CMPE 280

PROJECT REPORT

TOUR GUIDE FOR SPARTANS

Submitted By:
Chaya Malik
Minglei Lu
Yiming Zhai
Zening Deng

TABLE OF CONTENTS

ABOUT OUR APPLICATION

International Spartans want to spend a good time during weekend or holiday. Since they don't have too much time for this how about providing them with a web application that will help them plan their trip to different locations without taking much of their time.

Most of the Spartans are from either out of the California or out of the United States, they need a thorough guide to some "You can't miss" places on the west part of the United States Of America.

To help the Spartans explore the west coast and make sure they face few problems while doing that, we are coming up with a web application that serves as a travel guide for Spartans. Our website helps the Spartans choose the places they want to visit, and also gives them suggestions if they are not sure of places they want to visit.

Once they choose their place of interest we provide them information about the famous tourist spots, restaurants, local activities and other information which helps them make a decision about their next trip.

The difference between our application and an application like tripadvisor is that our app contains reviews by fellow spartans. This helps the people new to west coast of USA or california get easy reviews about how to visit places, enjoy amazing cuisines and still save some bucks.

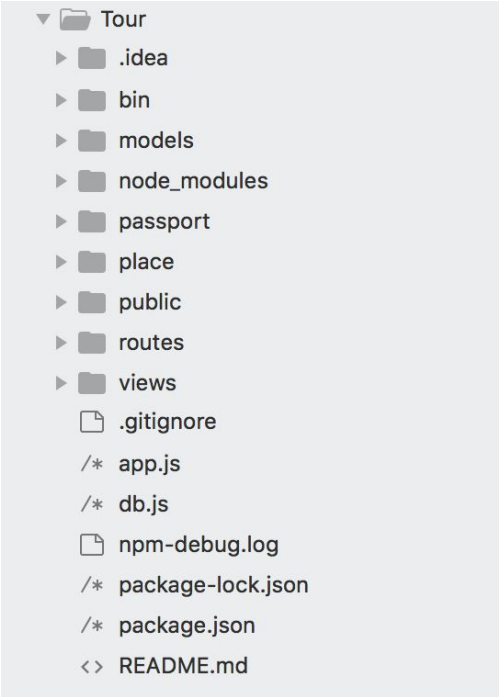
ARCHITECTURE

We create whole project using command line:

“express Tour”

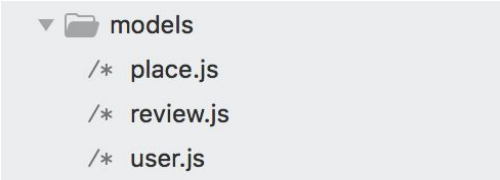
“cd tour&npm install”

Be default, the whole is structured as following:

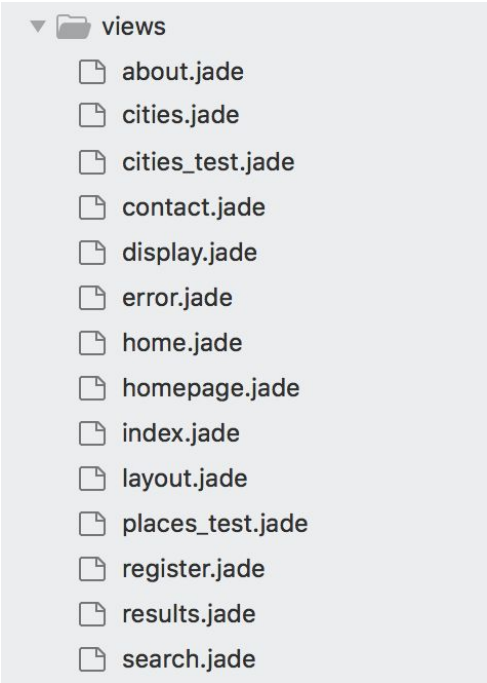


The structure is based on MVC (model - view - controller)

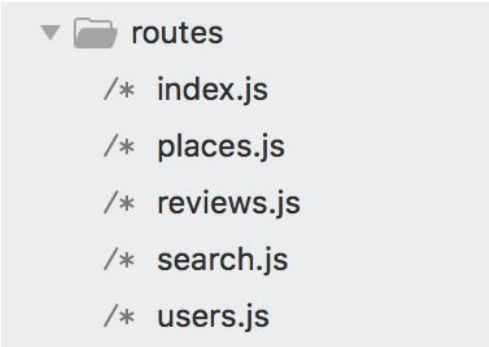
Models folder is where we handle our data. We use mongoose to communicate with database, as we define different schema for each collection.



Views folder is where we keep all the interfaces. We use jade as view engine to dynamically generate html page.



Routes folder is where we keep all our functions to handle HTTP request from client side. We implement it by routing between different middlewares.



REST API

We use REST API mapping database in routes folder. We design routes to handle HTTP request that contains all the information to manipulate the data in the database.

Following is an example code for implementing search function:

```
router.post('/search', function(req, res) {
  var keyWord = req.body.key;
  console.log(JSON.stringify(req.body));
  var regex = new RegExp(["^", keyWord, "$"].join(""), "i");

  Place.find({'keywords': regex}, function(err, doc){
     console.log(regex);
     console.log(doc);
     console.log('Got through');
     res.render('search', {results: doc, keyWord: keyWord});
  });
});
```

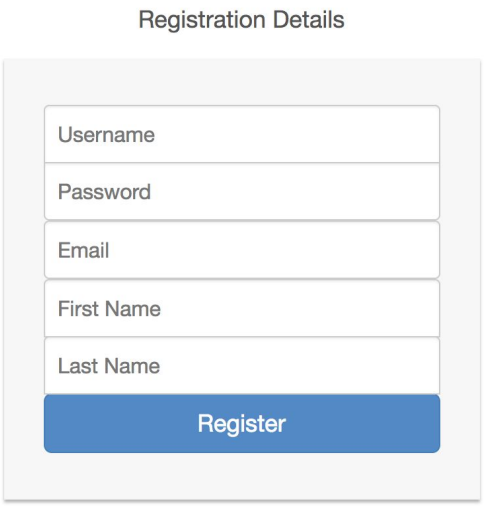
FEATURES & IMPLEMENTATION

AUTHENTICATION

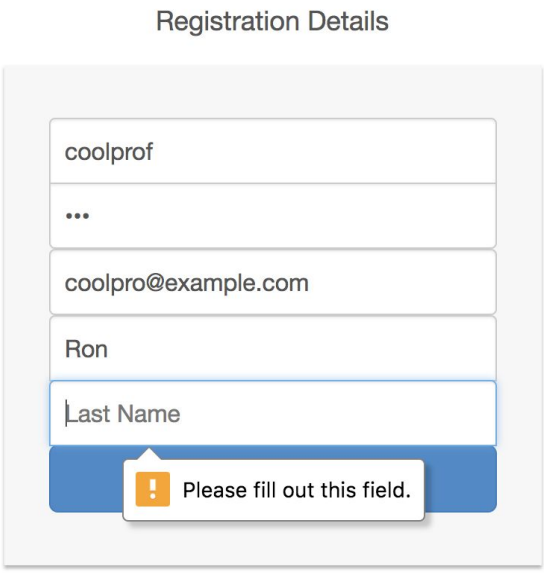
We have used Passport.js for implementing authentication in our web application.

Passport is authentication middleware for Node.js. We have used it to authenticate request from client side - to be specific, the user credentials and login authentication.

