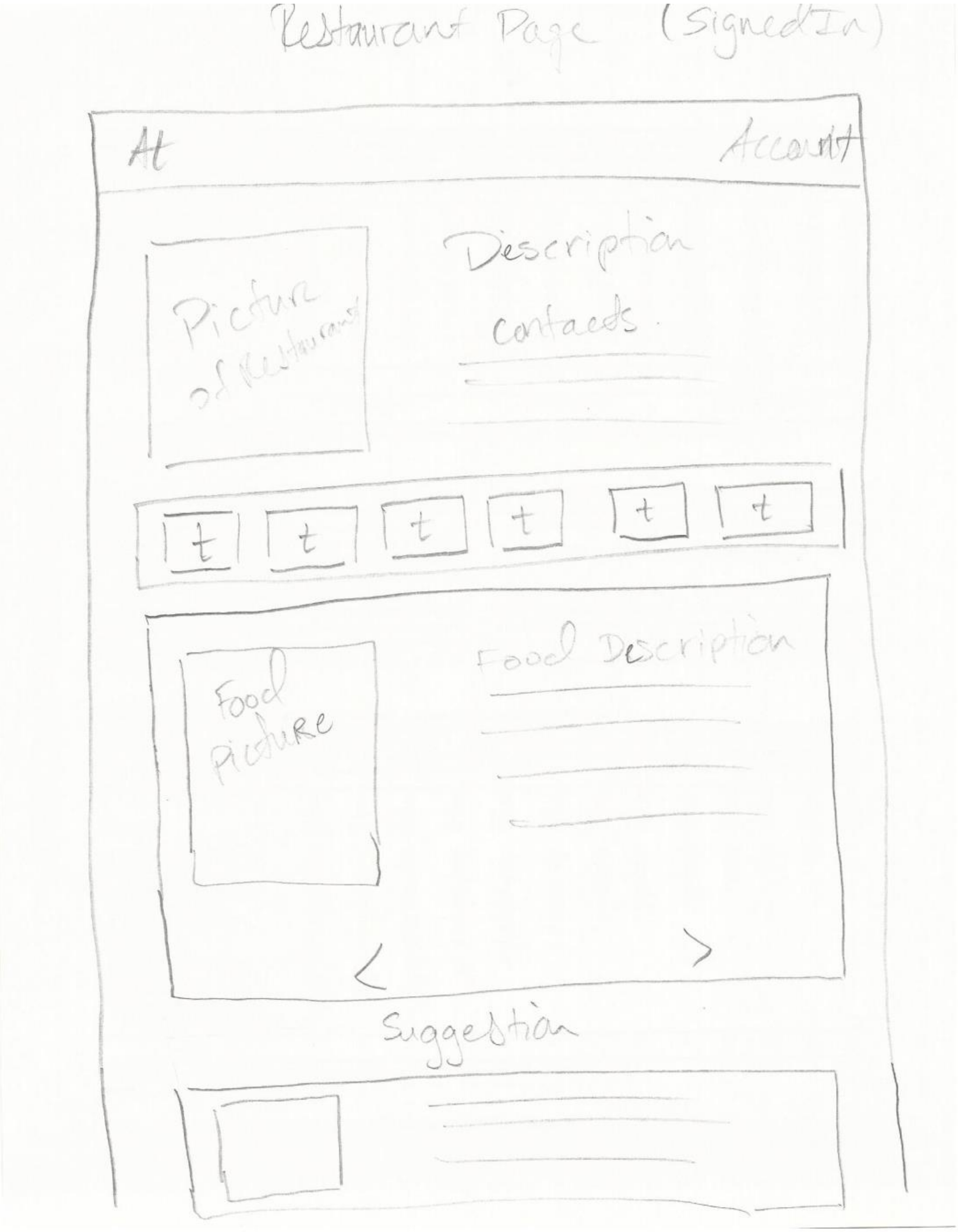
Page 6

If the time is unavailable for **registered patrons**, in addition to seeing the other times for the searched restaurants, they will also see restaurants of the same cuisine that do have the time available.


