SW Engineering CSC648/848 Fall 2015

BiteLine

Milestone 2

Team 5 (Local)

George Jone (jone@mail.sfsu.edu) SFSU, Steve Pedersen SFSU, Jacob Abba SFSU, Abdullah Alnamlah SFSU, Darin Vergara SFSU, Puyang Yu SFSU

October 22, 2015

History Table

V1	**Initial Doc**	**September 30, 2015**
V2	**After DP feedback**	**October 9, 2015**
V3	**Milestone 2 Draft**	**October 22, 2015**
V4	**After DP feedback**	**November 2, 2015**

1. Use Cases

Customer: After both getting home from work, Brett and his wife Björk, love to eat out at local restaurants in their neighborhood in San Francisco, since it is more often that not too much of a hassle to cook their own meals so often. They decide to use BiteLine in order to be more economical about their dining options because BiteLine offers real-time promotions and discounts at restaurants that are trying to boost their business. Additionally, instead of calling several different restaurants to inquire about reservations, Brett decides to enter his location on BiteLine, which displays to him all nearby restaurants, the table availability for each, any promotions offered at that time and most importantly the ability to reserve a table right away. When they arrive at the restaurant to check-in, the Host confirms their reservation made through BiteLine and informs them about the promotion that was offered when they made the reservation and that it will be applied to their bill automatically.

Hosts: On a very busy Friday night at Mario's Dining, Jill, the host, can use the simple Check-In interface of BiteLine to check the reservations of incoming customers. One couple that has a 6:30pm reservation arrives at the restaurant at 7:00pm. Jill is already able to see their reservation displayed on the interface in a section for expected guests. Another couple shows up an hour early and so their reservation is not automatically displayed to her, but Jill is able to quickly search for their reservation by name or reservation time using the Check-In interface. In addition to both of these Check-In features, Jill is also able to see if there are any Promotions that are to be offered for the present party.

Moreover, Jill can see a notification icon on the side representing customers who have made requests during reservation. When she's less busy, she wants to check reservation notes for special requests from the upcoming customers so that the restaurant can prepare appropriately. Since the restaurant gets busy during dining time, Jill will expect at least the above reservation features and information to show up on her Login page.

Owners: Mario, a new business owner of Fine Italian Dining wants to promote his business by offering discounts for first time customers. Mario decides to subscribe to Biteline, which enables him to fill his empty tables by making reservations available on BiteLine and offer discounts for customers. He uses the Submit-Restaurant interface on BiteLine in order to upload photos, add descriptions, menu items, specify when and how much of a discount to offer, and any other relevant information about his restaurant. Using Biteline, he is able to create a promotion that specifically targets first-time customers so that he can reach out to people that haven't yet experienced "Fine Italian Dining". After being approved, by the BiteLine-Admin, Mario's restaurant then becomes a live and active option for customers to be able to find on the site. Additionally, Mario is able to provision Host accounts for his host employees, so that they are able to manage Check-In's just as they normally would as one of his employees, but now they will use the BiteLine service.

Administrator: Keith is the site administrator of ***BiteLine***. First and foremost Keith deals with real-time technical issues that are coming in from Restaurant owners and users alike. For instance, Jim, the restaurant owner, unintentionally put a 50% promotion for all of his meals and can't seem to figure out how to remove the promotion. The restaurant owner would be able to contact Keith and then Keith would have access to alter the

content of the website and fix the issue. However, Keith does receive urgent requests from restaurant owners, sometimes. Keith wants a feature in the restaurant owner page that a restaurant owner can choose "Urgent" when sending the request. This “Urgent” flag could send a special notification to Keith, so that he has the ability to deal with the request right away. In order to keep users updated, she has to manage tasks in a timely manner and have a organized interface in order to do so. Therefore, Keith expects user messages in her Administrator-Page to be categorized into customers and restaurants owners. In this way, she can help users based on priority of the technical issue.

Aside from technical issues, the Admin role is also responsible for approving any pending requests from potential Restaurant-Owner’s to have their restaurant featured with BiteLine. This Approve-Restaurant interface is what legitimizes the BiteLine service. It ensures that everything necessary has been provided by the Owner, as well as keeps the layout and information about restaurants consistent throughout the BiteLine website.

2. Glossary/Data Dictionary

BiteLine – The name of the site.

Customer – Users who want to reserve tables at restaurants and find promotions. Can be registered or unregistered, but when making a reservation they must supply name and phone number, at the least.

Owner – Users who are restaurant owners (or business partners of the site).

Host – Employees of restaurants who use the website to assist in upcoming customers.

Administrator – Specialist who provides technical support for any other users.

Registered User – Become a member of the site and have unique user page and account. Has the added perk of being able to bypass filling out any form information when reserving a table (since they are already logged in).

Customer Page – For the registered or unregistered user of the site. Search bar and navigation links available. Customized page for registered user.

Owner Page – For restaurant owner who has registered a restaurant account. Shall show their restaurant details, current reservations, promotions, images, and any other relevant information for the business owner.

Host Page – For restaurant employees who are registered and logged in as a host. Shall have functionality to find reservations based on time or search, as well as the ability to add new reservations.

