

# **TripTips Status Document**

Members: Willow Livengood, Harry Chong, Paras Doshi, and Tommy Lê

## **Project Idea:**

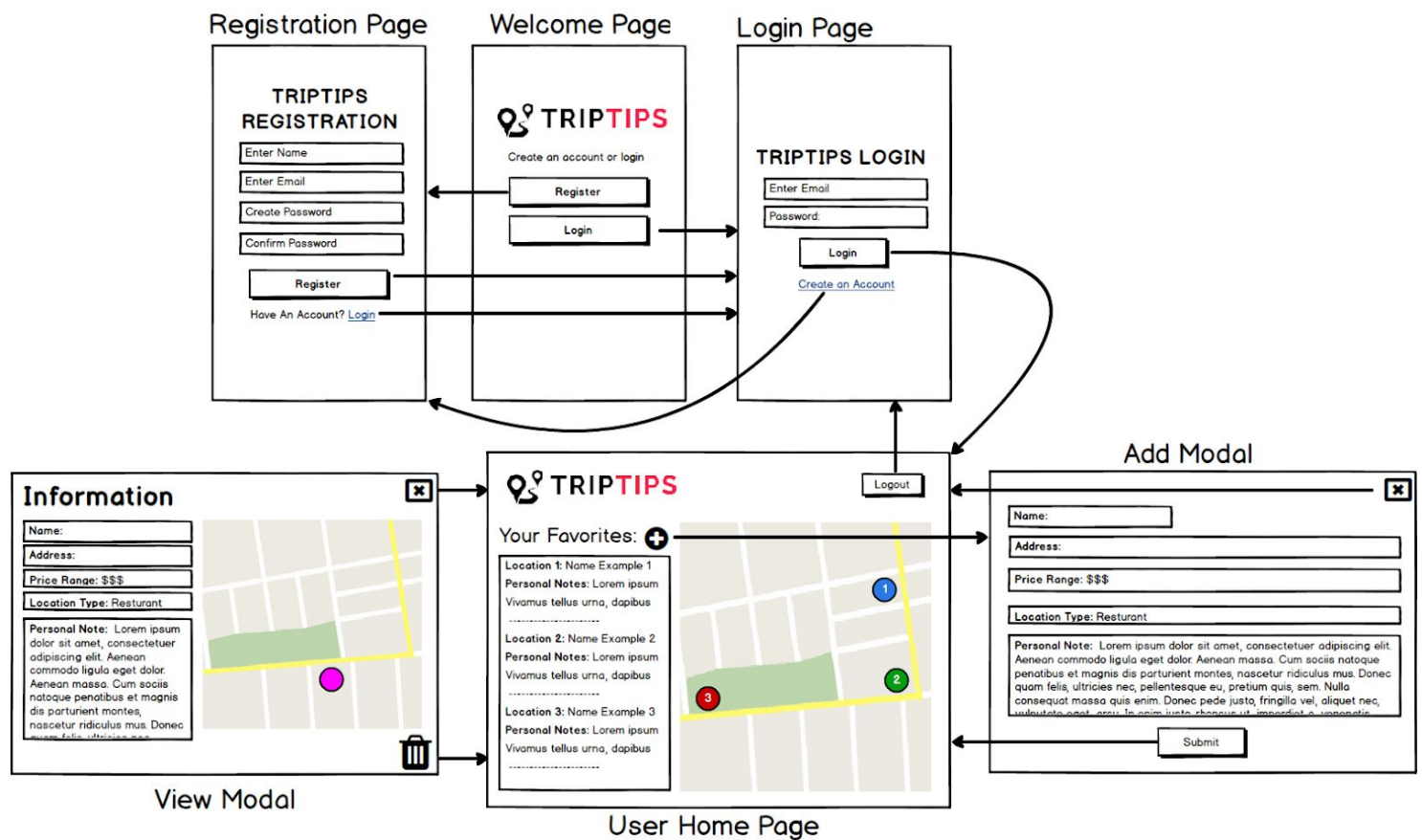
Our project idea is to create an application called TripTips, which stores, centralizes and uniformly formats travel recommendations from friends and family in one convenient location. A common situation when discussing travel plans with friends who have previously been to your vacation destination is that they will likely have recommendations of places to go and personal notes about their experiences. There are many different ways to collect this information, e.g. email, text or notes when talking in person, and many different ways to share this information, e.g. links to places on google maps or a simple list of names. The plethora of ways of sharing this information creates a challenge when trying to view recommendations on the go with limited cellular data. The person traveling, or collecting these tips from friends and family, will be able to register an account to begin crafting their own personal list. Each location that is entered by the user will be able to define the location name, address, approximate price, location type, and personal notes. Centralizing these recommendations based on location in an attractive user interface will create a stress-free method of quickly pulling up stored locations on the go while traveling.