No Results from Search
At
SU
SI
// Search Bar w/ Previous Search
Sorry, No Results found.
Suggestions (By City, and Time)
T
T
T
R
T
T
T
R

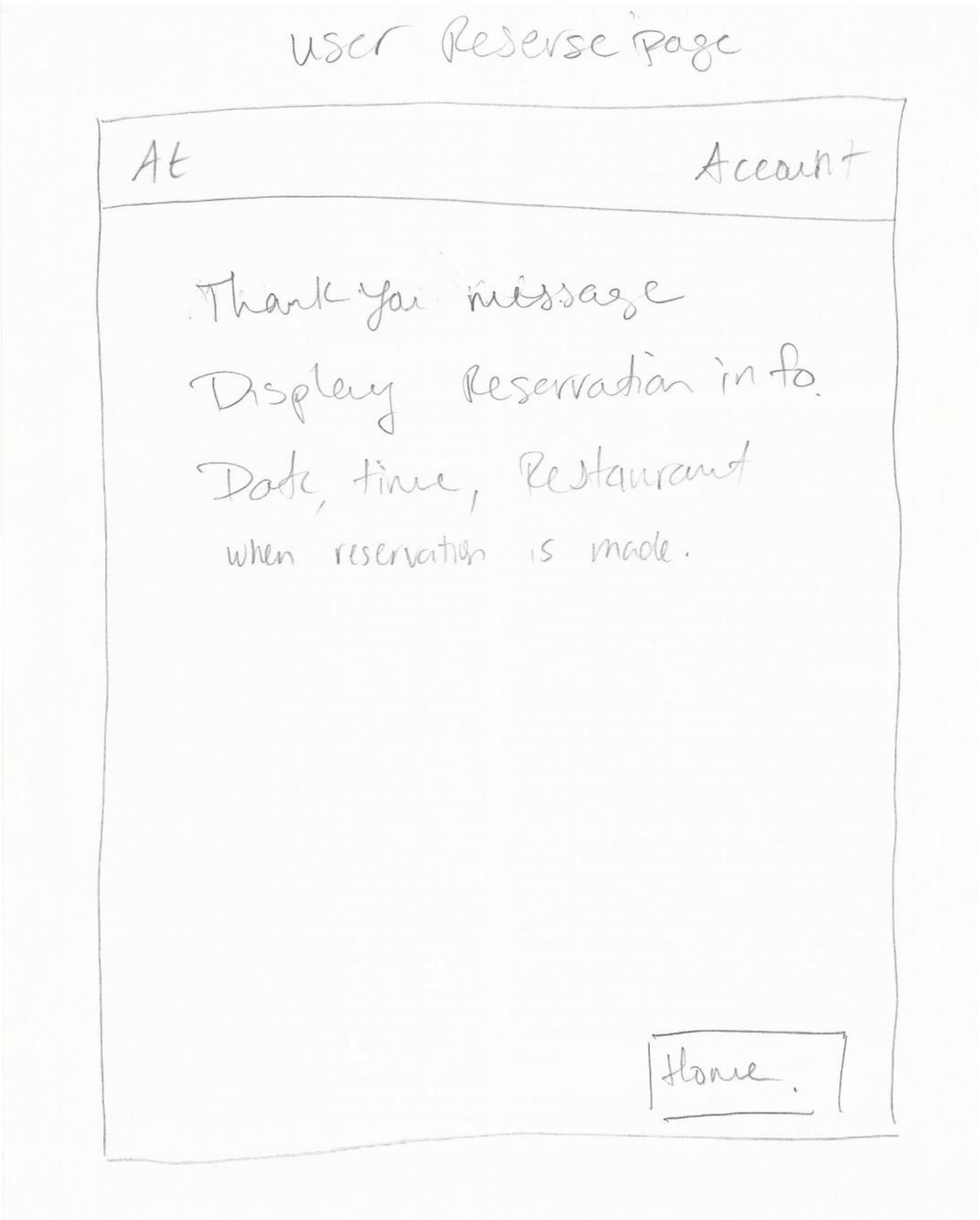
Page 7

If the search for the restaurant name is unavailable, suggestions will be displayed by the city and time slots available.



Page 8

Restaurant information page has information about the restaurant, as well as an expanded time selection. **Patrons** can also make their reservation from this page and go to the reservation confirmation page(Page 9).



Page 9

Reservation Confirmation page shows all the details of the reservation that the user just made.