Administrator Page – For people who have registered as site administrator. Shall have all of the same interfaces as the other pages, but the ability to edit the layouts and content.

Urgent Flag – A flag attached to a message when restaurant owner makes urgent request for technical help.

Restaurant - One of the designated restaurants that has signed up for the web-service. Restaurants are comprised of a customizable set of parameters, such as number of tables, contact information, description, cuisine, offered promotions and the ability to reserve a table.

Table - By default, a table seats two people. Larger parties will be accommodated by combining more tables.

Priceline - A hotel reservation website modeled similarly to BiteLine

Check-In - The interface for a Host or Owner that allows for customers to be checked in and see any relevant information about a particular reservation.

Submit-Restaurant - The interface or form for Restaurant owners to sign up and upload relevant information about their restaurant. This form will require the base level set of information about a restaurant in order to meet the BiteLine Convention and be approved. Submitted information shall consist of images, descriptions, contact information, and any promotions or discounts to offer and when to offer them.

Approve-Restaurant - Similar to the Submit-Restaurant interface, but for the Admin. This interface ensures that legitimate restaurants are displayed within BiteLine. This is a second layer of screening in order to assure quality restaurant information and BiteLine user experience.

Promotion - A discounted rate or special offer that is available to a Customer and specified by an Owner. Can be triggered based on table-availability, first-time customer, or some other reason specified by the restaurant.

Review - a 1-5 rating of the restaurant, with an optional text block explaining the review.

BiteLine Convention - finished BiteLine page layout, which will be implemented to fit most restaurants' needs for posting and describing their own business information.

3. Functional Requirements

Priority 1

- **Customer:**

  - **All Customers**

1. Customer shall be able to use the website without having to login
2. Customer shall be able to make a reservation by providing only their name and phone number.
3. Customers shall be able to search for nearby restaurants by locations.
4. Customers shall be able to select a neighborhood from a drop-down menu to see restaurants in that area.

- **Registered Customers**

1. Customers having BiteLine account shall be able to perform the same set of tasks which unregistered customers are able to do.
2. Customer shall be able to create an account using email and password.
5. Customer shall be able to login to the website account using email and password.
6. Customers shall be able to edit his/her account related to change of their profiles such as passwords, emails, names, and usernames.
7. Customer shall be able to delete their account with approval of BiteLine's administrator.
8. Customer shall be able to make/change/cancel reservations.

- **Host:**

1. Hosts shall be able to see upcoming/expected reservations on their home pages.

2. Hosts shall be able to search for reservations by time and name.

3. Host shall be able to create/cancel reservations.

- **Restaurant Owner:**
    1. Restaurant owners shall be able to register and upload information about their restaurants to subscribe to the service.
    2. Restaurant owner shall be able to upload a name, description, cuisine type, contact information, location and an image of a restaurant.
    3. BineLine shall follow business rules in order to cooperate with restaurant owners.
    4. Owners shall be able to create/manage host accounts.
- **Admin:**
    1. Admin shall be able to manage restaurants information and all users accounts.
    2. Admin shall be able to edit reviews and other posts on the website.
- **All Users (Host, Restaurant Owners, and Host)**
    1. The website shall be easy to use by average customer without additional training.
    2. Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.
    3. Encryption technologies shall be used to protect the users' passwords.
    4. Search results shall display results within San Francisco.

Priority 2

- **Customer :**
  - **All Customers**
    1. Customer shall confirm their account creation through email
    2. Customer shall be able to decline the account creation within the confirmation email.
  - **Registered Customers**
    1. A login customer shall have a personalized home page.
    2. Customers shall be able to post reviews.
- **Host:**
    1. Host shall be able to browse reservations by time.
    2. Hosts shall be able to type in a customer name, and select a time frame from a drop-down menu. If no results are found, site shall display results around that time frame.
- **Restaurant Owner:**
    1. Owners shall be able to add promotions to their restaurants.
    2. Owners shall be able to contact the administrator through the site, with the option to mark their help request as urgent.
- **Admin:**
    1. Admin shall be able to see help requests by restaurant owners and users.
    2. Admin shall be able to see help requests by restaurant owners and users.

3. Admin shall be able to respond to help requests by restaurant owners and users.

- **All Users (Host, Restaurant Owners, Host and Admin)**
    1. Search results shall be based on a default location if users do not specify a neighborhood from the drop-down menu for search.

4. Non-Functional Requirements

- Application shall be developed using class provided LAMP stack
- Application shall be developed using pre-approved set of SW development and collaborative tools provided in the class. Any other tools or frameworks have to be explicitly approved by Marc Sosnick on a case by case basis
- Application shall be hosted and deployed on Amazon Web Services as specified in the class

- Application be viewable in a standard desktop/laptop/mobile browsers, and shall render correctly on the two latest versions of all major browsers: Mozilla, Safari, Chrome and IE.
- Data shall be stored in the database on the class server in the team's account
- Application shall be served from the team's account
- No more than 50 concurrent users shall be accessing the application at any time
- Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.
- The language used shall be English.
- Application shall be very easy to use and intuitive. No prior training shall be required to use the website.
- Google analytics shall be added for major site functions.
- The website shall prominently display the following text on all pages "SFSU/FAU/Fulda Software Engineering Project, Fall 2014. For Demonstration Only". (Important so as to not confuse this with a real application).
- 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
- Supported Browsers
    - Chrome - versions C44, C45, C46
    - Firefox - versions FF39, FF40, FF41, FF42

