Testing:

This document is all about testing using data or user. Both tests helped in the development of the website by improving mainly on functionality of the website. Very important issues were found and were fixed in the process until the final release.

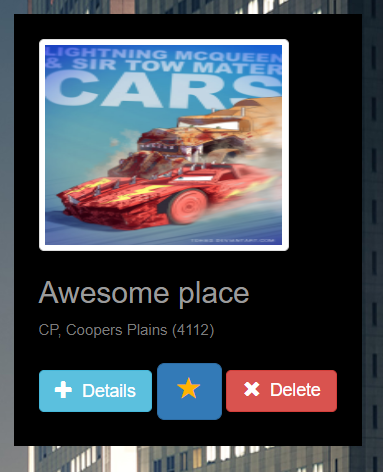
Note: Beta testers are just simply to see if the website was working for every new feature added, however they were team members of the project (beta testers is just a name I came up with)

# Dummy data:

Adding dummy data which is random data thought of on the spot quickly, this helped mainly to check whether the code actually executed (or was working) and worked with the database and was able to communicate to the database and display something on the screen, at the same time helping the developers say if it worked or not, so it was a crucial help to the Django code to see if the code was written correctly.

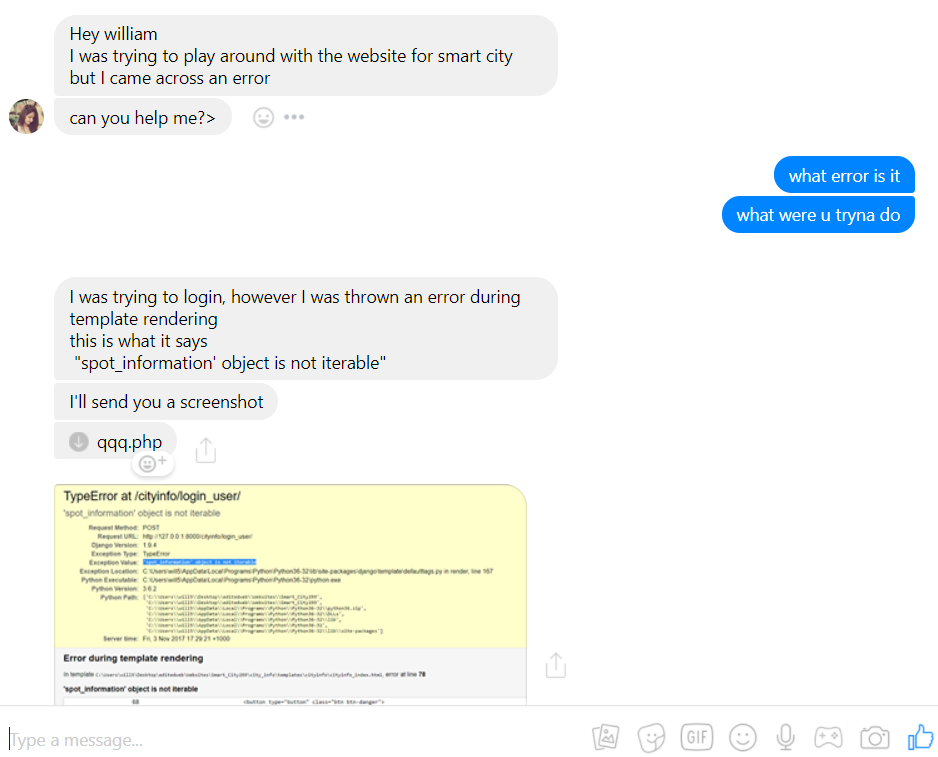
Without any dummy data there is no way of telling whether the website was able to display or not otherwise it would be very dependent on faith, so dummy data ensured it was able to connect to the database and process the data to display on the user’s screen.

