CS411 - Team 2: BU Mapper

Team Assignment 4/5: Test plan and Database documentation Deliverables: Upload your documentation to your github repo

Section I: Data Documentation

There is one part to the assignment for Section I:

1. Add to your git documentation folder a document with your data models. If you are using a document-based data store such as MongoDB/mongoose, your description will be a commented JSON schema describing the format and expected value of each field, which fields are required, and a note on which fields you will use for keys for each schema.

```
const express = require('express')
const router = express.Router()
const mongoose = require('mongoose')


mongoose.connect('mongodb://localhost/CS411')

const Schema = mongoose.Schema
const userSchema = new Schema({
    USER_ID          : String,
    LOCATION_NAME    : Number,
    FREQUENCY        : Number,
    LATITUDE         : Number,
    LONGITUDE        : Number

})
```

Section II: Test Plan and Execution

There are four parts to the assignment for Section II:

1. Form field validations. For each form in your site, create a spreadsheet with the following columns:

Page Name Form ID Field ID Expected Values

In the Expected Values field you will describe the type and the constraints on the field. For example, the field might accept integers between 1 and 100.

Page Name	Form ID.Field ID Expected Values	Test ID	Test Values	Pass/Fail
Main Address Search Bar	location-form.textInput string 0<= n <= 100	field.1	0	Pass
Main Address Search Bar	location-form.textInput string 0<= n <= 100	field.2	50	Pass
Main Address Search Bar	location-form.textInput string 0<= n <= 100	field.3	100	Pass
Main Address Search Bar	location-form.textInput string 0<= n <= 100	field.4	500	Fail
Main Address Search Bar	location-form.textInput string 0<= n <= 100	field.5	Empty string	Pass
Main Address Search Bar	location-form.textInput string 0<= n <= 100	field.6	non-English	Pass

---

2. Data validation. Your project is storing information in a database. For data validation testing, create a spreadsheet similar to the one in part 1 with fields like:

For each data item, exercise the page or script that the item is used in, and inspect the database to confirm that the value is correct. You might find that you need a few more fields to record the data, and you are free to come up with a spreadsheet format that makes sense to you as a team. Note that it also needs to make sense to me.

Table	Field	Type	R/W	Data Source	Constraints	Test ID	Pass/Fail
User	USER_ID	STRING	R/W	Facebook API	String < 500	data.1	PASS
User	LOCATION NAME	STRING	W	location-form.text Input	String < 500	data.2	FAIL
User	FREQUENCY	NUMBER	R/W	MongoDB	Int < 99999	data.3	PASS
User	LATITUDE	NUMBER	R	Geocode API	Float = 9	data.4	PASS
User	LONGITUDE	NUMBER	R	Geocode API	Float = 9	data.5	PASS

---

3. UI validation. Each button and menu on your pages should do something. For UI validation, list each UI element in a spreadsheet, their expected function, and a test for the function, something like:

Page Name	UI Element	Expected Function	Test ID	Pass/Fail
Side Navigation Bar	Button	Slide out sidebar	UI1	Fail
Default Button	Button	Plain Google Map	UI2	Pass
GreenLine Button	Button	Overlay map & show T estimate	UI3	Pass

57 Bus Button	Button	Overlay map & show 57 Bus estimate	UI4	Pass
BU BUS Button	Button	Overlay map & show BU Bus estimate	UI5	Fail
UBER Button	Button	Overlay map & show Uber estimate	UI6	Pass
Lyft Button	Button	Overlay map & show Lyft estimate	UI7	Pass
Walk Button	Button	Overlay map & show walking estimate	UI8	Pass

---

4. Remediation. For each of the tests in parts 1 to 3, use a spreadsheet to capture any failed tests. Decide on the severity of the bug, and assign it to a member of the team to work on. You can also use the spreadsheet to track last-minute changes to the project. Your spreadsheet will look something like:

Bug ID	**Test ID**	**Description**	**Severity**	**Assigned**	**Completed Date**
1	UI2	Need to prevent margin from changing when open	2	Ozzi	
2	Field.2	Search bar should handle more than 500 len string	2	Tommy	
3	UI5	BU BUS estimate is not showing up correctly	2	Kavi	
4	data.2	Location name should handle up to 1000 characters	2	Yuchen	