Deliverable 1 Software Testing Assignment

Team 3

In this document, we have test cases for the database and frontend components of the software.

The database team has created a test script called "test_database.py" which includes both Black Box and White Box tests. The script consists of one black box test and several white box tests. All of these tests were executed successfully, indicating that the database component of the software does not have any bugs in the code. However, the team will continue to create new test cases to ensure that the code remains bug-free. The figure below shows the outcome of these test cases.

On the other hand, the frontend team has developed a single black box test case in “BlackBoxTest_FrontEnd_1” . This test case involves rendering a heat map using data from the database. The test was conducted by utilizing the data in the script without directly accessing the database. The resulting HTML code displayed a homepage with heat spots on the map. The second figure illustrates the outcome of this test case.

Unfortunately, the backend team was unable to provide any test cases due to time constraints. Most of their time was dedicated to developing the frontend and backend functionalities of the project, leaving little time to focus on creating test cases. However, had they been able to create test cases, they would have included one to compare the data retrieved from the database with the data in an Excel sheet to ensure consistency. Another example would involve taking input data and generating heat spots on the heatmap.

Although there is still a considerable amount of testing remaining, the team is actively working on identifying the necessary test cases. Thus far, the software has demonstrated satisfactory performance in its intended application.

The figures can be found below, they are in a landscape orientation

Deliverable 1 Software Testing Assignment

Test Cases for `database.py`

Authors of All Test Cases on Database Side: Talha Hussain, Thomas Nguyen

Testers of All Test Cases on Database Side: Talha Hussain, Thomas Nguyen

Database Black Box Testing for Crime Database

#	Description	Input Parameters	Expected Output	Actual Output	P/F Criteria	Comments
1	database.py: Run Data Entry.	Not needed.	Database to Have All Tables: crime, crime_type, weapon, neighborhood Crime Table has 560,038 Rows. Crime Inserted includes 543,335 Rows. Crime with Missing Data includes 16,703 Rows. Crime Type Table has 86 Rows. Weapon Table has 23 Rows. Neighborhood Table has 279 Rows.	Database to Have All Tables: crime, crime_type, weapon, neighborhood Crime Table has 560,038 Rows. Crime Inserted includes 543,335 Rows. Crime with Missing Data includes 16,703 Rows. Crime Type Table has 86 Rows. Weapon Table has 23 Rows. Neighborhood Table has 279 Rows.	Actual Output = Expected Output	Everything worked as expected. Missing data includes: - Invalid Weapons - Invalid Latitude - Invalid Longitude

Database White Box Testing for Crime Database

#	Description	Input Parameters	Expected Output	Actual Output	P/F Criteria	Comments
1	Selecting Data from Database	Id: 12	"crime_id = 12, type_name = 6G, type_Description = LARCENY, longtitude = -76.609948, latitude = 39.354788, weapon_id = 0"	"crime_id = 12, type_name = 6G, type_Description = LARCENY, longtitude = -76.6099, latitude = 39.3548, weapon_id = 0"	Actual Output = Expected Output	Floating Point Values were rounded.
2	database.py: dropTables() Function	Not needed.	Return Value = 0, No Tables in the Database.	Return Value = 0, No Tables in the Database.	Actual Output = Expected Output	All working correctly.
3	database.py: createTables() Function	Not needed.	Return Value = 0, Tables: weapon, neighborhood, crime_type, crime	Return Value = 0, Tables: weapon, neighborhood, crime_type, crime	Actual Output = Expected Output	All working correctly.
4	database.py: insertWeapon(crime_data) Function	Not needed.	Return Value = 0, Length of Weapon Table = 11	Return Value = 0, Length of Weapon Table = 11	Actual Output = Expected Output	All working correctly.
5	database.py: insertNeighborhood(crime_data) Function	Not needed.	Return Value = 0, Length of Neighborhood Table = 170	Return Value = 0, Length of Neighborhood Table = 170	Actual Output = Expected Output	All working correctly.
6	database.py: insertCrime_type(crime_data) Function	Not needed.	Return Value = 0, Length of Crime Type Table = 33	Return Value = 0, Length of Crime Type Table = 33	Actual Output = Expected Output	All working correctly.
7	database.py: insertCrime(crime_data) Function	Not needed.	Return Value = 0, Length of Crime Table = 462	Return Value = 0, Length of Crime Table = 462	Actual Output = Expected Output	All working correctly.

All tests were able to be performed without issue, for the White Box Testing, we utilized a sub-sample of the first 500 lines from the "Part_1_Crime_Data.csv" Comma-Separated Values CSV File to ensure quick testing of the functions. The only tests with values that were not identical was White Box Test #1 where the floating point representation that MySQL Workbench used rounded the value to 4 decimal places.

There were no defects in the code written to build the database structure or to fill it with information from the Excel Microsoft Office Open Extensible Markup Language XML Format Spreadsheet and Comma-Separated Values CSV Files. We built the `test_database.py` file to automate testing of the individual functions and to provide data out for ease of tests.

Deliverable 1 Software Testing Assignment

Test Cases for app.py

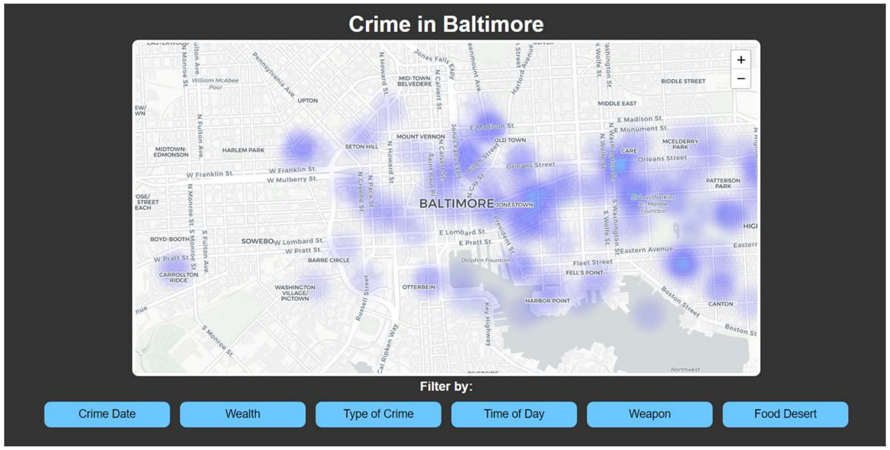
Test File is called BlackBoxTest_FrontEnd_1.py

Authors of the Test Cases: Hasan Ali, Riley Sheehy

Testers of the test Cases: Hasan Ali, Riley Sheehy

Front End Black Box Testing

#	Description	Input Parameter	Expected Output	Actual Output	P/F Criteria	Comments
1 app.py	Testing the frontend of the heatmap with dummy data	Part_1_Crime_Data.csv	A heat map showing the crimes	A heatmap showing the crimes	The UI shows a heatmap with crimes corresponding to the data from the input	We called and imported the data directly from the .csv file and not the database



This is a screenshot of the map with data. At the time our Part_1_Crime_Data.csv file was not populated with enough data. However, it does show crimes hot spot on the map.