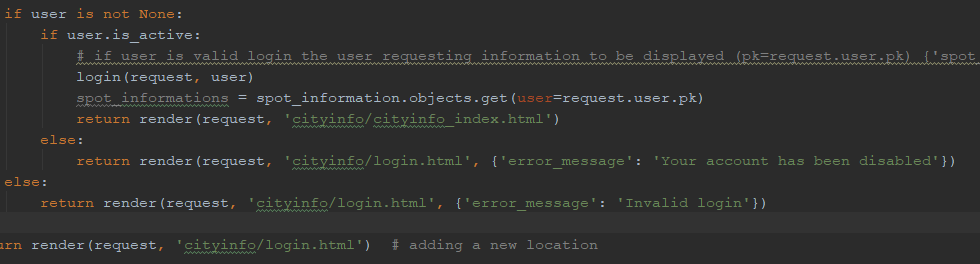
And example of dummy data:



# Beta testing:

This was handing the website to my team members to actually see if the code was running on there side, whenever a new feature was added to the website, this was uploaded to GitHub for other members in the team to run the website themselves, the website itself is considered in beta every time it is uploaded to git, beta testing or allowing users to use the website while still in development stage allowed others to not only use the website, but also at the same time allow the developer to receive crucial information such as issues or problems that arose. Shown below are the information from beta testing.

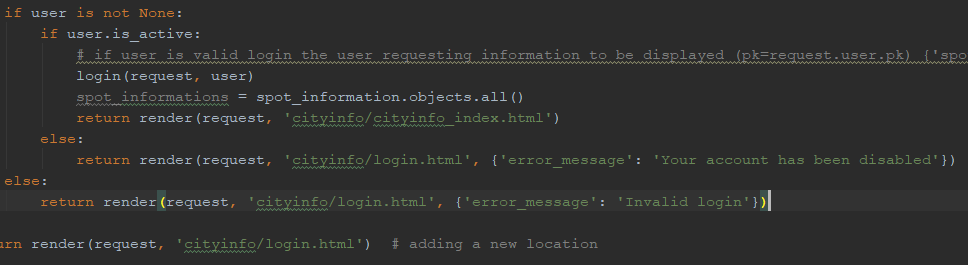
Beata testers Response: 



In views.py “def login\_user”

This line of code cause several issue, where the user was able to login, however, casted an error where it was passing the wrong data, this was missed by the developer until a user actually mentioned this problem and was unable to be redirected to the home page, instead casted a HTML error. Even though this problem should have been noticed by developer (me), however the problem was not noticed as it was working on the developers computer, but not the users computer

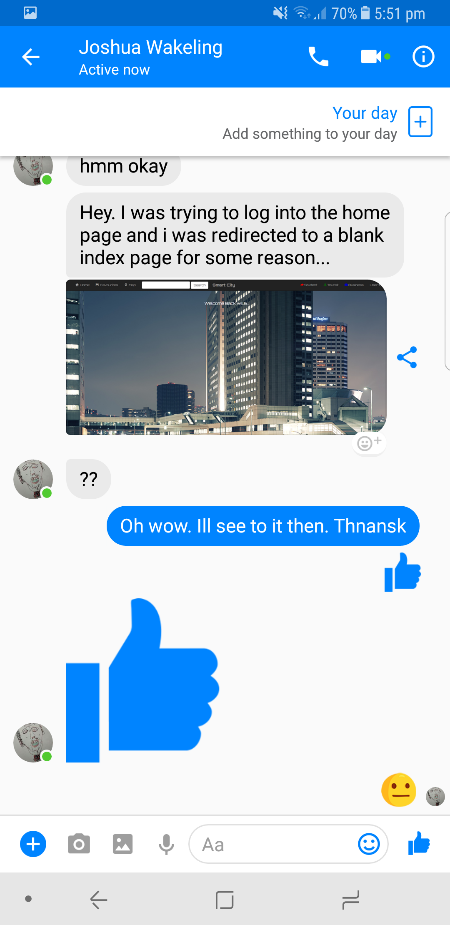
The Fix 01: In views.py “def login\_user”



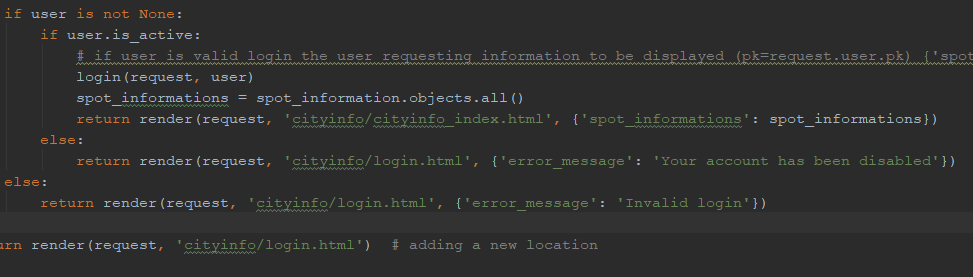
The problem was because the data being collected was not anything to do with the user, but rather the “spot\_information” data itself.

