UNIBLOCKS

A web application to search apartments nearby universities

Vinay Janardhanan
CMPE, San Jose State University
San Jose, California
Vinay.janardhanan@sjsu.edu

Apurva Pradeep Pawar
CMPE, San Jose State University
San Jose, California
Apurvapradeep.pawar@sjsu.edu

Christopher Pereira
CMPE, San Jose State University
San Jose, California
Christopher.pereira@sjsu.edu

Akshay Bhasme
CMPE, San Jose State University
San Jose, California
Akshay.bhasme@sjsu.edu

***Abstract*—Instructions regarding the basic guidelines for preparing the web application named "uniblocks.me" for getting nearby apartments for the university searched are presented. This document contains information of the usage of the web application, its features and various other terminologies implemented. Sections are also devoted to the preparation of acknowledgement, references and authors.**

Keywords: distance, maps, transit, apartment

I. Introduction

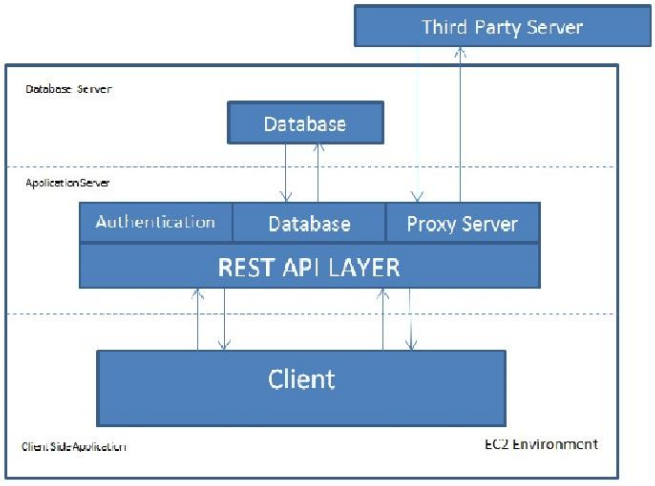
This document provides the details of finding and listing apartments nearby based on university location with the help of google maps. A student searches for the university from the search bar provided and selects an university. Based on the student's selection, a set of apartment nearby the university is displayed in format of a textual view and markers on maps. Students are provided with a list of transit stops nearby when clicking on the specific apartment student selects. Every specific apartment gives details about address(location), phone, website url, walkscore and listings(information specific to the apartment) which student can choose according to their necessity.

II. Architecture

This project was built using MEAN stack (Mongodb, ExpressJS, AngularJS, NodeJS).

A) Database Server

The database that the project uses is MongoDB which is a NOSQL database. User management and role details are saved inside this database. User details from facebook as well as native users are saved within the server. The database also saves and retrieves postings of an apartment from/to a user. The database also maintains a transit dataset which has information about all the transit stop around a particular university.



B) Application Server

The project uses NodeJS as the application server. The server handles the link between the user interface and the backend server. It handles the routing of request to the correct api within the servers. There is a local server as well as a proxy server. So when a call is made this layer redirects to the appropriate server to fetch the data. There are several api's open to fetch data such as google search api, neighborhood api, transit score and walk score api, renters api and an api to access information within the database that we maintain locally.

C) Client Side Application

Client calls the different api's that are available in the application server and displays that data to the user.

Several calls are made via the client to retrieve and save details to the server. The data is displayed in a simple yet elegant way to entice the user. The client calls for data from the server asynchronously using the api's provided by the server and displays it to the user dynamically without loading the entire page.

III. FUTURE ENHANCEMENT

There are many ways to move ahead with this project. One of them is to add a more algorithms to mark apartments within a particular range based on the transit and walk score details. A machine learning concept or sentiment analysis could also be added to search for apartments based on the person's likes and dislikes, after the user has signed in.

There are many possible routes in which this project could be taken, including the possibility of making money by including postings and ads of a user. As implementation of facebook authentication, Google+ and twitter authentication be used to enhance our product.

