SOFTWARE REQUIREMENT SPECIFICATIONS

App: miCity

Authors:
William Briggs
Sushant Chatufale

Version: 1.0

Date: 25/11/2016

Introduction:

miCity will be an interactive app for IOS devices that dynamically tracks and makes suggestions of nearby places to the user for the city they are in. It will have a multitude of features such as:

- Twitter integration
- Visualisation of emotion lexicons through form of word cloud
- Sentiment analysis of Twitter data providing an overall emotion of the city
- Game related to the emotions of the city
- Weather display
- Favourite places with pictures
- Visualisation of current location in the city through Google maps

User Characteristics:

The app will be available to all the users, there is no concrete demographic, although the user is assumed to be familiar with social media and usage of Google maps.

Functional Requirements:

Requirement ID	Requirements	Mandatory /Desirable/Optional
Func_01	App should display a splash screen when on booting	Optional
Func_02	Application should display “Welcome Message and App Info”on the splash screen for 3 seconds	Optional
Func_03	Navigate to the “home screen” page	Mandatory
Func_04	The “home screen” page should display Google map using “Google maps framework”	Mandatory
Func_05	The “home screen” page should use“Google places framework”for the current location of the user	Mandatory
Func_06	The application should point to the current location of the user using pin pointer from the Google maps	Mandatory
Func_07	The “home screen” page should have a“+” button to zoom into the map	Mandatory
Func_08	The “home screen” page should have a“-” button to zoom out of the map	Mandatory
Func_09	The application should extract the current location of the user and display it in the bottom left corner of the view	Mandatory
Func_10	The view page should have a button “Socialize”	Desirable
Func_11	The “Socialize” button should navigate to the “social screen” page	Mandatory
Func_12	Data of 100 tweets should be extracted and classified using sentiment analysis tweet channel as positive, negative or neutral every 80 seconds	Mandatory
Func_13	The twitter data should be extracted using the external“STTwitter” library	Mandatory
Func_14	The sentiment analysis should be done using twitter dataset from www.thinknook.com containing over 1.5 million pre-classified tweets	Mandatory
Func_15	App should update the overall emotion of the city with newly classified Twitter data	Mandatory

Func_16	The “social screen” page should display a word cloud of sentiment lexicons	Mandatory
Func_17	The word cloud should display the terms that express emotion in the Twitter data	Mandatory
Func_18	The words with most occurrences should be bigger in size in the word cloud	Mandatory
Func_19	The cloud should provide a button on top right corner of the social page to pull more Twitter data manually	Desirable
Func_20	The “social screen” page will have 1 button on bottom right which gives the overall emotional status of the city	Mandatory
Func_21	The “social screen” page will have 3 buttons on bottom left for words related to the emotions positive, negative and neutral in the word cloud	Mandatory
Func_22	The 3 buttons or click on any word from the word cloud should navigate to the “twitter emotion” page	Mandatory
Func_23	The “twitter emotion” page should display a grid of usernames and the subsequent tweets with an option to reply to a particular tweet	Mandatory
Func_24	The “twitter emotion” page should have a “Camera” button to take a photo and send it with the tweet	Mandatory
Func_25	The “twitter emotion”page should display all tweets containing positive emotion if “positive” button is pressed	Mandatory
Func_26	The “twitter emotion”page should display all tweets containing negative emotion if “negative” button is pressed	Mandatory
Func_27	The “twitter emotion”page should display all tweets containing no/neutral emotion if “neutral” button is pressed	Mandatory
Func_28	The “twitter emotion”page should display all tweets containing the particular word when that word is clicked in the word cloud on “social screen” page	Desirable
Func_29	The “twitter emotion” page will display an animation of Twitter logo on the bottom left side	Optional
Func_30	The application should extract weather information of the city from “OpenWeatherMap” API using latitude and longitude co-ordinates	Mandatory
Func_31	The main view page should display weather information in the bottom left side of the screen	Mandatory
Func_32	The weather information should be displayed by a symbol, weather description and current temperature of the city	Mandatory
Func_33	The “home screen” page should have a “Nearby” button on the bottom right side	Mandatory
Func_34	The “Nearby” button should bring up a pop-up of all the nearby places	Mandatory
Func_35	The app should navigate to the “Add To Favorites” page upon clicking on any nearby place of the pop-up	Mandatory
Func_36	The “Add To Favorites” page should carry some core information inherited from the Google places framework and also allow editing of that data	Mandatory
Func_37	The “Add To Favorites” page should have a “Browse Gallery” button to associate a picture with a place	Desirable
Func_38	The “Add To Favorites” page should have a “Camera” button to associate a picture with a place	Desirable
Func_39	Upon associating image with place, tapping the image should enlarge it encompassing the screen	Mandatory