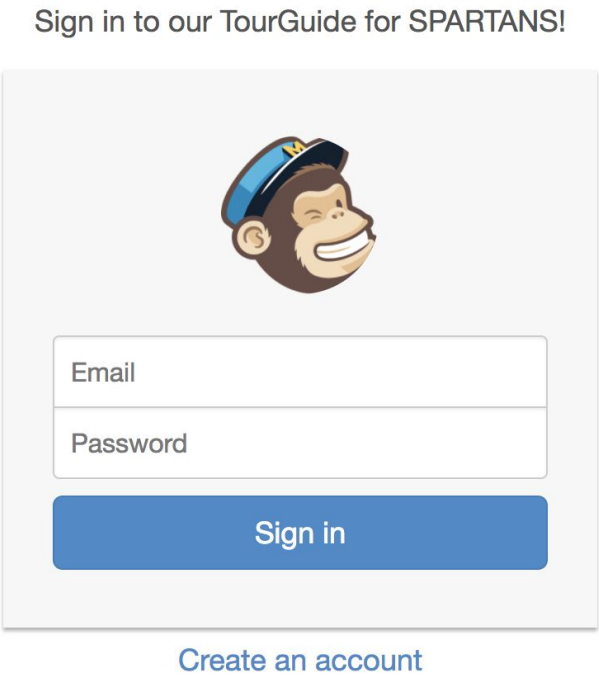
User Signup



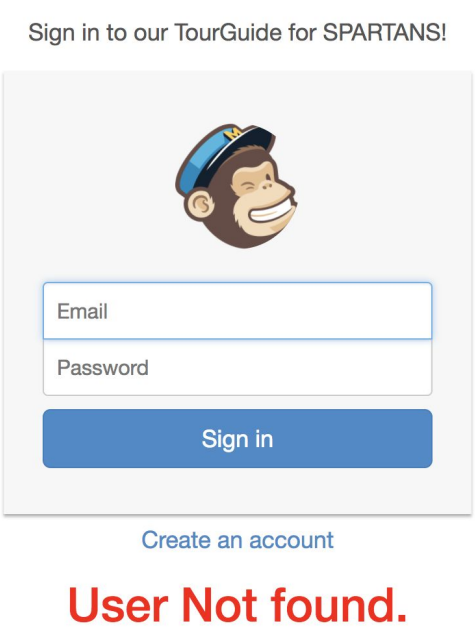
User Signup fields required checked

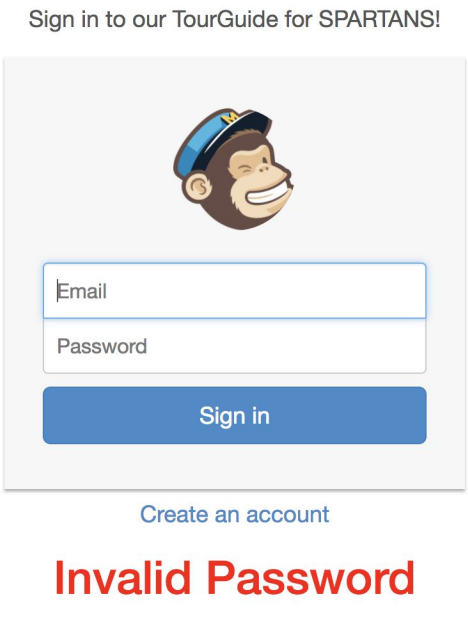


User login



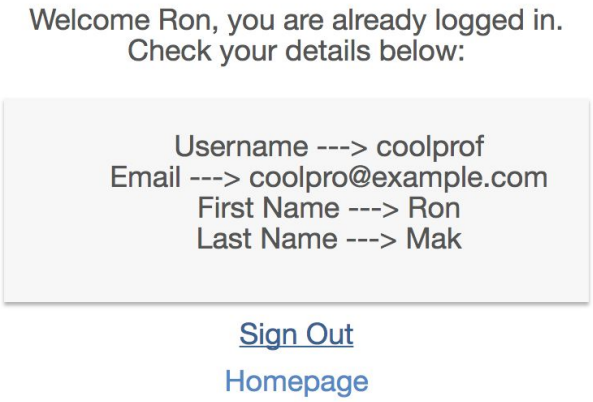
Login email and password authentication



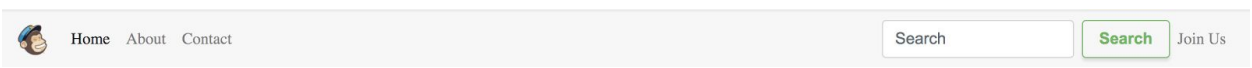


Login successfully

Logout

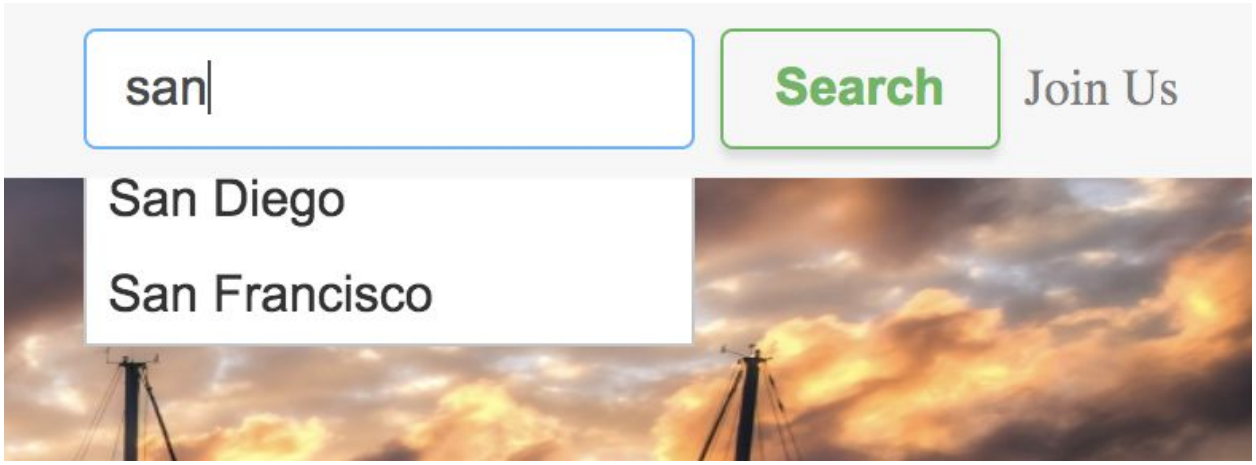


Logout successfully

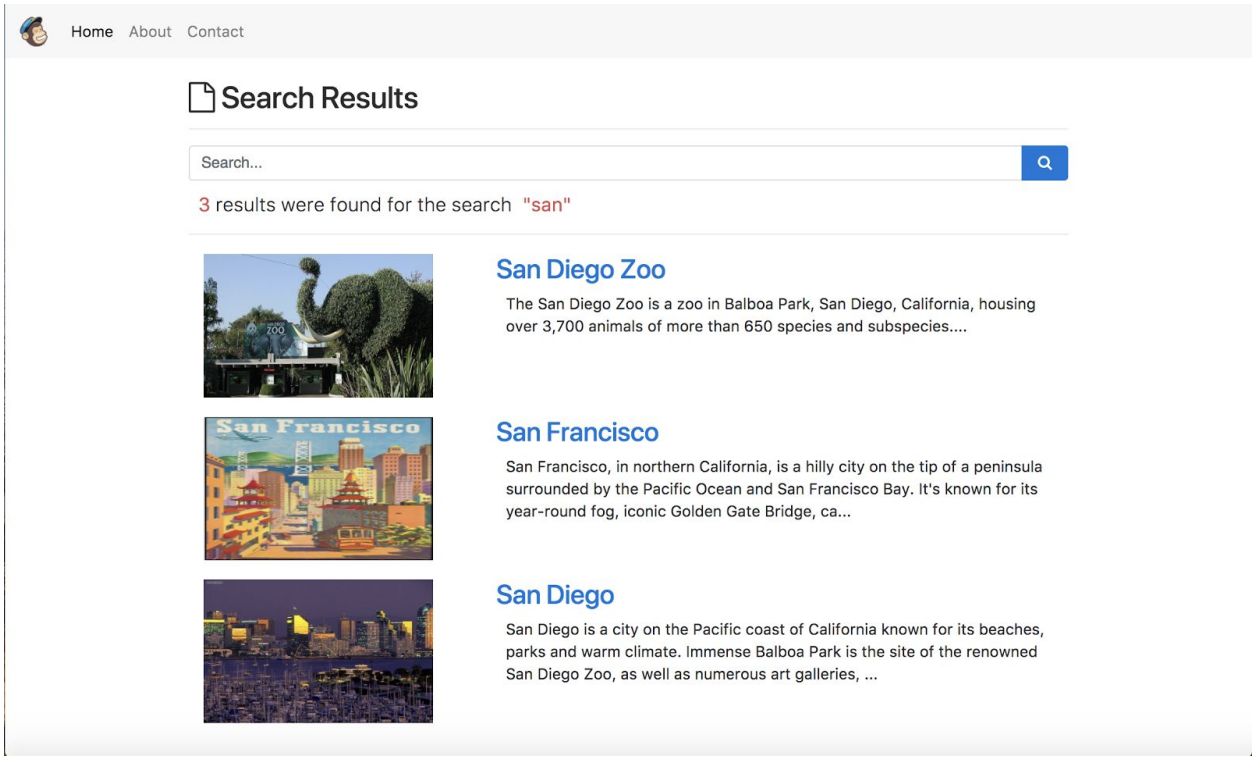


SEARCH BAR TO LOOK UP PLACES

Search bar with autocomplete function:



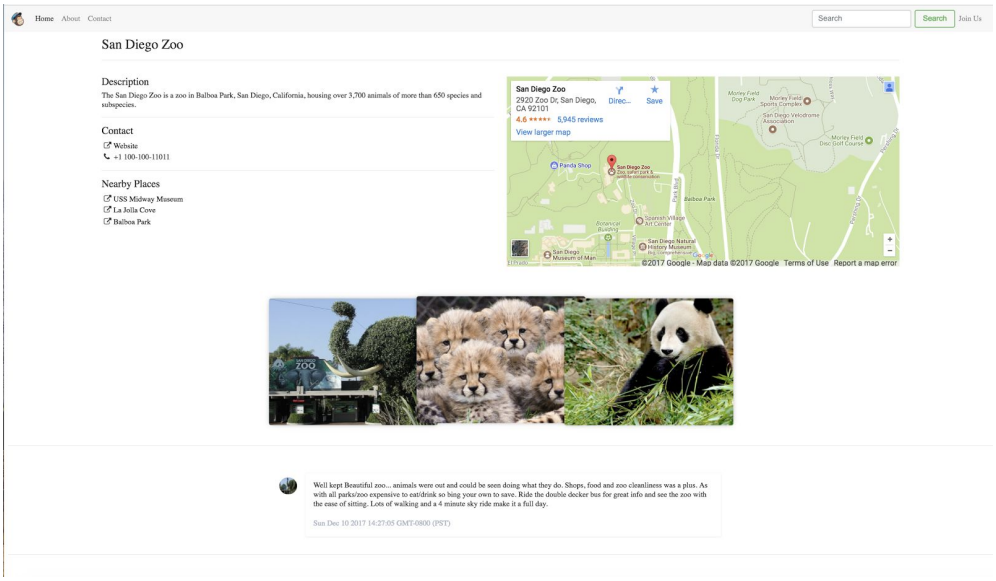
Search result page with fuzzy search:



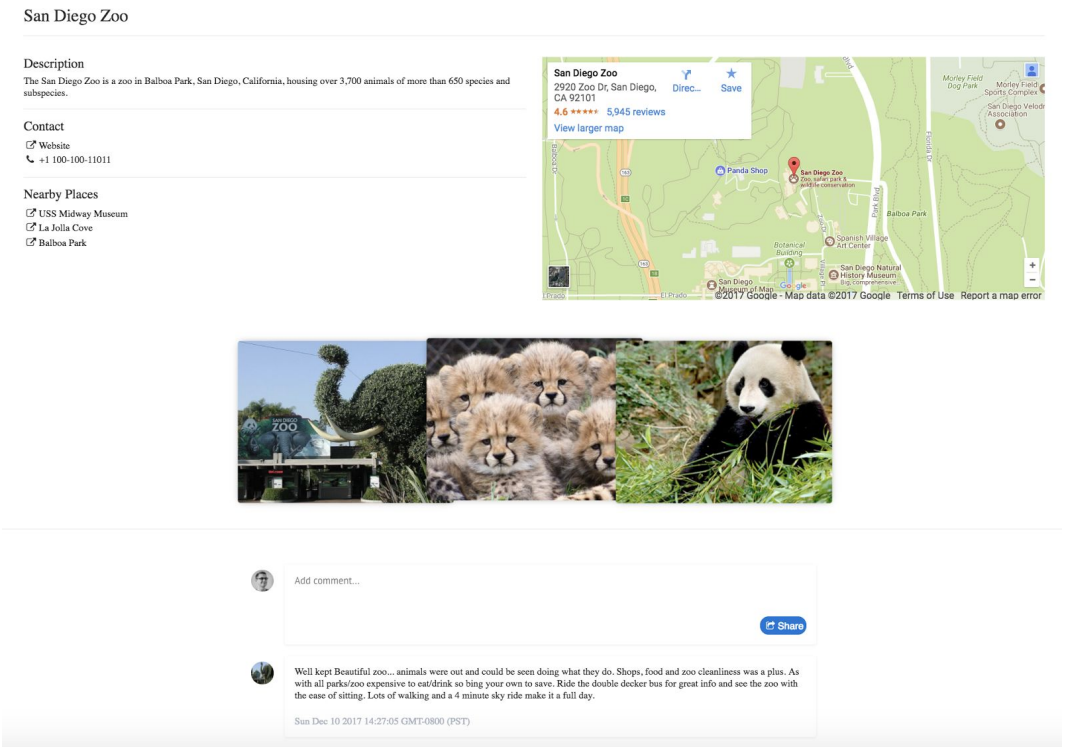
SHARE YOUR REVIEWS IN FORM OF COMMENTS

To be able to write review for place, you have to login first.

Page before login:



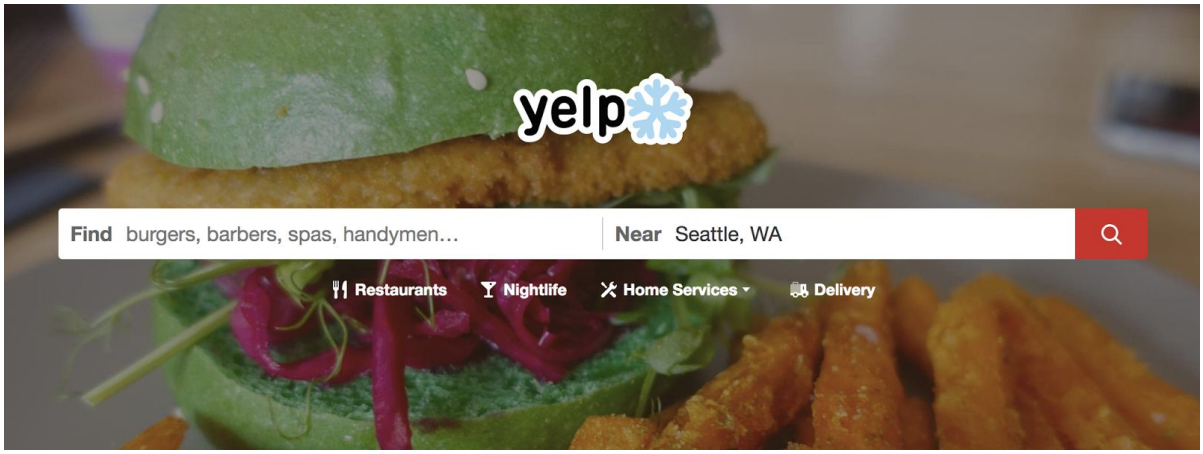
Page after successful login:



Page after write review:

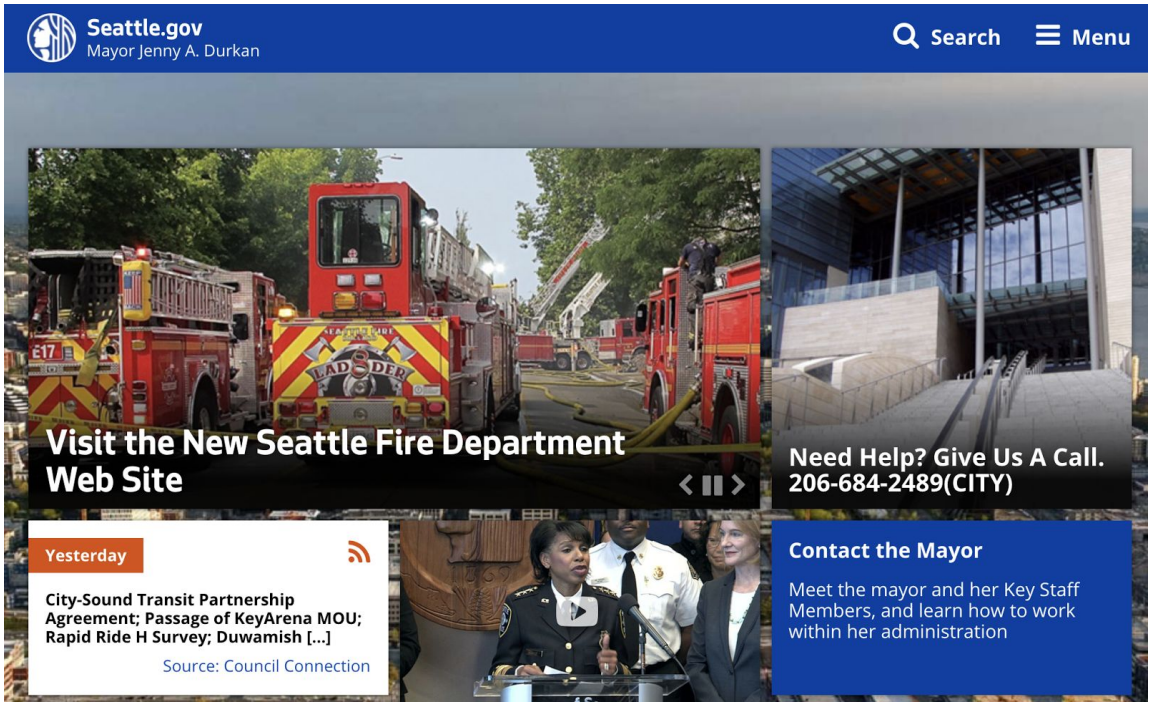
DETAILED INFORMATION ABOUT CITIES

1. Restaurants (redirect to yelp, looking for best local restaurants)



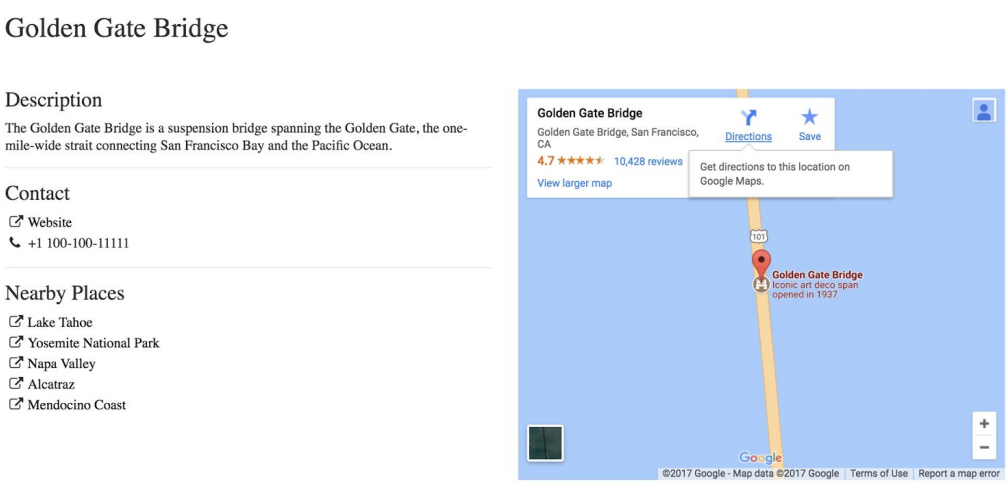
2. Transportation (redirect to official local transportation website)