VII. Competitive analysis

Features	Open Table	Restaurant Reservations	Zomato	**Alloca-table**
Search bar (Name, Cuisine)	++	+	++	++
Can pick from multiple time slots	+	-	+	+
Shows time slots for restaurants of the same cuisine if current restaurant unavailable	-	-	-	++
Restaurant information page	++	+	+	+
Last minutes reservations for elite users	-	-	-	+
Reviews from elite users	+	-	+	+
Gorgeous UI	+	-	++	++

Our main competitor is Open Table. Therefore we will aim to add 2 functions on our website that we believe will separate us from them. The first is that we will show restaurants of similar cuisine with available time slots if the desired time slot at the desired restaurant is taken. Although this may seem like it takes away business from one business, in the long run, both restaurants will benefit from increased customers. The second feature we wish to implement is for our **registered patrons**. We will allow them to opt into a system that informs them of last minute reservation slots for nice restaurants. By adding these two functions, and maintaining a baseline of regular table reservation functionality, we plan to make a system that is more user friendly to eaters, which will increase the amount of people eating out, increasing business for all restaurants.

VIII. High-level system architecture

This following section will describe the tools, frameworks, and APIs being used in assisting our development of this project and how they will tie into the different components that brings this application together.

LAMP

Our application will be developed following the traditional LAMP model environment that consists of the following:

- Linux
  - We are deploying our application on the Ubuntu Linux platform.
  - Hosted on Amazon Web Services (AWS), the cloud-computing platform.
- Apache
  - Our application will be served through the Apache web server to allow users to access our application over the web.
- MySQL
  - The database backend will be managed with MySQL to store information such as users, restaurants, schedules and more.
  - Allows us to easily retrieve any data using our own search algorithms and display it when needed.
- PHP
  - The PHP scripting language allows us to create a very dynamic and content-driven web application.
  - Code will be hidden on the server-side, which is perfect for backend development.

Frameworks and APIs

As for the front-end of our application, we'll be utilizing the following frameworks to assist in our development:

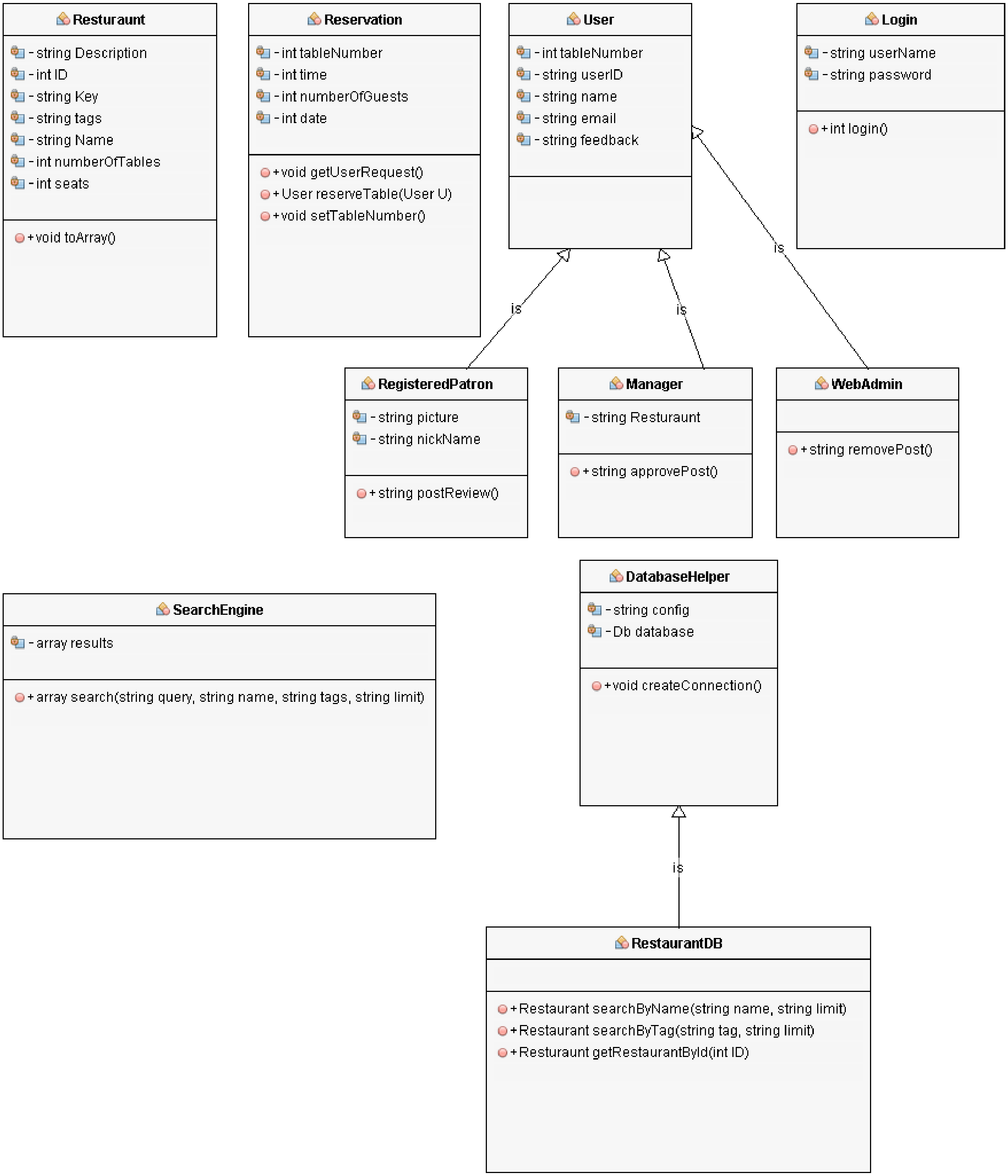
- Bootstrap
    - A web development framework that allows us to get started quickly by providing a series of templates and themes.
    - Provides a collection of custom generic components such as navigation bars, carousels, and more.
    - Supports responsive web design to allow us to easily design for different screen sizes.
- jQuery
    - A JavaScript library that eases cross-browser development using it's own API.
    - Speeds up development by providing wrappers to many existing functions.
    - Enables us to write cleaner JavaScript code by having to write less.
- Google Maps API
    - Provides us the ability to integrate Google Maps into our application.
    - Allows us to design our application around a map view for displaying nearby restaurants or just the restaurant itself.