Internet Explorer - versions IE9, IE10, IE11, EDGE

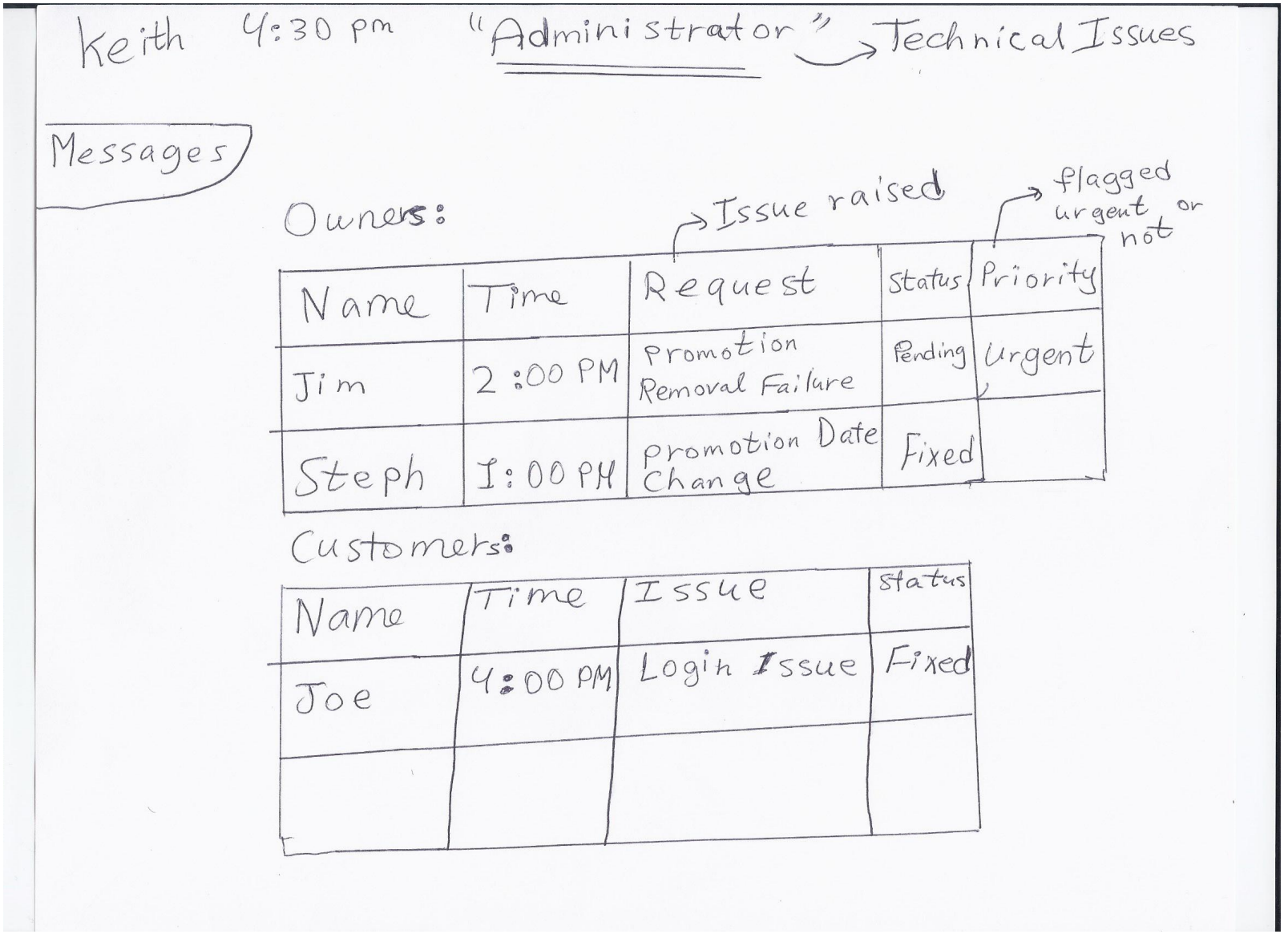
-

5. UI Mockups and Storyboards (high level)

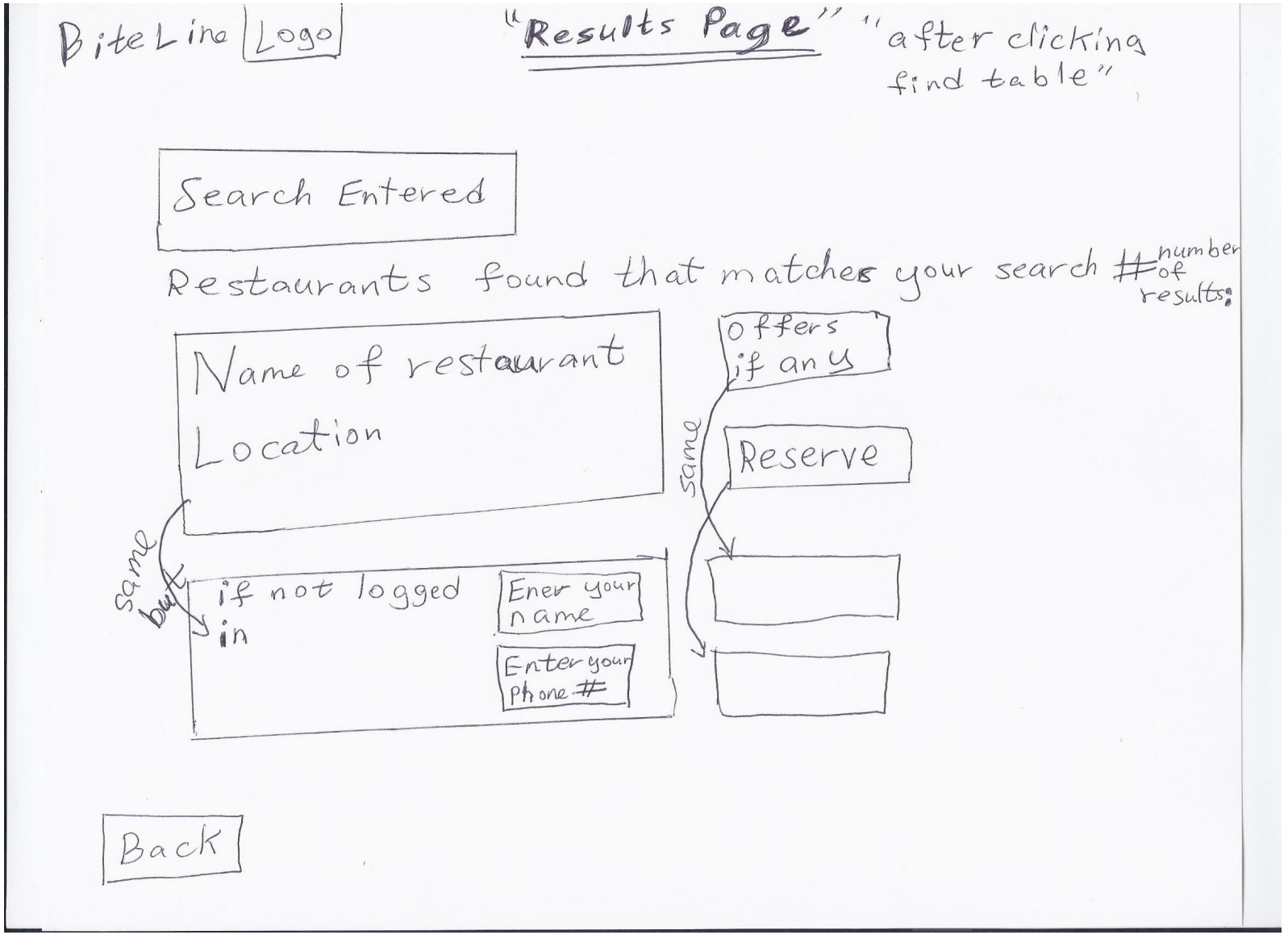
Home Page:

Administrator Page:

Results Page:



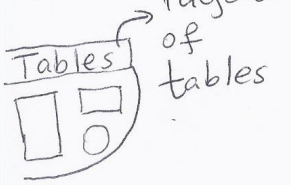
Host Page:

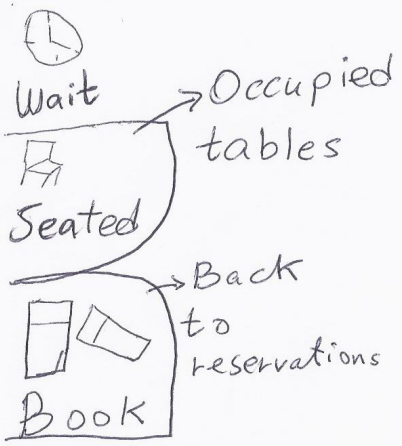

BiteLine Logo
"Results Page" "after clicking find table"
Search Entered
Restaurants found that matches your search # number of results:
Name of restaurant
Location
Offers if any
Reserve
same
same but
if not logged in
Ener your name
Enter your phone #
Back

Jill 2:30pm

"Host"