3. Public Services (redirect to official local government website)



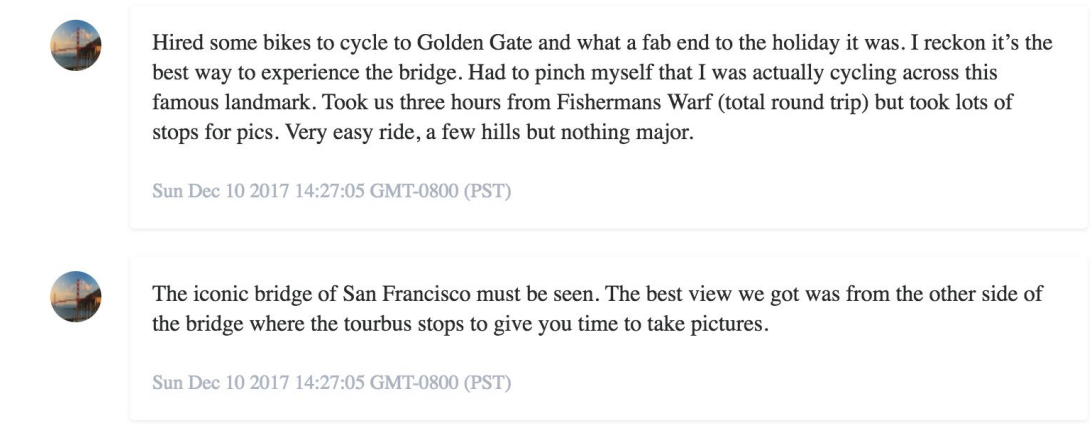
4. Weather (redirect to yahoo weather that display local weather)

5. Get Directions to the place using Google Maps

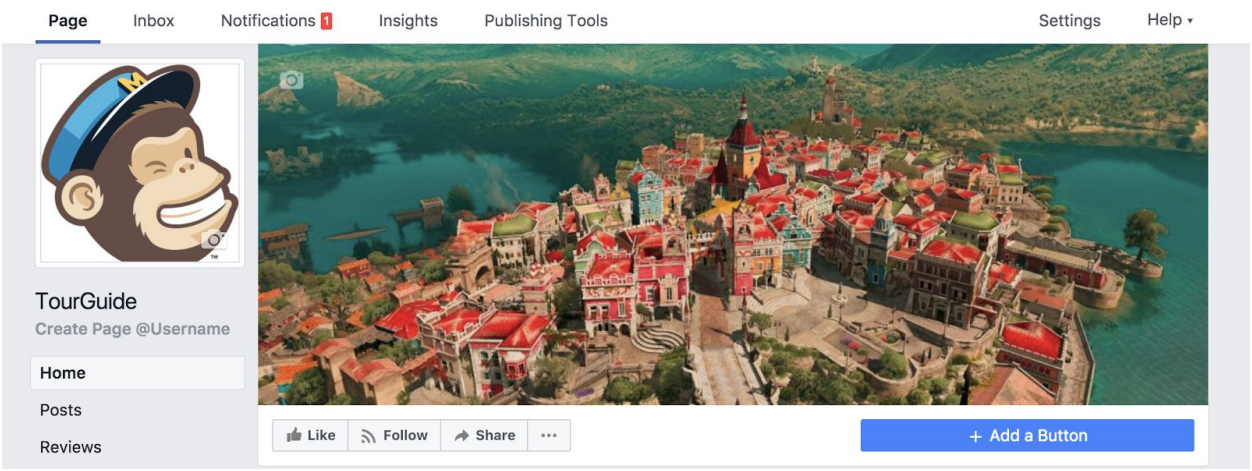


We are very accessible:

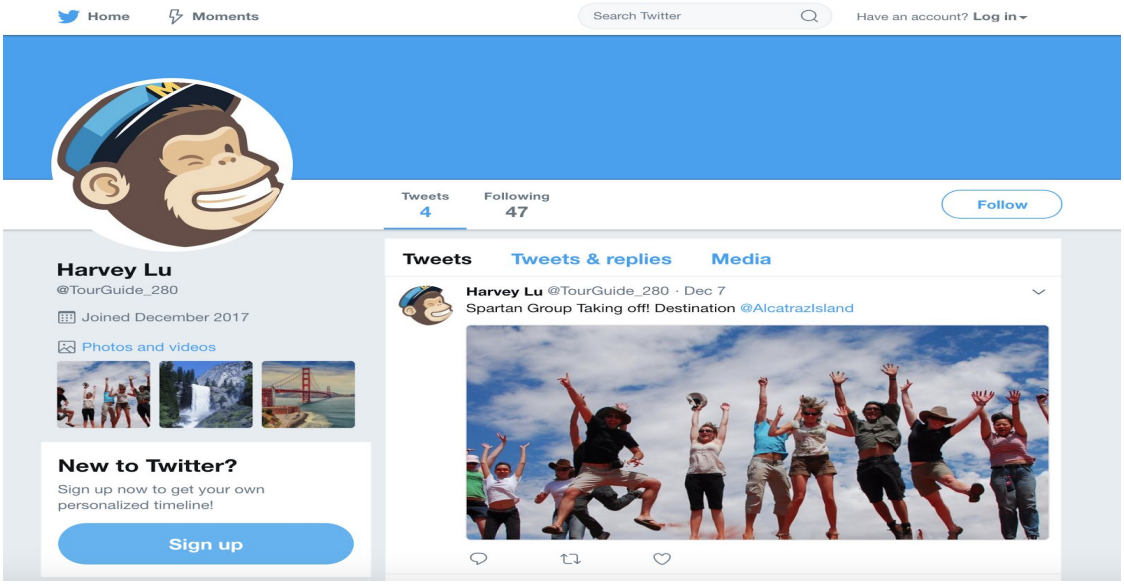
At the end of each page, we put links which will redirect our users to our facebook and twitter official account. Where you can make a post or see all the post other users send.



Our official facebook page:

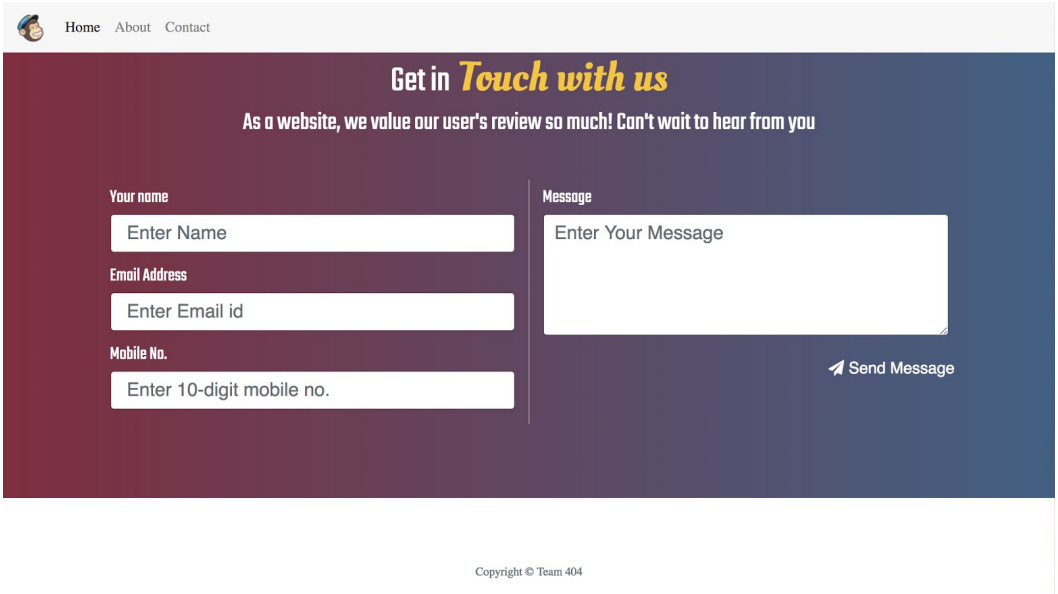


Our official twitter account:



Our contact page:

You can also get in touch with us via our contact page.



TECHNOLOGIES USED:

The technologies used and the reasons for choosing them are listed below:

Why HTML5?

- HTML5 provides us with a cleaner code which is easy to understand and write.
- Using HTML5, we can take advantage of Smart Local Storage and store the data in browser instead of sending it to server for every request. This makes the application run faster.
- Better Interactions, HTML5 not only displays the content but also allows a user to interact with it, which is the demand of user centric applications like ours.
- Legacy/Cross Browser Support: HTML5 can work on both old and new browser and also on all browsers and hence was an obvious choice for our application.