Classes of Major Components

The following classes described below will be providing the core functions for our application:

1. Restaurant Class
    a. Encapsulates restaurant information retrieved from database.
    b. Data stored here will be used by restaurant pages to display relevant information to the user.
2. Reservation Class
    a. Provides an API to handle new and existing reservations made by the guests.
    b. Assesses the reservation parameters given by the user to see if such reservation can be made.
    c. The host can check the availability (time slot) of the tables that were booked.
3. Login Class
    a. The login class is used to authenticate a user's username and password combination.
4. User Class
    a. Contains information about the user like his/her user id, name, address, table no. etc.
    b. This class will also have a field for the user's feedback.

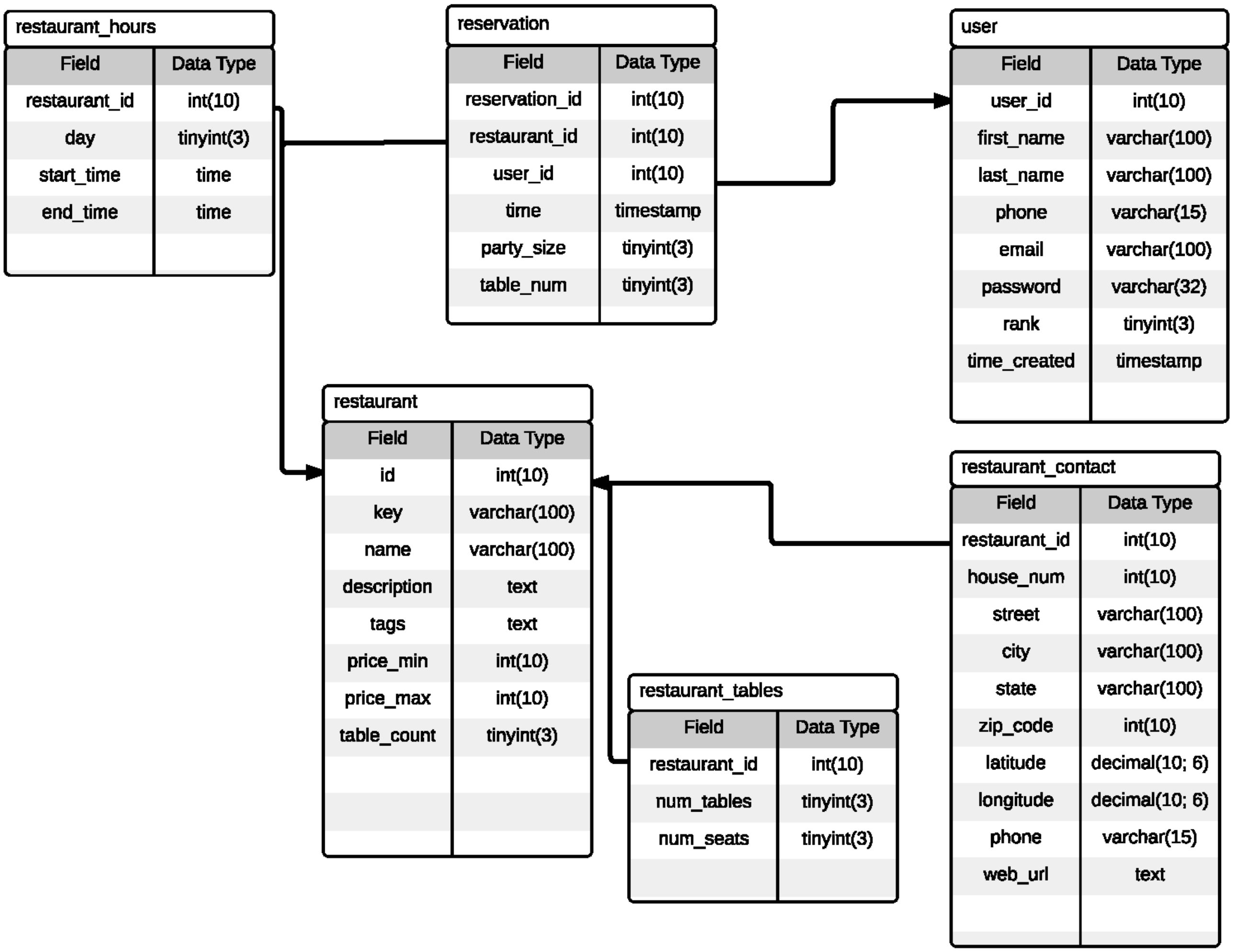
5. Database Helper Class
    a. Enables connection to the database to perform queries.
6. Restaurant DB Class
    a. Provides an API to retrieve restaurant information such as name, location, etc. from database.
    b. Can also be used to insert restaurants into the database.
7. User DB Class
    a. Provides an API to handle new and existing user information from database.
8. Search Engine Class
    a. Provides an API to retrieve restaurant suggestions from the database by using the name or types of food being served.
    b. Ties directly with the search box found in the website.


Resturaunt
- string Description
- int ID
- string Key
- string tags
- string Name
- int numberOfTables
- int seats
+ void toArray()
Reservation
- int tableNumber
- int time
- int numberOfGuests
- int date
+ void getUserRequest()
+ User reserveTable(User U)
+ void setTableNumber()
User
- int tableNumber
- string userID
- string name
- string email
- string feedback
Login
- string userName
- string password
+ int login()
is
is
is
RegisteredPatron
- string picture
- string nickName
+ string postReview()
Manager
- string Resturaunt
+ string approvePost()
WebAdmin
+ string removePost()
SearchEngine
- array results
+ array search(string query, string name, string tags, string limit)
DatabaseHelper
- string config
- Db database
+ void createConnection()
is
RestaurantDB
+ Restaurant searchByName(string name, string limit)
+ Restaurant searchByTag(string tag, string limit)
+ Resturaunt getRestaurantById(int ID)

Database Schema

This section will describe in high level how data will be organized using the database.