IV. DISTANCE CALCULATION

Here we are using 'Haversine' formula to calculate the great circle distance between two points

Haversine formula:

$a = \sin^2(\Delta\varphi/2) + \cos\varphi1 \cdot \cos\varphi2 \cdot \sin^2(\Delta\lambda/2)$

$c = 2 \cdot \text{atan2}(\sqrt{a}, \sqrt{(1-a)})$

$d = R \cdot c$

where φ is latitude, λ is longitude, R is earth's radius (mean radius=6,371km);

note that angles need to be in radians to pass to trig functions!

Javascript code:

```
var R = 6371; // km
var φ1 = lat1.toRadians();
var φ2 = lat2.toRadians();
var Δφ = (lat2-lat1).toRadians();
var Δλ = (lon2-lon1).toRadians();

var a = Math.sin(Δφ/2) * Math.sin(Δφ/2) +
Math.cos(φ1) * Math.cos(φ2) *
Math.sin(Δλ/2) * Math.sin(Δλ/2);
var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));

var d = R * c;
```

V. FUTURE ENHANCEMENT

There are many ways to move ahead with this project. One of them is to add a more algorithms to mark apartments within a particular range based on the transit and walk score details. A machine learning concept or sentiment analysis could also be added to search for apartments based on the person's likes and dislikes, after the user has signed in.

There are many possible routes in which this project could be taken, including the possibility of making money by including postings and ads of a user.

VI. CONCLUSION

The main motive of our project was to help students feel safer while moving to a new country for studies, without the overhead of finding a place to stay. After thoroughly understanding the users requirements and difficulties, the project that we built provides a simple at the same time effective way to find apartments within a university. It saves time and helps visualize the location and distance of an apartment from a university. It also helps students find roommates like them and get contact information about them.

VII. CONTRIBUTIONS

1. Christopher Pereira:

Responsible for user and role management in the system. Provided authentication for facebook as well as local logins using PassportJS. Included server side APIs to login and access user details. Also included a hash algorithm to hide user passwords.

2. Vinay Janardhanan:

Responsible for Algorithm and Design. Calculated distance using the location co-ordinates and displayed google maps markers of different colors using the nearest distance. Displayed transit data according to the appartment selection. Using walkscore api, returned the walkscore generated according to university location.

3. Apurva Pradeep Pawar:

Responsible for modelling and cleaning the data sets received. Also responsible for managing the Database and Application. Developed REST APIs and a Proxy server on the Application server. Deployed and automated the application on the cloud (Bluemix and EC2) and was responsible for migration from one service to another. Designed the logo for the application.

4. Akshay Bhasme:

Responsible for architecting and designing the front end of Uniblocks Developed the app using AngularJS, Bootstrap and Google Maps API. Also contributed to how the Rest sever APIs should work on the server. Developed and

implemented the “Sticky” listing feature to make Uniblocks a potentially sellable product.

REFERENCES

[1] Information on the Walk Score https://www.walkscore.com/professional/walk-score-apis.php

[2] Google Maps to render map https://developers.google.com/maps/

[3] Rental dataset in San Francisco Bay Area http://rentrent.org/RENT/API/index.html

[4] VTA data set for San Francisco Bay Area VTA https://data.vta.org/

[5] SFMTA data set http://www.sfmta.com/about-sfmta/reports/gtfs-transit-data

[6] ACTransit http://www.actransit.org/planning-focus/data-resource-center/

VIII. PROJECT DETAILS

1. GitHub URL : https://github.com/theapurvapawar/cmpe272-team4.git

2. Public URL : http://uniblocks.me/

3. Cloud URL: http://ec2-54-183-68-182.us-west-1.compute.amazonaws.com:3000/#/

4. Youtube Introduction URL: https://www.youtube.com/watch?v=plydqdJXZXA