Why CSS?

- The CSS code is lightweight and hence an ideal choice for rendering templates and building applications, since it doesn't affect the response time.
- CSS is platform independent, and hence we can have the same styling for our web pages on all the platforms.
- CSS is Cross Browser Compatible and hence we can smoothly render our web pages using CSS styling on all the browsers.
- CSS provides us with a lot of different formatting options and hence an obvious choice for styling our web pages.

Why Jade?

- Jade allows for easy and rapid development of the web pages templates.
- It reduces tedious markup as in case of HTML and hence a smooth development experience.
- The code consistency feature in jade provides us with a neat and easy to understand code, which is easier to work with.

Why JavaScript?

- The Input Validation that JavaScript provides at the client side is very useful in building web application like ours which take a lot of data as input from the user.
- Usage of JavaScript speeds up the execution of code since most of the code is executed at client side, with less requests to server.
- JavaScript is a simple and easy to use language.
- Javascript is light weight , so fast request and response and a hence faster web

application.

Why ExpressJS and NodeJS?

- To set up middlewares to respond to HTTP requests: using expressJS and NOdeJS it was easy to write routing logic for the different web pages and web application services.
- For routing requests: To route and map incoming HTTP requests, since Express and Node provide functionalities to handle the routing
- For dynamically rendering HTML pages
- Session Management
- Authentication: We have used Passport.js for implementing authentication functionality in our application
  Passport is authentication middleware for Node.
  We use it to authenticate request from client side - be specific, the user login authentication.

Why MongoDB?

- Schema less database
- Powerful and deep query availability
- Quick Iteration over data
- Returns cursor instead of dataset on querying which improves performance.

DATA SOURCES & STORAGE

We have stored our data online in mlab, which is a leading Database-as-a-service for MongoDB.
There are two collections Users and Places in our database.
The Users collection is populated when a user interacts with the application, like signup, review a place using comments, etc.
The Places collection has been populated by us using the information available on internet and also using the inputs of users about a place.

For incorporating MongoDB in our web application, we have used “mongoose”, which is a MongoDB object modeling tool designed to work in an asynchronous environment.

It can be installed by executing the following command in the root directory of our application in the command window.

“npm install mongoose”

Connection to database using mongoose has been implemented as shown in the following code snippet:

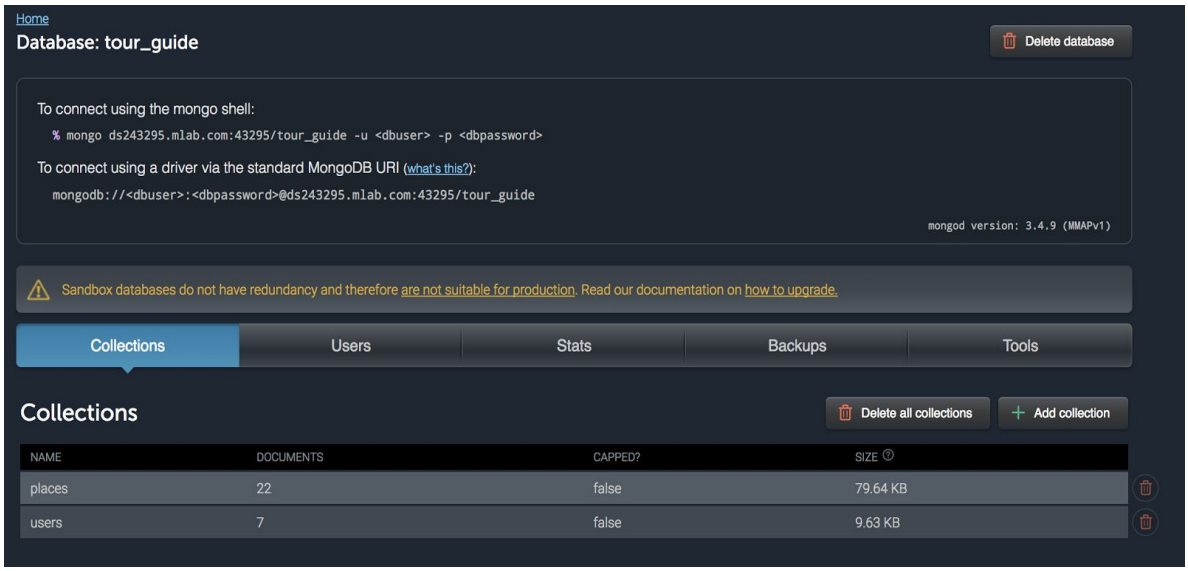
```
var dbConfig = require('./db');
var mongoose = require('mongoose');
module.exports = {
  'url' : 'mongodb://dbuser:dbpassword@ds243295.mlab.com:43295/tour_guide'
}
```

Online Database Host:

We use mlab as online mongodb host. The reason for using it so that the whole team can access the data at a central location and also the data is not lost or corrupted.

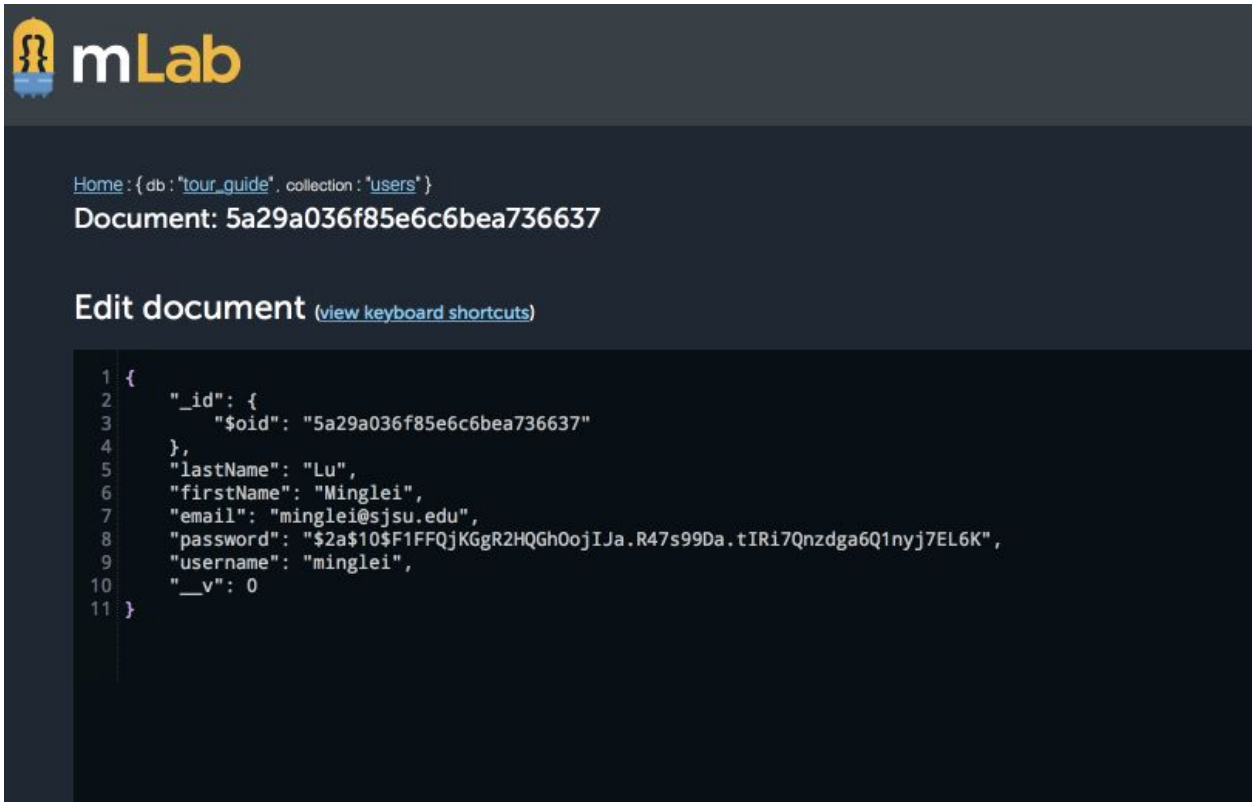
Online Database:

The screenshot below shows our database as stored online in the mlab service.



- **Password Encryption**

- The password entered by users will be encrypted and then stored in the database. It's the functionality of middlewares of express, passport and bcrypt-nodejs, that help to validate, authenticate and encrypt all the account information to provide better security for our application.



- The object document mapping for our database has been described below.