1. Restaurant Table
    a. Contains basic information about the restaurant such as name, description, table counts, etc.
    b. Restaurant managers will be responsible for creating a restaurant entry of this information for guests to view on the website.
2. Restaurant Contact Table
    a. Contains contact information about the restaurant such as location, phone, website URL, which is stored separately from the main restaurant table.
    b. Generated by restaurant managers for guests to view on the website.
    c. Guests will use the information to reach the restaurant if needed.
3. Restaurant Hours Table
    a. Contains restaurant hours range for every day of the week
4. Restaurant Tables Table
    a. Gives an idea of how many tables of how many seats for each restaurant to create reservations.
5. Reservation Table
    a. Store reservations for every restaurant with data such as time, date, party size, guest name, and restaurant key that references the Restaurant table.
    b. Data in this table is populated by guests creating reservations using the website.
    c. These reservations will be displayed by both restaurant host and the guest that made the reservation.
6. User Table
    a. Contains basic information about the type of user logging into the website such as registered patron. It also contains the user ID.
    b. Registered Patron will have an user id, email, username, registration date and the table no. for the restaurant manager to create.



restaurant_hours

Field	Data Type
restaurant_id	int(10)
day	tinyint(3)
start_time	time
end_time	time

reservation

Field	Data Type
reservation_id	int(10)
restaurant_id	int(10)
user_id	int(10)
time	timestamp
party_size	tinyint(3)
table_num	tinyint(3)

user

Field	Data Type
user_id	int(10)
first_name	varchar(100)
last_name	varchar(100)
phone	varchar(15)
email	varchar(100)
password	varchar(32)
rank	tinyint(3)
time_created	timestamp

restaurant

Field	Data Type
id	int(10)
key	varchar(100)
name	varchar(100)
description	text
tags	text
price_min	int(10)
price_max	int(10)
table_count	tinyint(3)

restaurant_contact

Field	Data Type
restaurant_id	int(10)
house_num	int(10)
street	varchar(100)
city	varchar(100)
state	varchar(100)
zip_code	int(10)
latitude	decimal(10; 6)
longitude	decimal(10; 6)
phone	varchar(15)
web_url	text

restaurant_tables

Field	Data Type
restaurant_id	int(10)
num_tables	tinyint(3)
num_seats	tinyint(3)

Media Organization

This following section will briefly describe how we'll be organizing media content specifically for restaurants.

- Images and other media will be stored in the file system where they will reside in their respective restaurant directories.
- Images will be stored in JPEG format as well as any thumbnails generated from it to keep files smaller for both storage and bandwidth purposes.

Search Architecture

The following will be describing certain aspects of what users can search as well as how the search algorithm will work.

- Users will be able to search using restaurant names as well as types of food the restaurants may serve.
- Searches will be queried against the database using a combination of LIKE and SOUNDEX to return suggestions back to the user. Working in combination with an AJAX search input system, users will get instant feedback with suggestions as they type character after character. This creates a guided search for the user and places the user in a controlled environment.
- Search will specifically cross reference table fields such as restaurant name and food categories.

Other Algorithms

This section will be describing any additional algorithms being used in our application.

- Reservation Algorithm
    - In general we will have a method through which we will try to place the request at their desired time slot without having too many unused seats. This model will try to fit a given pending request at a table that will result in the fewest unused seats. If two table assignments result in the same number of unused seats, the restaurant will offer the customer the time-slot that is closest to their desired time-slot.

IX. Key Risks

1. **Skills Risk-** We decided not to use the blob because no one on our group knew how to use them. We decided to use the frameworks that we were familiar with: Bootstrap, CSS, jQuery.
2. **Schedule Risks -** We have our meetings on Thursday from 5-7 every week. It's the only time of the week that we are all able to make it. The team is committed to keeping the project simple. Our main priority is to be able to complete the website. Given our realistic goals, it's very possible for us to complete our project. Once that priority is reached, we will start on our special features.
3. **Technical Risks-** We're still trying to figure out how to implement an auto-complete drop down menu for the search bar. This is being achieved through cooperative research.
4. **Teamwork Risks -** Since we have a large team, our discussions can become heated when everyone wants to share their opinion. Also, since we have more people, there isn't necessarily a job for everyone, sometimes we've had to double up on jobs (which is okay).
5. **Legal/content Risks-** We're creating our own restaurants so there won't be any legal risks in term of copying restaurant names. We'll be poaching pictures off of Google and Yelp, which should be fine given their terms of use.

X. Team

Name	Role
Stephen Tsay – stsay@mail.sfsu.edu	Team lead
Gary Ng	Tech lead, Database
Casey Jones	Front End Design
David Karapetian	Front End Design
Anish Kumaramangalam	Front End Design
Khanh Le	Front End Design
Sai Undurthi	Back End