

**CAB432 Cloud Computing**

**Mashup/Docker Project**

**Submitted by**

**Thanat Chokwijitkul n9234900**

**Tutor: Malcolm Corney**

**Tutorial: Tuesday 4 pm – 6 pm**

**Queensland University of Technology**

**14th September 2015**

# **Introduction**

Place Finder provides users with visual and interactive way of exploring nearby places. The application detects locations of the web visitors and displays some basic information, including an interactive map and weather information in the area. Users can discover the nearby places using some provided buttons labeled with place types/categories specified by the Google Places API. However, other place types/categories still can be searched via a search box.

The primary purpose of this API mashup is to provide users an easy and clear way to explore nearby businesses and services around a specified area using services and data APIs listed as follows:

#### Telize

The Telize API allows users to request a website visitor's geolocation information.

URL: <http://www.telize.com>

#### Open Weather Map

The Open Weather Map API allows users to retrieve the current weather information with various options of parameters, such as city name or geographic coordinates.

URL: <http://openweathermap.org/api>

#### Flickr

The Flickr API allows users to access and retrieve photos from the Flickr photo sharing service.

URL: <https://www.flickr.com/services/api/>

#### Google Maps

The Google Maps JavaScript interface is used to embed Google Maps onto web application.

URL: <https://developers.google.com/maps/?hl=en>

#### Google Places

The Google Places API is a web service that provides information about nearby places within a specified area.

URL: <https://developers.google.com/places/>

#### Foursquare

This API allows access to the Foursquare’s database of location as well as information on venue check in's.

URL: <https://developer.foursquare.com>

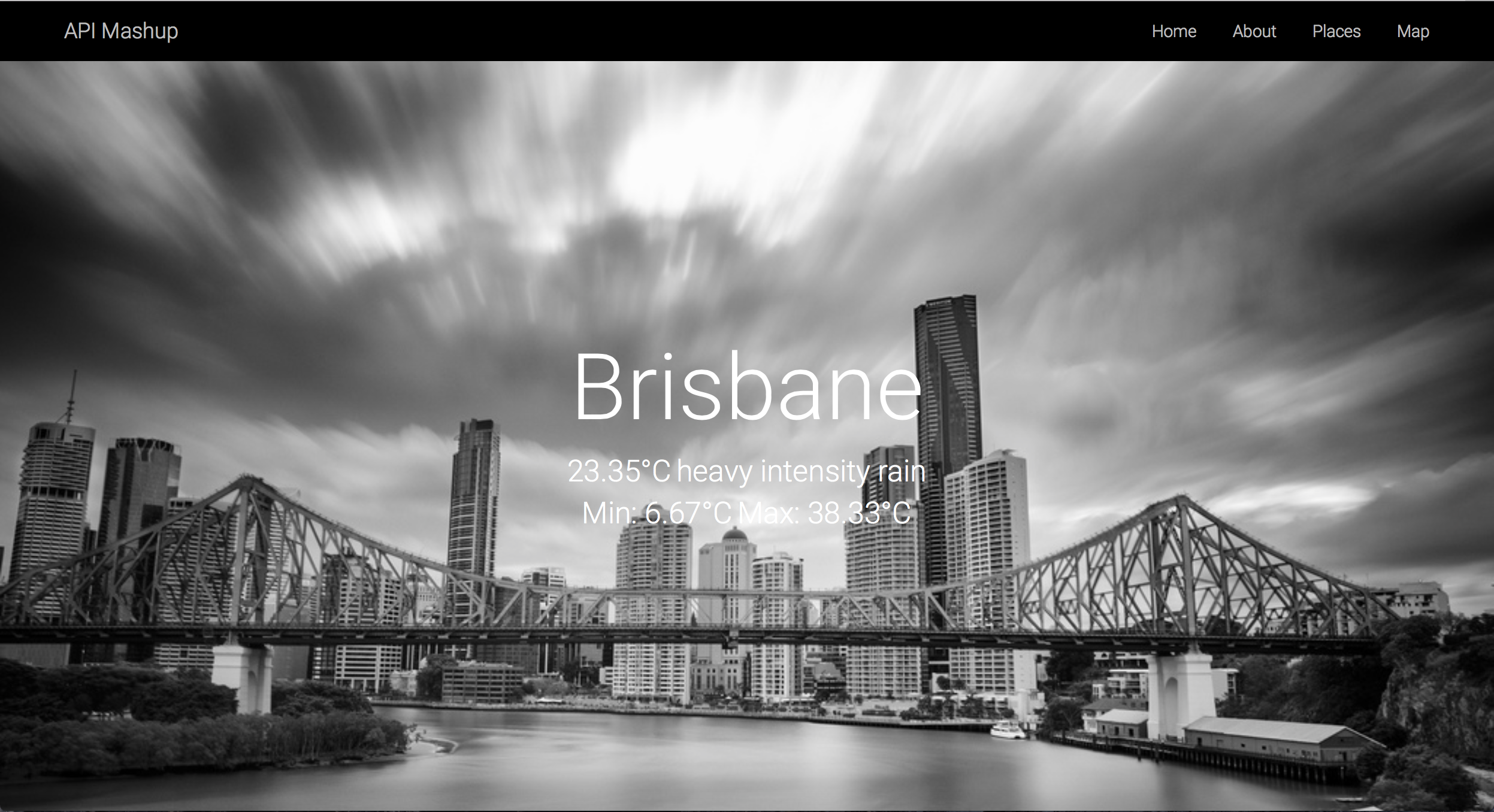
The results displayed on Place Finder are continuously updated by Telize, Open Weather Map, Flickr, Google and Foursquare. By the incorporation of the mentioned APIs, users should be able to find nearby places in a specified area with decent information.