- **User Schema**

- This schema consists of details of a registered user. The user details that are stored are: Username, Password(encrypted), email id, first name, last name.

```
var mongoose = require('mongoose');

module.exports = mongoose.model('User',{
    id: String,
    username: String,
    password: String,
    email: String,
    firstName: String,
    lastName: String
});
```

Places Schema

This schema consists of details of the places to be visited. The details that are stored are the variables on the left hand side as shown in the screenshot below:

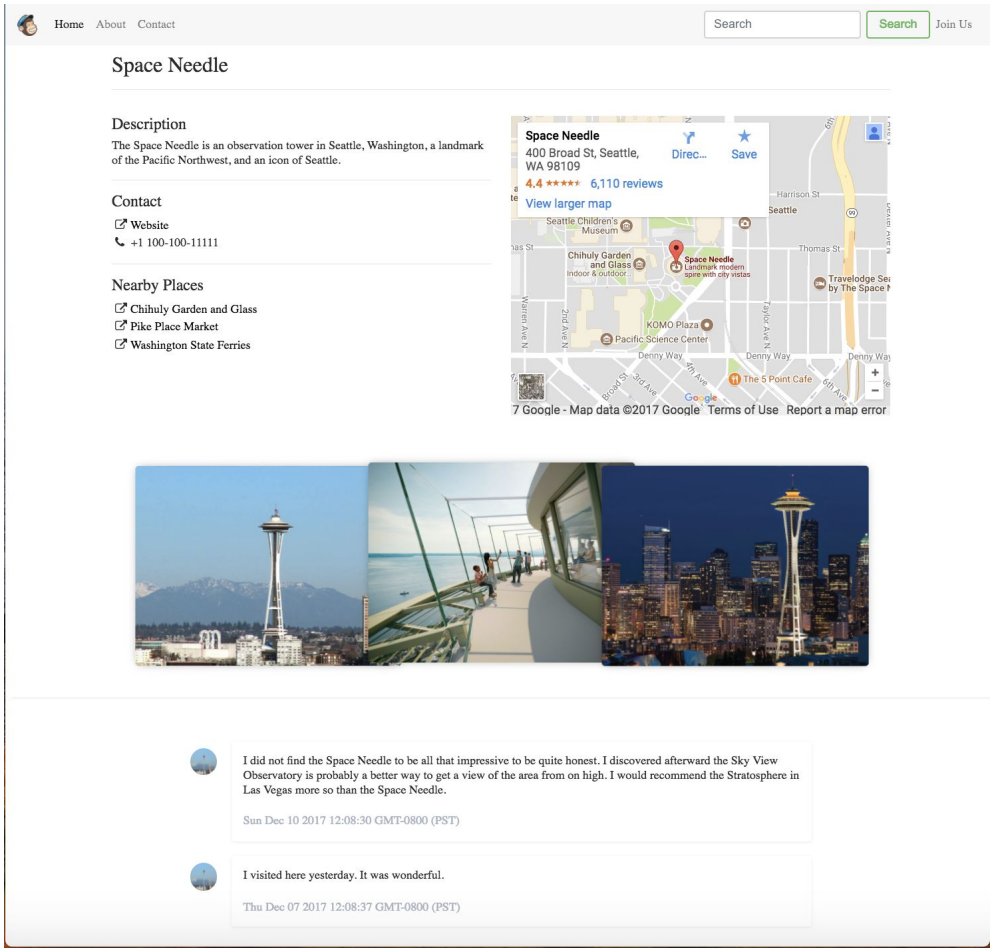
```
var Place = mongoose.model('Place', {
    name: String,
    url: String,
    keywords: [String],
    info: [{
        type: String
    }],
    website: [String],
    description: String,
    images: [{
        type: String
    }],
    scenes: [{
        name: String,
        url: String
    }],
    reviews: [
        {
            userName: String,
            content: String,
            createDate: {type: Date, default: Date.now()},
            updateDate: {type: Date, default: Date.now()}
        }
    ]
});
```

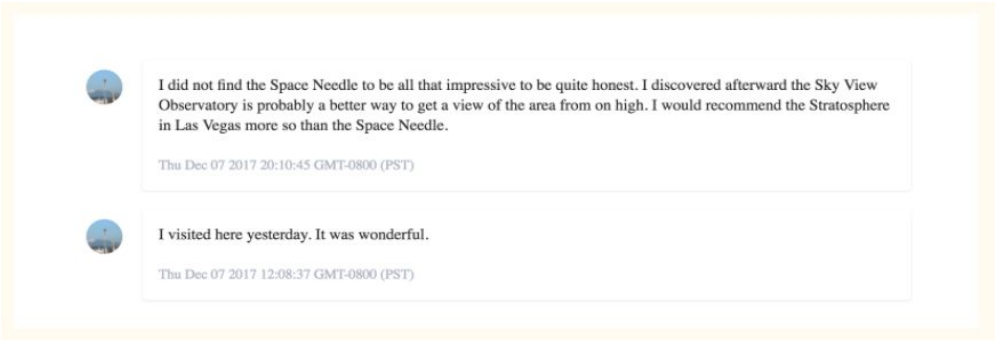
LOOK & FEEL

GESTALT PRINCIPLES

1. PROXIMITY

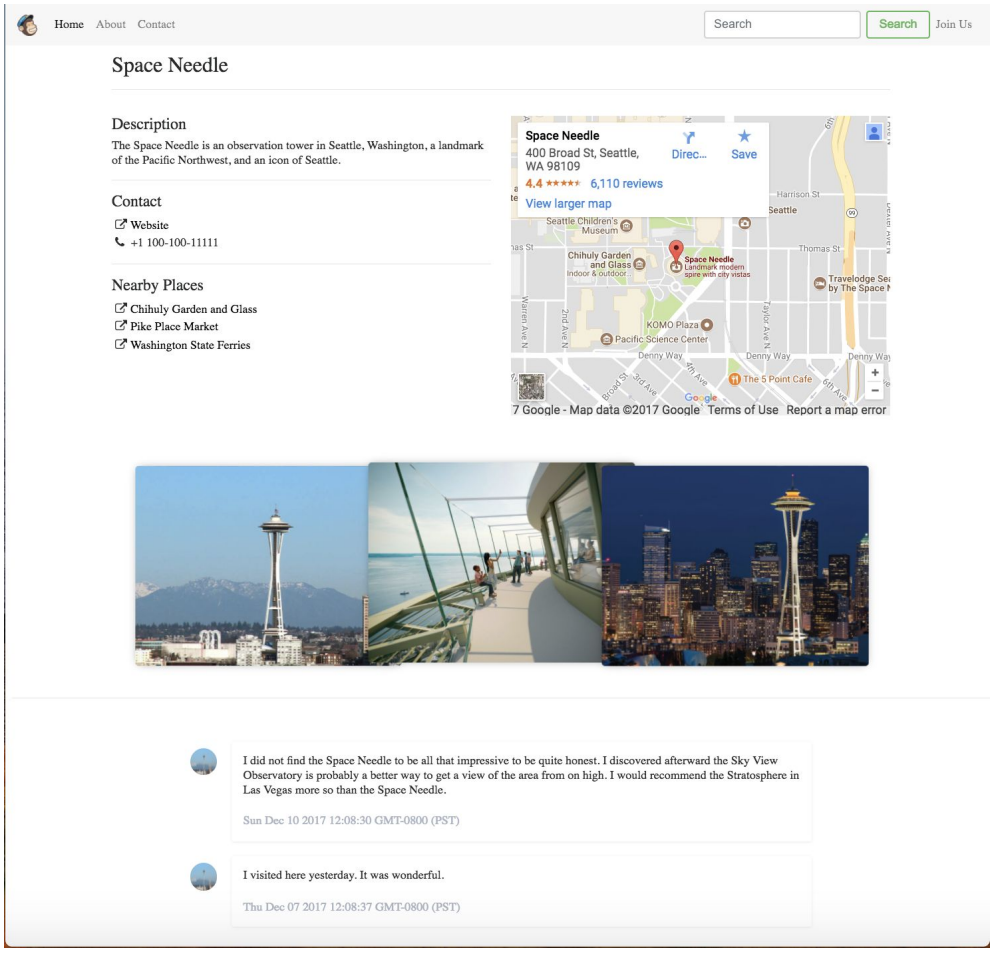
Viewers will associate together items that are placed close together.
This can be seen in the screenshot below where all the user comments can be seen close and next to each other.