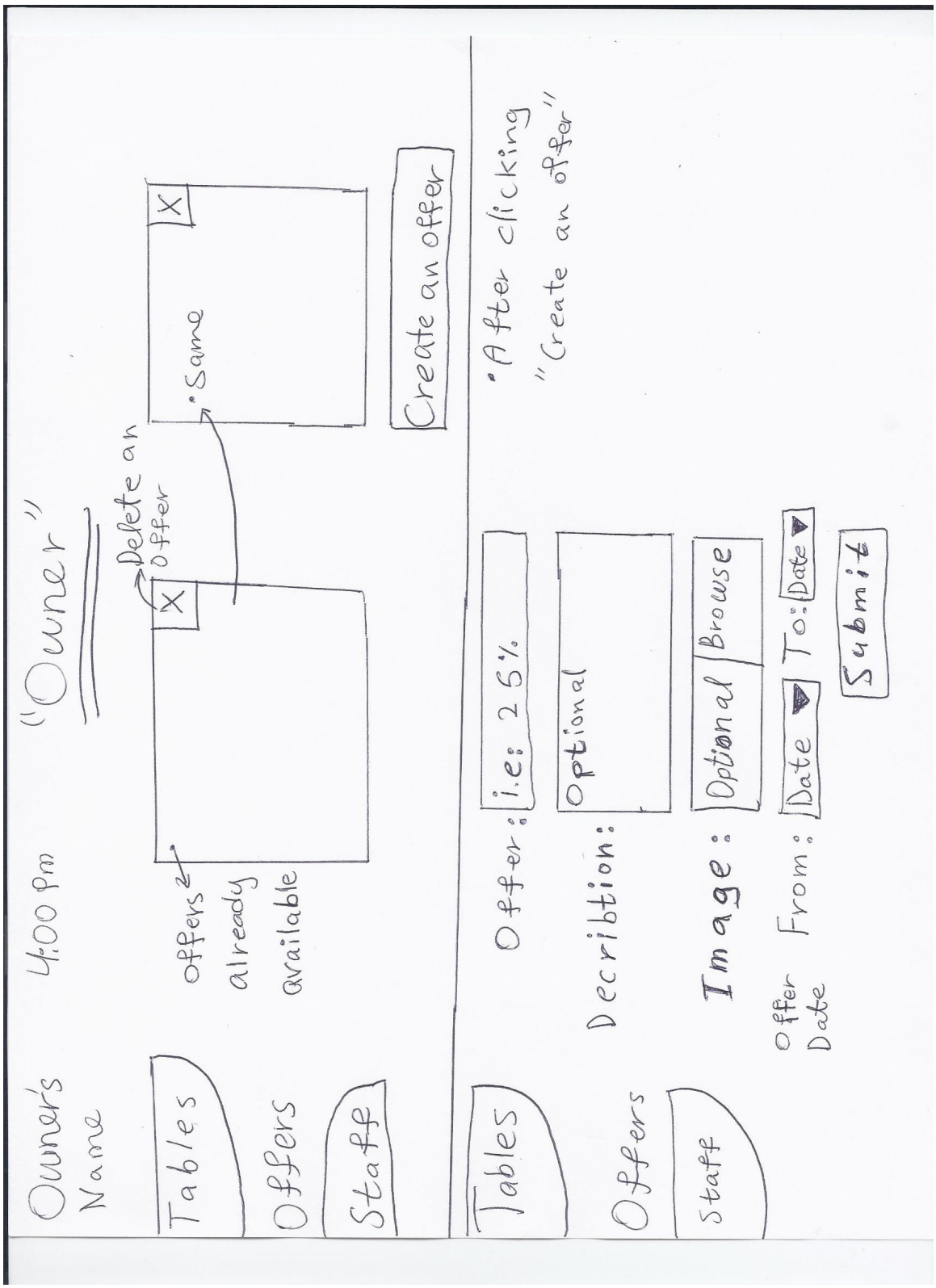
↑
Hostname





Guest	Size	Time	Table#	Notes/ Messages
Abby	4	4:00 Pm	2	Birthday
James	6	Arrived	11	Request Patio
⋮	⋮	⋮	⋮	⋮

Owner Page:


Owner's Name
4:00 pm
"Owner"
Tables
Offers
Staff
offers already available
Delete an offer
X
X
•Same
Create an offer
•After clicking "Create an offer"
Tables
Offers
Staff
Offer:
i.e: 25%
Decribtion:
Optional
Image:
Optional
Browse
Offer Date
From:
Date
To:
Date
Submit

6. High level Architecture and Database Organization

1. System
    - Linux
    - Apache
    - MySql
    - PHP
    - jQuery
    - HTML/CSS/Javascript
2. API
    - Google APIs
3. Tools
    - Netbeans
    - SVN
4. Framework
    - Bootstrap

BiteLine Database Design (High-Level Description)

Basic Description:

- Database structure for BiteLine shall include a series of basic entities.
- Each entity represents a table with specific columns in database.
- Each entity shall store only data related to the that entity.
- If an entity requires information from another entity, a foreign key representing the separate entity is established in current entity.

- Images for BiteLine shall be stored as BLOB data type.
- PhpMyadmin is used as a tool to create tables and ERD (model) diagrams for demonstration.
- Database data shall be retrieved using SQL SELECT clause with PHP.

BiteLine Database Design shall include the following Entities:

- **Cuisines:** the main food style of a restaurant which is described by a distinct name and basic description of the food style. Each type of cuisine is identified by a unique cuisine id. Addtionally, a cuisine also has a name, image, and description explaining the food style.
- **Accounts:** account for registered users including Customers, Restaurant Owners, Hosts, and Administrators of BiteLine. Each account contain information of the current user. Each account has a unique account id and unique user name. A user name is the name registered to BiteLine. A real name is also required for an account which is used for reservation.
- **Images:** stores all images of BiteLine including Account images, Restaurant images, and Neighborhood images. Each image is assigned with a unique image id.
- **Neighborhoods:** all neighborhoods in San Francisco which will be categorized according to the current District Map of San Francisco. Each neighborhood has a unique neighborhood id. Each neighborhood has a name and brief description.
- **Reservations:** information that is sufficient to identify a successful table reservation to a restaurant for each customer. When each reservation is made, a

unique reservation id is generated. Date, time, and restaurant information are recorded for each reservation as well.