# Mashup Use Cases and Services

This section outlines the use cases supported by the mashup and the service API calls used in the application to satisfy each user story. The term “user” used in each story refers to both tourists and local users within a specified area.

“As a user, I would like the system to be able to detect my location and display the current weather information, such as current temperature and weather description so that I can decide whether I should go out or not.”

When accessing the website, the system will detect where a user is by sending a request to Telize in order to obtain geolocation data (geographic coordinates). Afterwards, the piece of data obtained from Telize will be sent to the Open Weather Map API to get current weather information, and also the Flickr API to get a current location related photo.

**Figure 1**: The screenshot displaying the city name with current weather information.

“As a user, I would like the system to be able to detect my location and display an interactive map so that I can explore the area using that map.”

By using the Google Maps JavaScript interface, an interactive map will be embedded into the webpage. It will automatically ask for the permission to detect and display a user’s current location. If the permission is given, the map should correctly display the user’s location in the Map section.

“As a user, I would like the system to provide buttons labeled with place types/categories, such as restaurant, cafe, bar etc., so that I can find commonly searched businesses and services around my area easily without typing.”

Several buttons labeled with place types/categories specified by the Google Place API are provided in the section, called “Places”. The Places section corporates with the “Map” section. As a result, if a button is clicked, the webpage will automatically scroll down to the Map section and point out all the nearby places according to the specified type/category using markers.

“As a user, I would like to be able to search on the map so that I can find other nearby places using other keywords apart from the provided categories.”

A search box is provided on top of Google Map using the Google Places API JavaScript library. Users can search using any keywords, such as city names, place types/categories, place name etc. Searching via the search box will give some results regardless where users are.

“As a user, I would like the system to display details of each place, such as address, contact number, official website, rating or reviews, so that it can help me decide whether I want to go to that place.”

When searching for nearby places using either the provided buttons or search box, several markers will be pinned down onto the map according to the search results. Each marker contains specific information of a particular place. Therefore, when a marker on the map is clicked, the server will send a request to the Foursquare API to obtain the details of the location pointed by the marker, and then display the data inside the current infowindow. However, if the system cannot get data from Foursquare, it will send another request to get some details at least an address and rating from the Google Places Web Service instead.