## **Functional requirements:**

The application registers and stores multiple users. It has a custom user experience based on each account. Each user has a homepage that shows all saved locations on a map, and a quick overview of each location. The homepage also allows the user to log-out. If there are no locations, a prompt to add a location will be displayed to the user. Users are able to add locations to their saved lists through a form. Each location has the attributes: location name, address, approximate price, location type, and personal notes. Except for location name or address, each of these attributes are optional. If the location is not found in the map, the entry will not be added to the database. Once saved, the location is plotted on a map and their details will be shown in an information page. Users can also remove their saved locations on the detailed information page. See the architectural wireframe for a detailed diagram of the user experience workflow. The main external component of the application is the use of the Google Maps Embed API and Google Geocoding API on the basic tier. Documentation link provided in the Technologies Used section. Requests to the API are used to search for and display the locations using longitude and latitude information. The geolocations are then used to mark the location on a Google map. Another major component is database accessing and storage with MySQL. MySQL will store the users, and corresponding saved location information.

## Architectural Design and Wireframe:

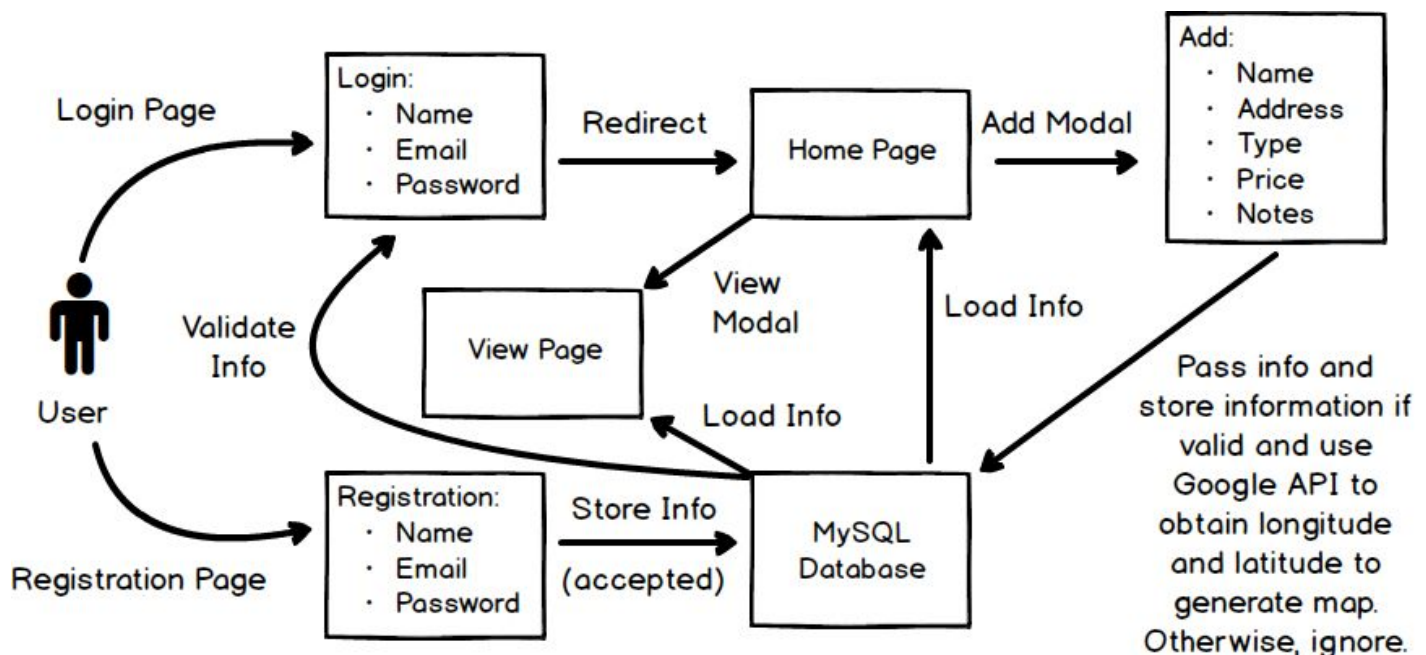
When a user accesses the application, they will first be greeted by a welcome page. Returning users will be able to log-in and view their custom home page. However, if a new user accesses the application, they will have an option to create an account. Once on the home page, users will have access to their location list and a map showing pins for each location. The user can then either add a new location or view saved ones. To add a new location, the user has access to an Add Modal where the user will be expected to define the location name, address, approximate price, location type, and personal notes. Once the user has locations stored in their account, they can view the locations. To view a location, the user will be redirected to the View Modal which displays the chosen location plotted on a map, and their notes and descriptions about the location. The user will also have the option to delete the location on this page if needed. Below is a wireframe diagram of the application workflow.