However this also created another issue, the “Fix 01” was not casting an error but instead ended up showing a blank page, with no data showing on the index/home page. As it was noticed by anothere member of the team.

Beata testers response:



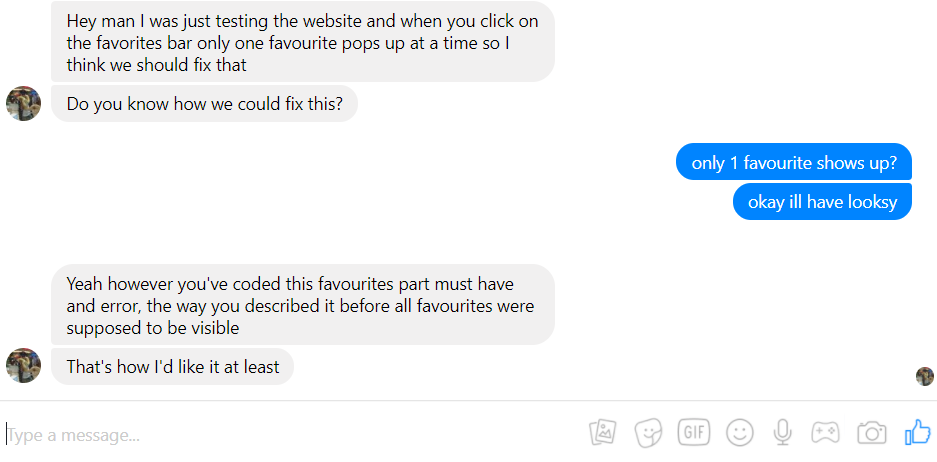
The Fix 02: In views.py “def login\_user”



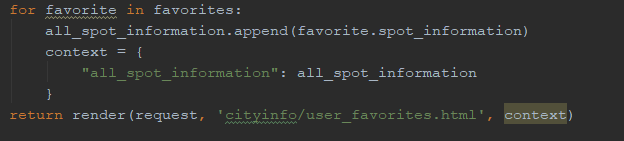
In views.py “def user\_favorites”

Another error which was noticed by ben was one favourite appearing.

Beta testers Response:

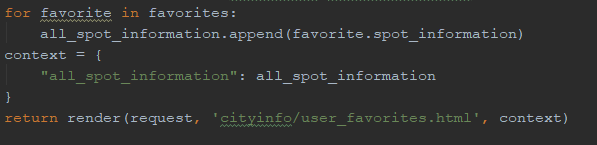


One of the features of the website is favouriting which allowed the user to favourite a favourite spot information to his/her own favourites. This problem was raised by everyone during its first release of the favourites feature, where users (team members) were trying to view all favourited places, however could only view one favourite in the favourites page which was the very first favourited spot/place.



One of the things which Django is separated from other languages is how important indenting is, programs like C or C# require functions to have “{“ and “}” to separate functions, however in Django there is none of that in fact relies on how you indent your code, the above code is an example, the reason being because the “context” variable was indented one tab to far, so the data was not able to be passed, hence during the execution of the code, it was ignored and only runs once and only displayed the first favourite in the table.

The Fix 01: In views.py “def user\_favorites”



In models.py “def spot\_information”

Another issued raised by team members was when adding places to the database, this had made some errors when users where users tried to select “entertainment” in the category section where a warning message was said the characters was over the limit for the column, this was caused because the field “max\_lenght” was set to 10, and the word “entertainment” has 13 characters. 

This was a simple fix by increasing the number to 13 or above.