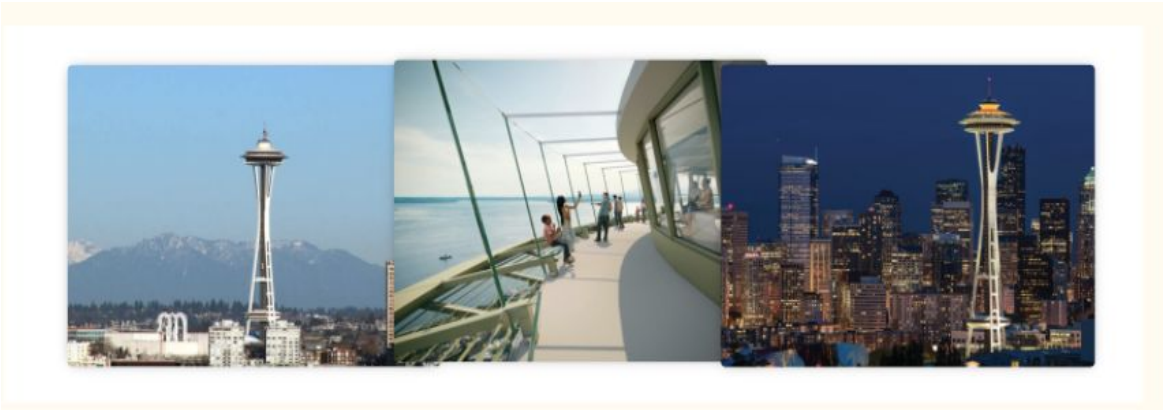




2. SIMILARITY

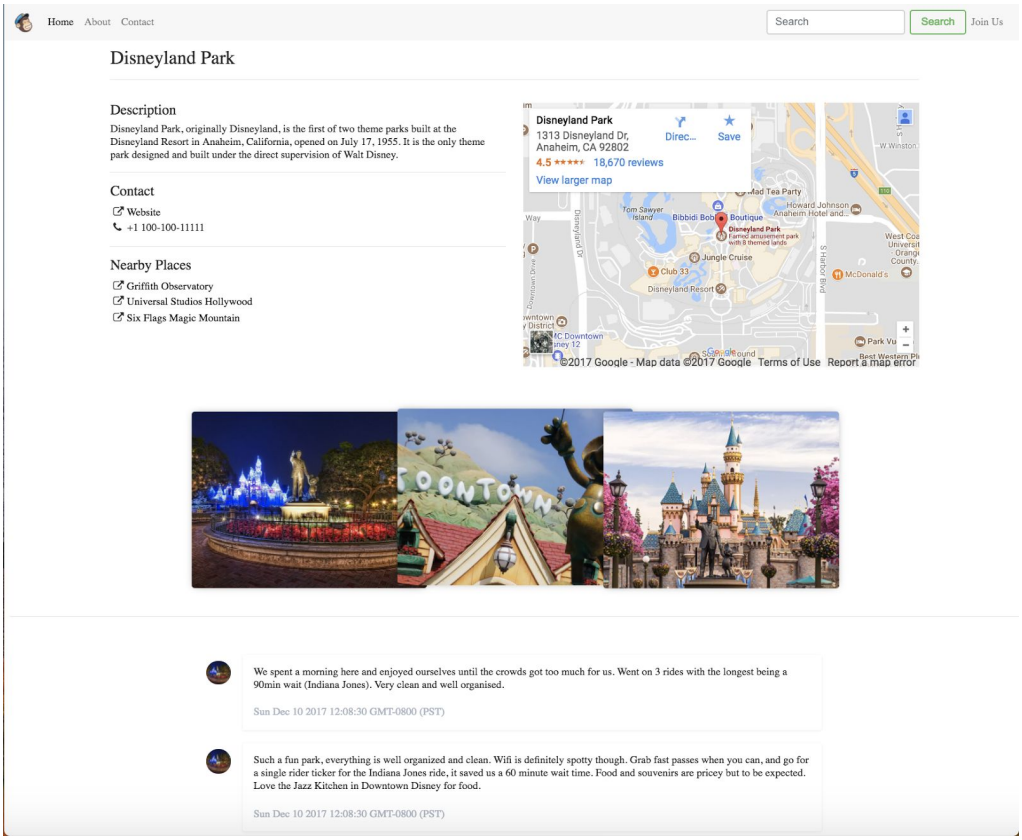
Viewers will associate two things that have the same shape, size, orientation, etc. On the web page for a place there is a section that contains images of that place. That section that looks like a gallery where pictures are put together, in the same shape and size.

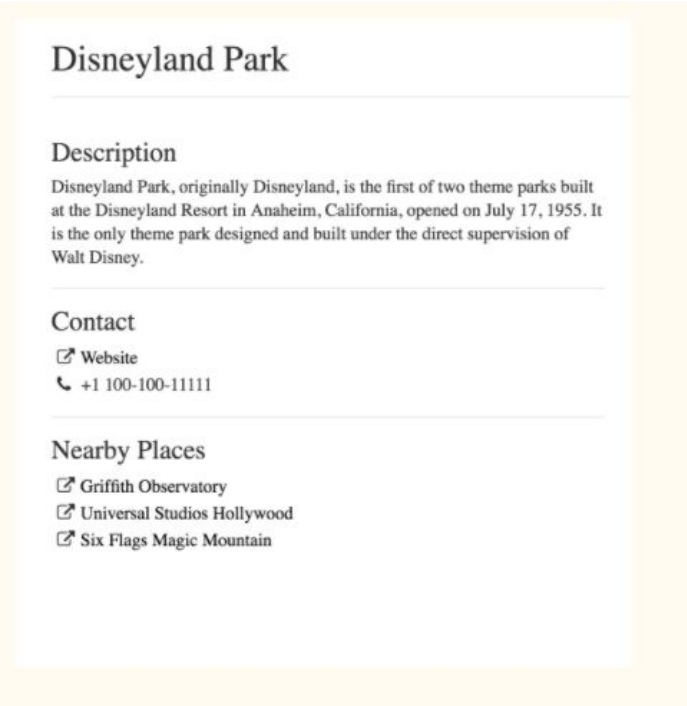




3. CONTINUITY

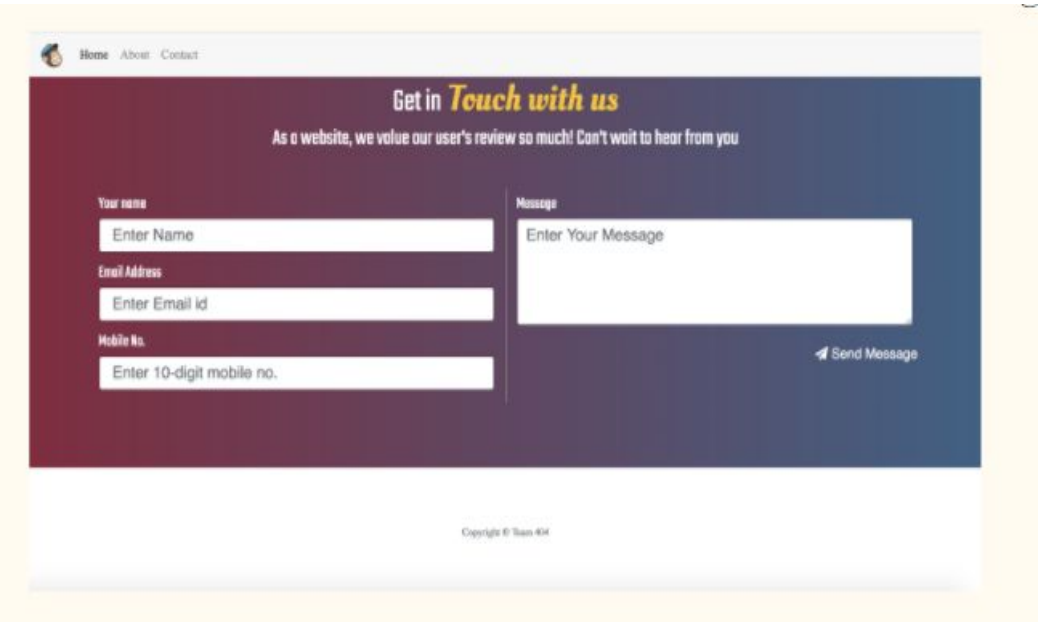
Viewers' eyes want to see continuous lines and curves formed by the alignment of items. The web pages that display that information about a place have information displayed next to each other.





4. CLOSURE

Viewers' eyes want to see implicit simple closed forms such as rectangles. The implementation of this principle can be seen in our Contact page.There is a closed form in our contact page, which makes the key attributes of the form stand out.