## Information Flow Diagram:

Upon starting the application, the user can either log-in or create a new account. In both cases, the user will need to supply their name, email, and password. On the registration page, if the inputted credentials are valid, the information will be stored in the database. On the login page, the inputted credentials will first be validated from any preexisting information in the MySQL database. If the results are valid, the user will be redirected to the home page.

Upon accessing the home page, the page will pull and load any information that is linked to the user account. This includes any location information and maps. On the home page, the user has two options, add a new location or view a saved location. With the add option, the user will need to input the location name, address, type, price, and an optional personal note. When the information is submitted by the user and verified, all the information will be stored into the MySQL database which will then be used in the “new” updated home page. With the view option, any information related to the chosen location will be pulled from the MySQL database and loaded onto the application. Below is an information flow diagram of the application.



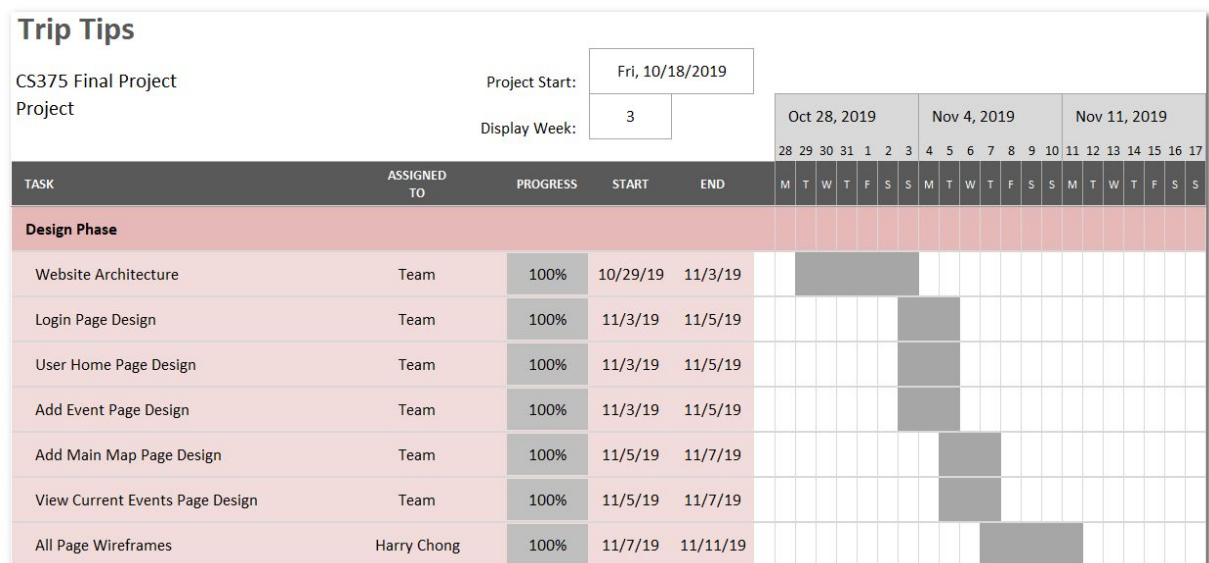
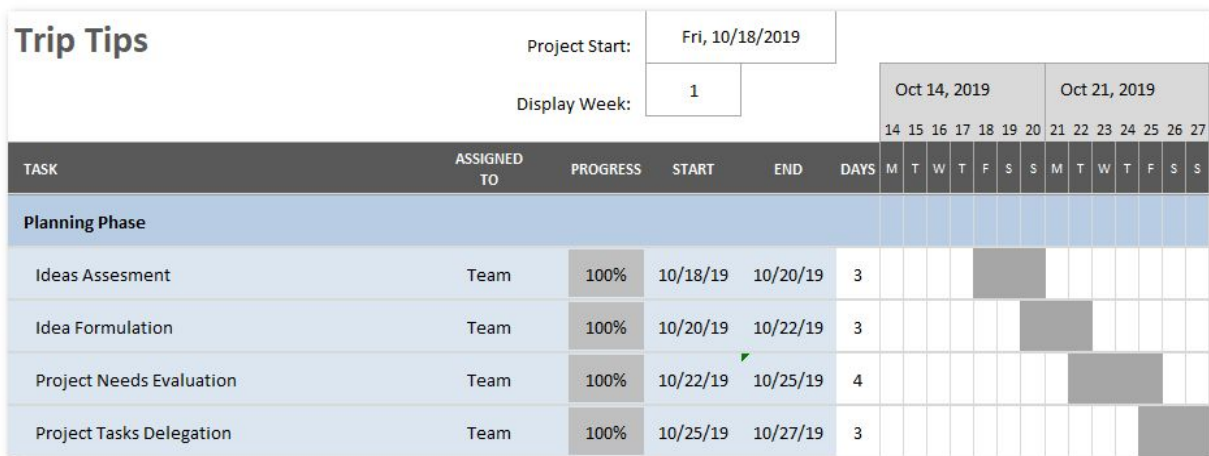
## Technologies Used:

Creating the appearance of the application relies on HTML5 and CSS. JavaScript, under the ES7 standard, and AJAX, for the endpoint calls, drives the functionality of the application. JSON formatting and parsing are vital in accessing and responding to endpoints and database requests. Node.js is the server running the application. A database was created using MySQL. Google Maps Embed API on the basic tier is used for location services. Documentation for this API is located at <https://developers.google.com/maps/documentation/embed/guide>. The Google Geocoding API was also utilized in order to obtain the longitude and latitude of the location to generate a map with markers. The use of the passport.js module simplifies the process of authenticating and validating the user's credentials. Lastly, the use of ejs (Embedded JavaScript Template) helps with generating HTML markup with plain JavaScript.

## Gantt Chart:

The development of this application is divided into four roles, one for each team member. The frontend was handled by Harry and Willow. Tommy focused on QA testing and CSS styling of the application. Willow focused on the HTML and JavaScript functionality of each page and required Ajax calls. The backend was handled by Harry and Paras. Paras focused on the creation of endpoints in the server required for communication between the database and web pages as well as the GeoCoding API. Harry was in charge of database creation and management, and the creation of API calls.