- **Restaurants:** stores associated restaurants (business partners) of BiteLine. Each restaurant will provide contact information stored into Database and used by BiteLine. Each restaurant is assigned with a unique restaurant id. Restaurants with different restaurant id's can have the same name. For example, there are many McDonald's in San Francisco, but each McDonald has a unique restaurant id.
- **Roles:** types of Biteline Accounts including Customer, Restaurant-owner, Host and Administrator. Description of the role duty and the name of the corresponding role shall be stored in this table. Each role has a unique role id.
- **Rt_contacts:** short for Restaurant Contacts. Rt_contacts stores contact information of a restaurant including its address, phone number, and website. Each contact has a unique contact id.

Database Schema Design with data type specified:

- **Cuisines** (cuisine_id: *INT*, name: *CHAR[70]*, description: *CHAR[255]*, img_id: *INT*);
- **Accounts** (ac_id: *INT*, username: *CHAR(50)*, realname: *CHAR(50)*, password: *CHAR(50)*, email: *CHAR(50)*, role_id: *INT*, img_id: *INT*);
- **Images** (img_id: *INT*, image: *BLOB*, name: *CHAR(50)*);
- **Neighborhoods** (nbh_id: *INT*, name: *CHAR(70)*, description: *CHAR(255)*, img_id: *INT*);

- **Reservations** (r_id: *INT*, num_of_people: *INT*, date: *DATE*, time: *TIME*, request: *CHAR(255)*, rt_id: *INT*, ac_id: *INT*);
- **Restaurants** (rt_id: *INT*, name: *CHAR(70)*, description: *CHAR(255)*, approve_status: *BOOLEAN*, nbh_id: *INT*, cuisine: *INT*, contact_id: *INT*, img_id: *INT*);
- **Roles** (role_id: *INT*, name: *CHAR(50)*, description: *CHAR(255)*);
- **Rt_contacts** (contact_id: *INT*, address: *CHAR(95)*, phone: *CHAR(13)*, website: *CHAR(255)*);

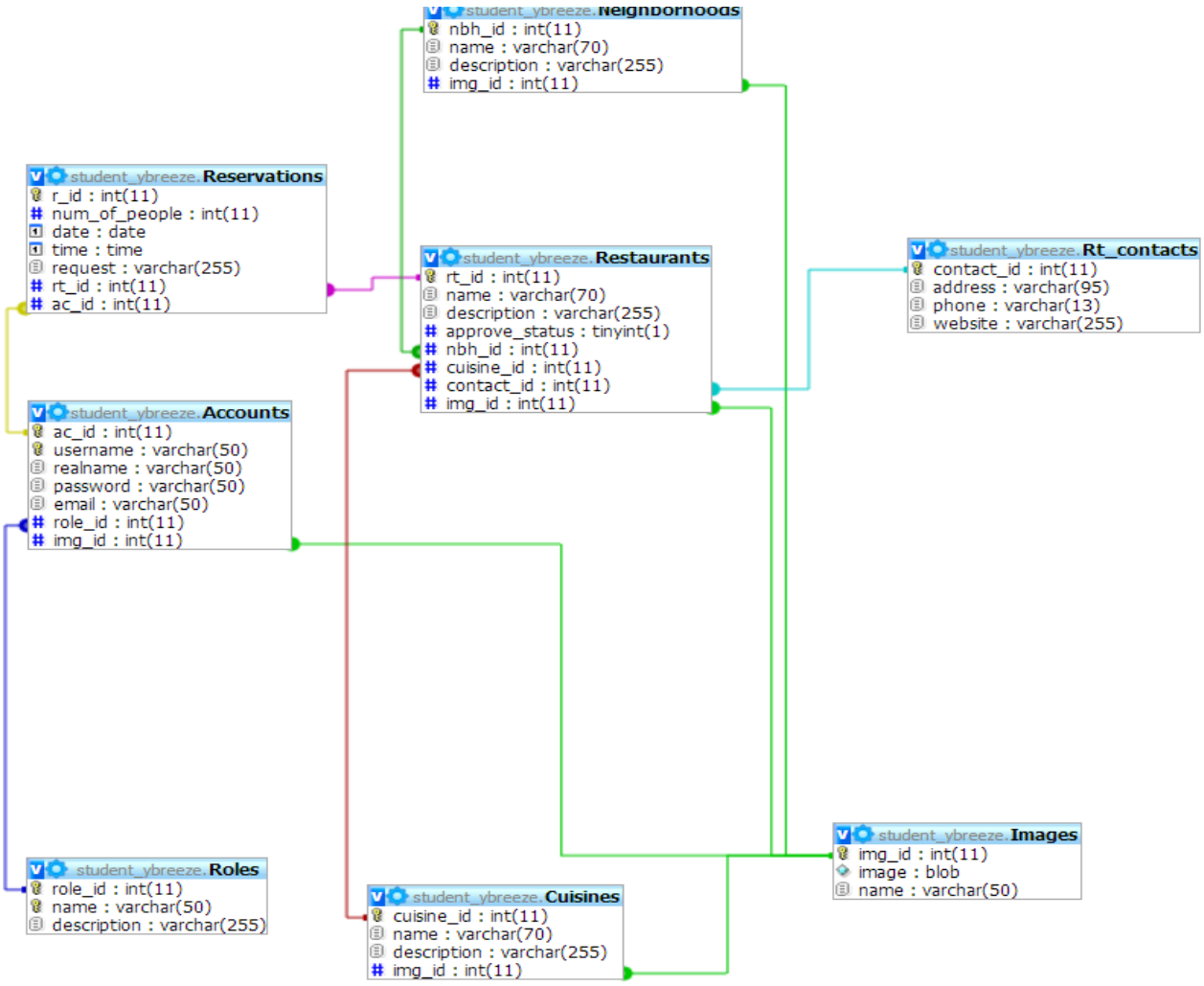
Relations between database entities:

Note: the relation description is based on foreign key constraints and actual data which will be stored into the database.

- **Restaurants-Rt_contacts:** the relation states that each restaurant has its own contact information.
- **Restaurants-Neighborhoods:** the relation states that each restaurant locates at an existing neighborhood in San Francisco.
- **Restaurants-Cuisines:** the relation states that each restaurant has a major food style.
- **Accounts-Roles:** the relation states that each account has an account type which is categorized as Customer, Restaurant Owner, Host or Administrator.
- **Reservations-Accounts:** the relation states that a reservation is made by an existing account. Moreover, different reservations can be made by the same account.

- **Restaurants-Images:** the relation states that each restaurant has an image (building or interior).
- **Accounts-Images:** the relation states that each account has an account image.
- **Neighborhoods-Images:** the relation states that each neighborhood has an image that represents the neighborhood.
- **Cuisines-Images:** the relation states that each food style has an image that identifies a food style.

Relations between database entities in ERD model view:



Database schemas in table views:

- Table: Cuisines

cuisine_id	name	description	img_id

- Table: Accounts

ac_id	username	realname	password	email	role_id	img_id

- Table: Images

img_id	image	name

- Table: Neighborhoods

nbh_id	name	description	img_id

- Table: Reservations

r_id	num_of_people	date	time	request	rt_id	ac_id

- Table: Restaurants

rt_id	name	description	approve_ status	nbh_id	cuisine_id	contact_id	img_id

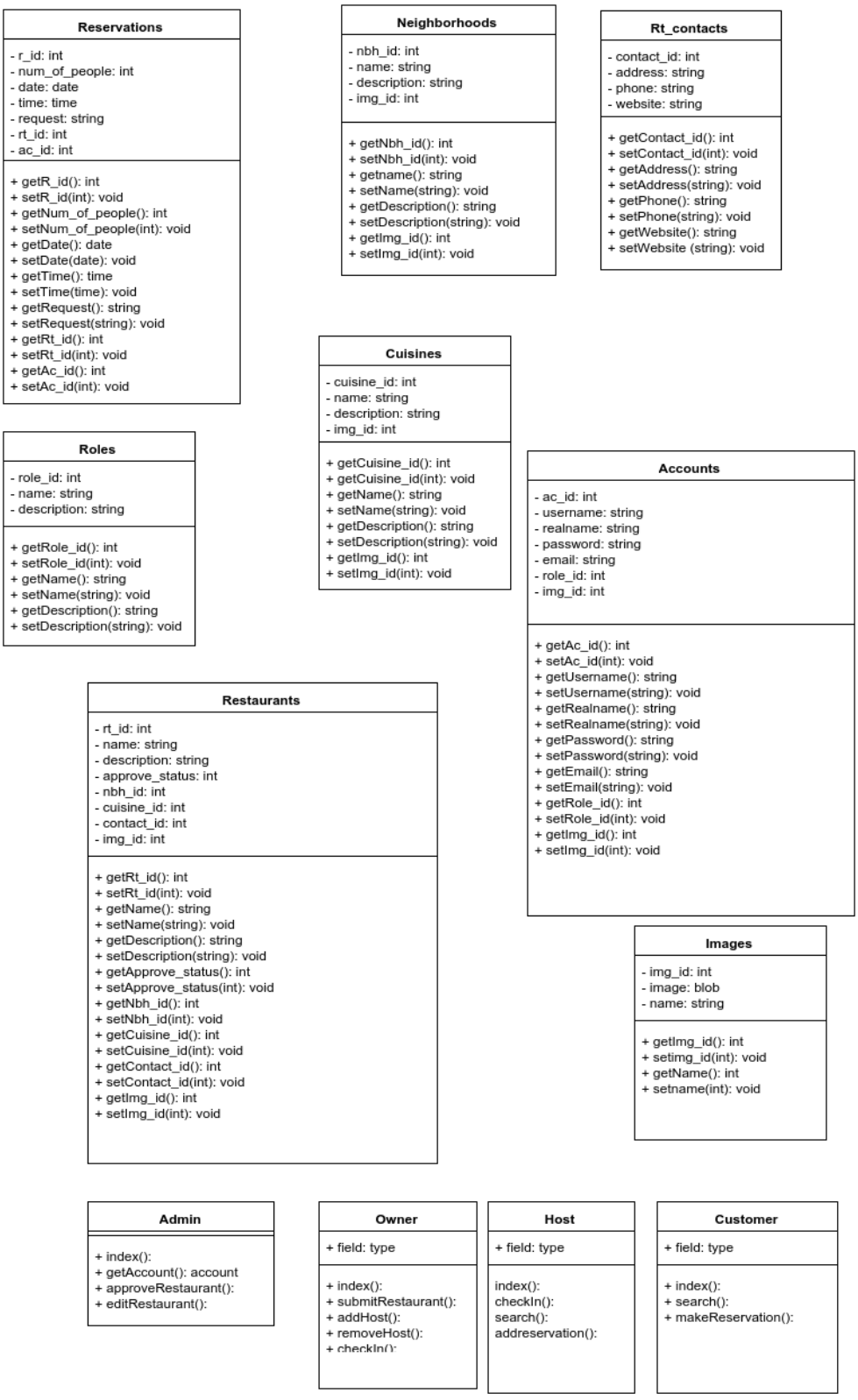
- Table: Roles

role_id	name	description

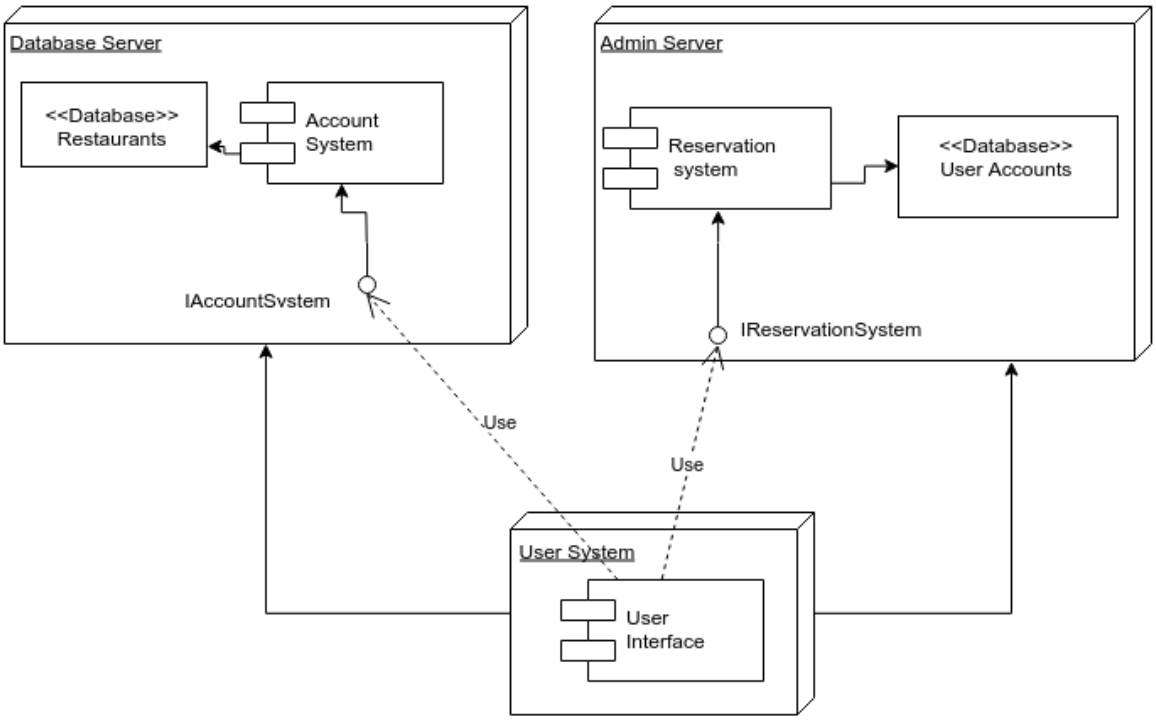
- Table: Rt_contacts

contact_id	address	phone	website

7. High Level UML Diagrams



Deployment Diagram



8. Key Risks

- Skill risks - Most of our team is unfamiliar with web development, so it will be a bit of a challenge for us to collaborate together while learning how elements mesh together.
- Technical risks - The server may go down sometimes when we need to deploy, so we have to make sure to complete development early or risk not deploying on time.
- Schedule risks - Although all team members have one day out of the week to meet in person other that the class meeting, our schedules can’t accommodate meeting any more than that.

9. Team Organization

George Jone - Team lead. Responsible for task organization, assignment, and make sure milestones are met on time.

Steve Pedersen - Tech lead. Responsible for overseeing back-end design and creating an initial framework scaffold/vertical prototype.

Jacob Abba - Responsible for back-end and BLOB demo.

Abdullah Alnamlah - Responsible for front-end and creating the mock-ups.

Puyang Yu - Responsible for back-end and database design for further improvement.

Darin Vergara - Responsible for UML diagrams and general back-end architecture.