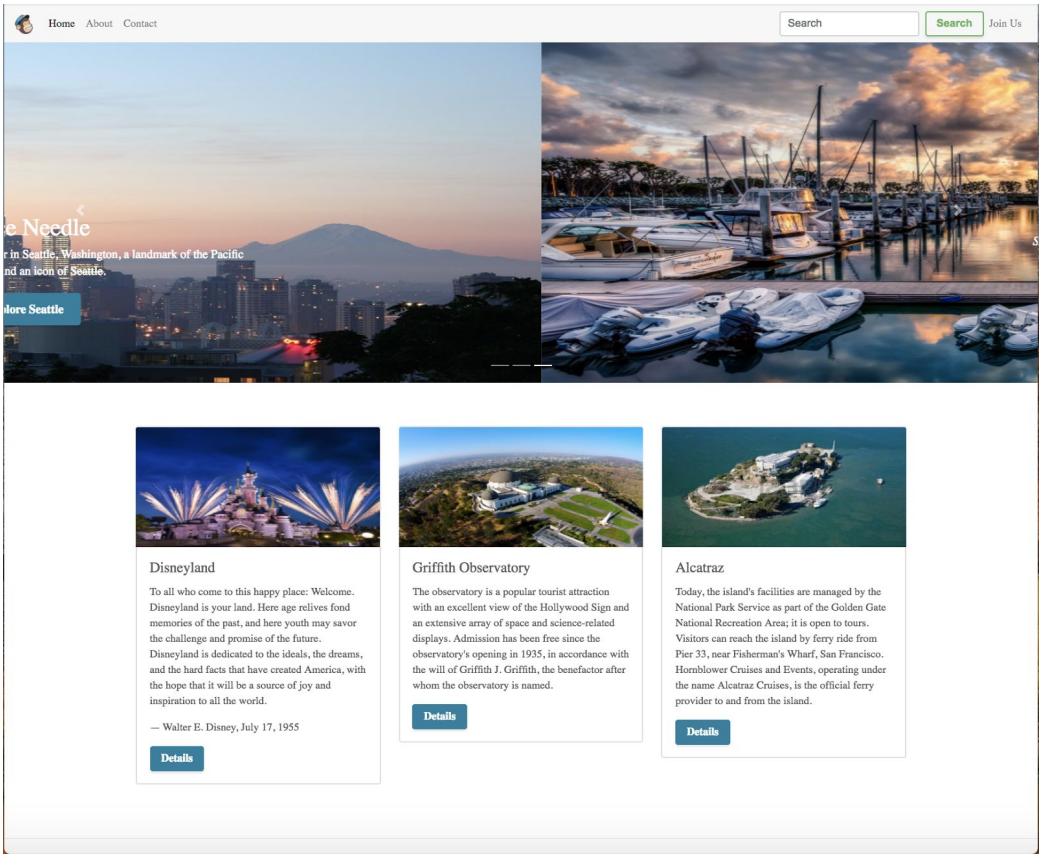


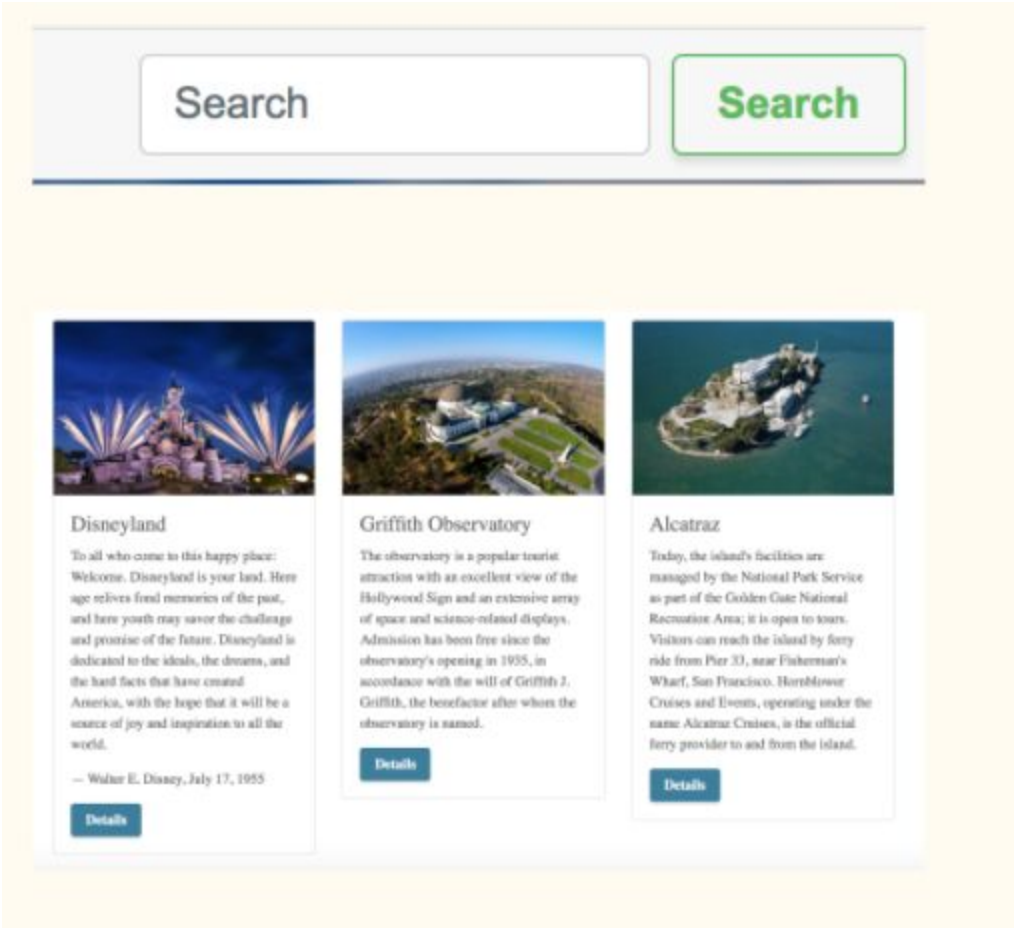
UI DESIGN PATTERNS

The UI Design Patterns that have been implemented in our web application have been listed below along with figures that show the implementation:

1. FEATURE SEARCH & BROWSE

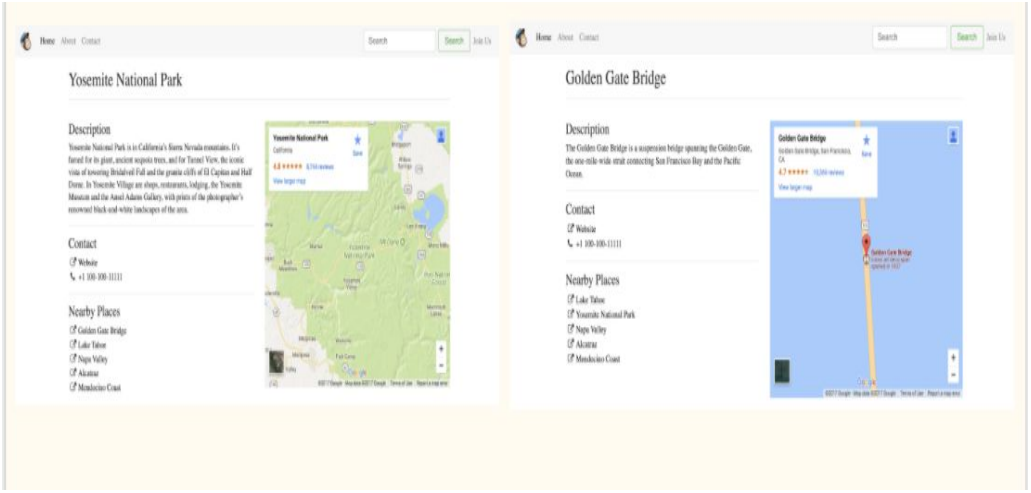
We use search bar for the search function We also list featured items on the homepage to provide other options for the users.





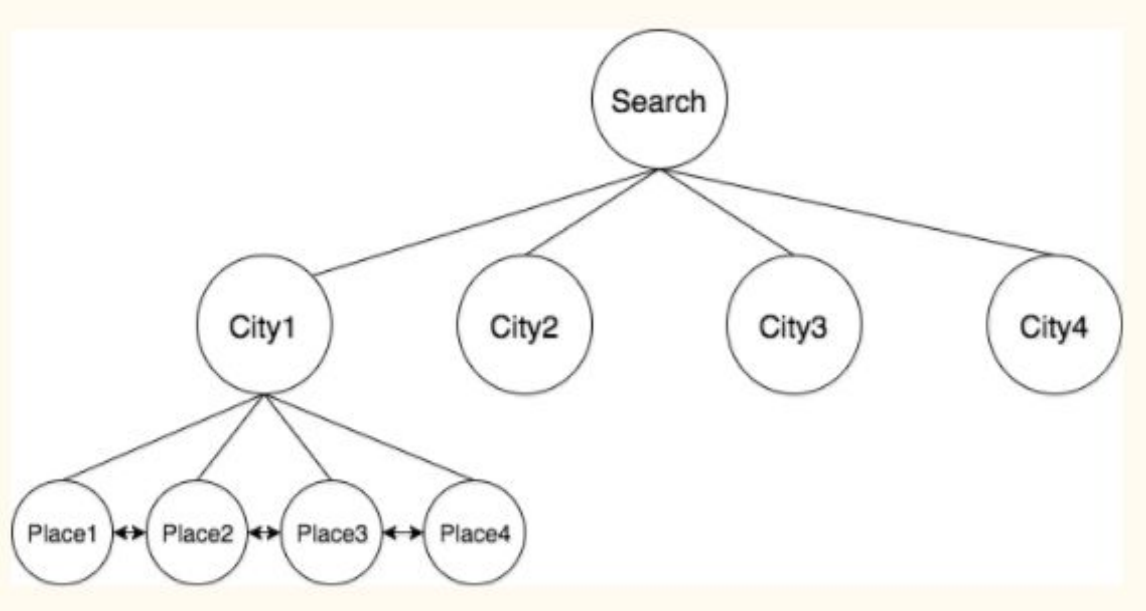
2. VISUAL FRAMEWORK

We use the same layout for the similar webpages, which makes it looks like they belong to each other.



3. MULTILEVEL

We follow multilevel navigation design pattern. The first level is search page. Second is all the cities in our database. Third is places which are connected to each other.



4. TILED SECTIONS

Tiled sections layout is visually separate section of content and each section has a strong title. We implement such design pattern in our homepage to introduce our website's basic function.

Share your Travel Experiences

Have you visited a place, if yes then share your reviews about the places you visited. Help your fellow spartans!! Let them know how to plan the trip? What are the famous spots to visit? Where to eat? How to reach there?

Share your memories

Can't describe the beauty in words! But want to share your experience? No worries, we have got it covered for you. Share the pcitures of places you want to visit. Let others know about the beauty they are yet to explore!

Gather Information for your next trip

Planning to visit a new place, but so much information on the Internet confuses you. Take help of your fellow spartans. They have shared their experiences and reviews which will help you plan your next trip.