Below are four screenshots of the project's Gantt Chart created through Excel. Each screenshot represents individual phases of the project and the corresponding weeks and days of when certain tasks should be started and completed. As a reminder based on what has outlined above, each task is assigned to either an individual member of the team or the entire team. Key events began in the design phase which is where we started designing the main structure of the application. The fabrication phase was the core step of when we created a working prototype of the application. It should also be noted that, the blocks in grey suggest the task has been completed, whereas purple suggests that it still needs to be worked on. Blocks with both colors are in progress (with respect to the day of writing).



## Trip Tips

CS375 Final Project  
Project

Project Start: Fri, 10/18/2019

Display Week: 5

Project		Display Week:	5	Nov 11, 2019							Nov 18, 2019							Nov 25, 2019							Dec 2, 2019								
				11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	1	2	3	4	5	6	7	8		
TASK	ASSIGNED TO	PROGRESS	START	END	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S		
Fabrication Phase																																	
Endpoints for login and home pages	Paras Doshi	100%	11/14/19	11/21/19																													
HTML and CSS for login and home pages	Harry Chong	100%	11/14/19	11/21/19																													
Javascript for login and home pages	Harry Chong	100%	11/14/19	11/21/19																													
User and location database instantiation	Harry Chong	100%	11/14/19	11/17/19																													
Endpoints for Add and View Events	Paras Doshi	100%	11/21/19	11/26/19																													
HTML for Add and View pages	Willow Livengood	100%	11/21/19	11/26/19																													
Javascript for Add and View pages	Willow Livengood	100%	11/21/19	11/26/19																													
Test Website and Debug Issue	Team	100%	11/27/19	12/2/19																													

## Trip Tips

CS375 Final Project  
Project

Project Start: Fri, 10/18/2019

Display Week: 5

Project	Display Week:	5																															
			Nov 11, 2019							Nov 18, 2019							Nov 25, 2019							Dec 2, 2019									
			11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	1	2	3	4	5	6	7	8			
TASK	ASSIGNED TO	PROGRESS	START	END	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	
Project Pitches	Team	100%	10/24/19	10/29/19																													
Status Document	Team	100%	11/16/19	11/22/19																													
Poster Document	Team	100%	11/29/19	12/7/19																													
Final Project Presentation	Team	100%	11/29/19	12/7/19																													

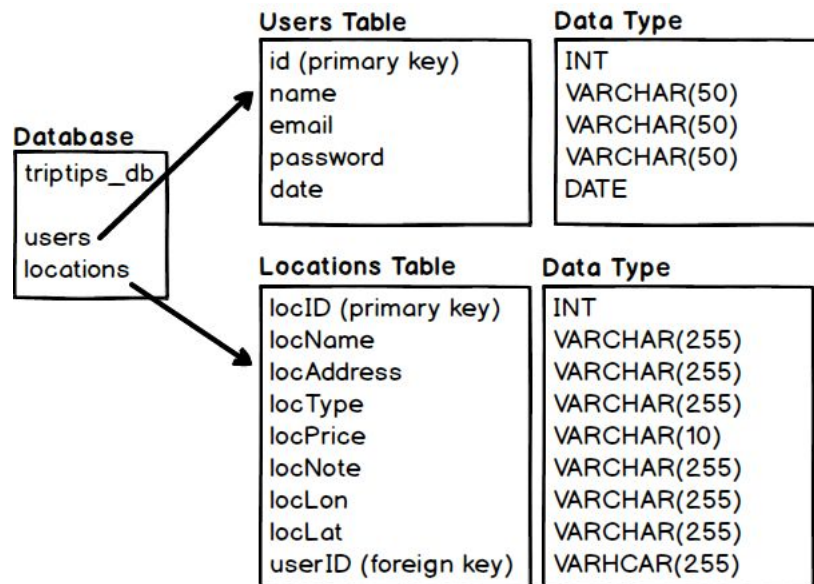
## Database Schema:

Since the application is using a MySQL database as a way to store information, the database setup is critical. The database is called triptips\_db and within this database, there will be two tables: users and locations. The users table contains all the information needed to track whether a valid user account exists or not. In addition, the user account will be needed to display the appropriate information. As a result, the users table uses the id as the primary key and tracks the user password (password) and email as a form of validation for user identification. The date column is used to record when the account was created.