Func_40	The “Add To Favorites” page should have a “Save” button to save the data and the pictures to an “SQLite” database	Mandatory
Func_41	The “Add To Favorites” page should have a “Pin To Map” button to pin the location to the map on the home screen	Optional
Func_42	The “home screen” page should have a “Fave Places” button on the top left side	Mandatory
Func_43	The app should navigate the “Place Editor” page on pressing the “Fave Places” button	Mandatory
Func_44	The “Place Editor” page should have a “Delete” option to remove the place from the list	Mandatory
Func_45	The “Place Editor” page should navigate to the “Add To Favorites” page upon clicking on any of the places	Mandatory
Func_46	The “home screen” page will have 1 button on bottom right which gives the overall emotional status of the city	Desirable
Func_47	The “home screen” page will have “Game” button on bottom right of the screen	Desirable
Func_48	The app should navigate to the “Game” screen when the “Game” button is pressed	Desirable
Func_49	The “Game” screen should have a “Start” button to start the game	Desirable
Func_50	The “Game” screen should have a “Pause” button to pause the game	Desirable
Func_51	The “Game” screen should have a “Wave” label to display the current level of the game	Desirable
Func_52	The “Game” screen should have a “Scores” button	Desirable
Func_53	The “Scores” button should navigate to the “Display Scores” page	Desirable
Func_54	The “Display Scores” page displays table of player name, score associated with it, the wave and emotion of the city at the time of playing.	Desirable
Func_55	The complexity of the game should be affected by the emotion of the city which can be viewed by pressing the “Emotional Status” icon	Desirable
Func_56	The game should have touch controls for navigation.	Desirable
Func_57	The game should feature level based progression, power-ups, increasing difficulty, a scoring system and some form of enemies.	Desirable
Func_58	The app should provide a option to submit a score to the SQLite database upon game end	Desirable

<u>Non-Functional Requirements:</u>

Requirement ID	**Requirements**	
NonFunc_01	The twitter data should be extracted asynchronously to maintain fluid UI	Efficiency
NonFunc_02	The weather data should be extracted asynchronously to maintain fluid UI	Efficiency

NonFunc_03	The game should be well tailored towards touch based controls	Usability
NonFunc_04	The database queries and extraction should be fast	Efficiency
NonFunc_05	The app should be able to reply to a tweet without delay	Efficiency
NonFunc_06	There should be security measures in place to ensure we don't go beyond tweet threshold limit	Robustness

Dependencies and Assumptions:

Dependency	Usage
STTwitter library	The STTwitter library is used to extract twitter data for sentiment analysis
OpenWeatherMap.org	The weather information and weather icons from this site are used for displaying weather in the app
Apple Social API	The Apple Social API is used for more 'high-level' twitter access in the form of being able to respond to tweets using the user account associated with the device
Google Maps	The home screen will feature a map showing the users currently location
Google Places	The nearby button will use the google place api to show places around the user
Camera	The camera is needed to associate images with a favorite place.
Assumption	The app was modeled using an iPad Air device. Hence this should be the preferable device to run the app
User twitter account	User Twitter account required to reply to a tweet

Game – miCity Racer

In order to show the influence of the emotion of the city in a more interactive manner we decided to integrate a game into our application. The basis for the game is; the happier the city, the more points you can score while playing the game.

The game is a simple top-down racing game where you must avoid traffic and collect stars for points. Every 30 seconds the wave is increased and the game becomes more difficult.

- A player can hit 5 pieces of traffic before the car explodes and they are given the game over screen
- The road features obstacles that can make the players controls become distorted
- The game features various power-ups. Some for repairing the status of the car, others for making it easier to dodge around traffic.

Upon crashing the player will be asked to submit a score along with the current emotion of the city.