The locations table contains any listings that have been created by a user. Individual listings will be linked to the username (userID) as the foreign key. This will ensure that any listings created by a user will be displayed accurately. In addition, the location table will hold the location name (locName), location address (locAddress), location type (locType), location price (locPrice), location note (locNote), and the longitude (locLon) and latitude (locLat). The longitude and latitude of the address is generated through the Google Geocoding API. If the address submitted is invalid, the information will be ignored and will not be added to the database. These values will be changed based on the user's input.

All of the data that is being recorded has been assigned the appropriate data types. Below is a diagram of the database schema.





## Screenshots:

The images below shows the final presentation and layout of the TripTips application for the various pages and modals.

### Welcome Page:

**TRIP TIPS**  
Create an account or login

REGISTER

LOGIN

### Login Page:

**TRIP TIPS LOGIN**

Email  
Enter Email

Password  
Enter Password

LOGIN

No Account? Register

## Registration Page:

# TRIP*TIPS* REGISTRATION

Name

Enter Name

Email

Enter Email

Password

Create Password


Confirm Password

Confirm Password

REGISTER

Have An Account? [Login](#)

## Home Page:



# WELCOME HARRY CHONG

## YOUR FAVORITES

**LOCATION 1:**  
Times Square

**ADDRESS:**  
Manhattan, NY 10036, USA

**LOCATION TYPE:**  
Attraction

**LOCATION PRICE:**  
N/A

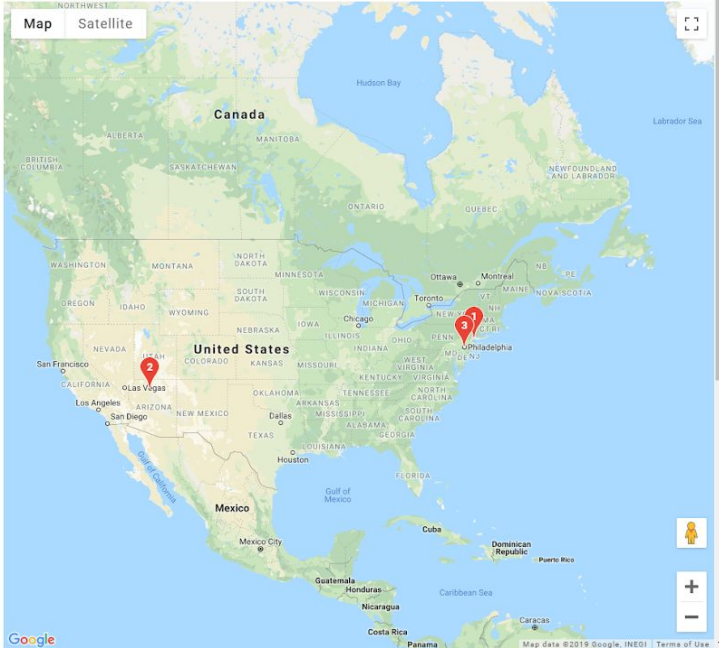
**LOCATION 2:**  
Grand Canyon National Park

**ADDRESS:**  
Grand Canyon National Park, Arizona, USA

**LOCATION TYPE:**  
Attraction

LOG OUT

Map Satellite



Google

### View Modal:

[illegible]

### Add Modal:

ADD LOCATION

NAME:

ADDRESS:

PRICE RANGE:

N/A

LOCATION TYPE:

N/A

PERSONAL NOTE:

SUBMIT