Broadsword GIS
NavUP
System Testing
Report

Johan du Plooy - 12070794
Dimpho Mahoko - 15175091
Bernhard Schuld - 10297902
Mankgwanyane Tlaka - 14351872
Kamogelo Tsipa - 13010931
Hendrik van der Mewe - 15101283

May 5, 2017

Contents

1 Over-View Tests

1.1 Ad-hoc testing

1.1.1 description

This type of software testing is very informal and unstructured and can be performed by any stakeholder with no reference to any test case or test design documents.The person performing Ad-hoc testing has a good understanding of the domain and workflows of the application to try to find defects and break the software. Ad-hoc testing is intended to find defects that were not found by existing test cases.

1.1.2 What is being Tested

The GIS module is responisble for all the GIS related parts of the software, this includes admin being able to CRUD any location; a user being able to be given coordinates of a location.

This module provide services to search for locations such as landmarks, buildings as well as venues such as offices, lecture halls, labs, etc.

Use Cases to be tested

Admin: CRUD any kind of singular element

Import: Create a batch of elements via upload of a file in specified formats.

User: Get XYZ coordinates of a named location

1.1.3 Results after testing

All CRUD functionalities of the systems do work well as expected, without any delay or having to understand the code the stakeholder is able to grade this functionality as working perfect.

When trying the create a batch of elements via uploading a file, the system files. it seems that this functionality is not implemented since a stake holder does not see an option to upload a batch file of elements.

1.2 Acceptance Testing

1.2.1 description

Acceptance testing is a formal type of software testing that is performed by end user when the features have been delivered by developers. The aim of this testing is to check if the software confirms to their business needs and to the requirements provided earlier. Acceptance tests are normally documented at the beginning of the sprint (in agile) and is a means for testers and developers to work towards a common understanding and shared business domain knowledge.

1.2.2 What is being Tested

The main user of the system is the everyday student/lecturer and visistors to the University of Pretoria. The functionalities an end user would be intrested in testing are the following:

- Get XYZ coordinates of a named location
- search for locations such as landmarks, buildings as well as venues such as offices, lecture halls, labs, etc.

1.2.3 Results after testing

Acceptance testing meets the user's expectations. The modulle is able to perfome all the functionalities that the end user is intrested in. The end user grades the system a 9/10 for the functions mentioned above.

1.3 Accessibility Testing

1.3.1 description

When doing accessibility testing, the aim of the testing is to determine if the contents of the system can be easily accessed by disable people. Various checks such as color and contrast (for color blind people), font size for visually impaired, clear and concise text that is easy to read and understand.

1.3.2 What is being Tested

The GIS is a moduleIt is about the creation and maintenance of a GIS Map of the campus and persisting information that can be applied to determine the location of a device based on WiFi signal strengths and other available sources of GIS information.

1.3.3 Results after testing

Accessibility testing assummes that the entire system is complete and features which are associated with Human and Computing interaction and User Experience are already in place. as a result of this, the results of this test are depended on the access module and one cannot conclude good results by only looking at the GIS modulle

2 API Testing

2.1 Type of test to be applied: Black-Box Testing

2.1.1 Reason for choosen Testing Method

We choose Black-Box Testing since GIS is a modulle to the NavUP software. this allows us to examine the functionality of the Gladios GIS modulle.

2.2 Test Cases

Functions implemeted

- public String geAlltBuildings(Double lat, Double lon);
- public String getBuilding(Double lat, Double lon);
- ArrayList getLectureHall(String building);
- public ArrayList getLectureCoordinates(String room);
- public ArrayList getBuildingCoordinates(String building);
- public ArrayList getBuildingInRadius(double mLat, double mLon, double radius);

- public void insertBuilding(String name, String description, String geometry, String coordinates, String table);

Table 1: Test cases

Function	Description	Mark(10)	comment
getAllBuildings	This function returns a list of all the buildings in the GIS database	6	Apply additional error handling Function very reliable
getBuilding;	This function returns the details of a building specified by the user	10	comment
getLectureHall	This function returns the location of a lecture hall specified by the user	6	Make use of quey optimizing strategies to improve query execution time Add Additional error handling for incorrect request Function very reliable
getLectureCoordinates	This function returns the coordinates of a lecture hall	10	comment
getBuildingCoordinates	This function returns the coordinates of a specified bulding	10	comment
getBuildingInRadius	Not sure what this does	10	comment
insertBuilding;	This function inserts a building to the database using the parameters	10	comment

3 Non-Functional Requirements Tested

3.1 List of Non-Functional Requirements

Non-Functional Requirements

- Performance
- Scalability
- Security
- Accessibility
- Maintainability

3.2 Marking

Table 2: Non-Functional Requirements

Function	Description	Mark(10)	comment
Performance	Does the system perform well under different possible environments and situations	10	System performance is sufficient
Scalability	Can the system handle a sufficient amount of data at a time and overtime	7	Processing lots of data at once insufficient. Can only add 1 building at a time
Security	Is the system sufficiently protected from the possible threats that could occur	8	Confirm user type before fulfilling the query, as a guest user can access admin user functions.
Accessibility	Is the system easily accessible to people with disabilities	6	Does not indicate friendly wheel chair paths or any disability provision
Maintainability	Can the system be maintained with ease regarding database and source code	9	Updating the database works fine and the source code is structured in a modular enough manner to add new features with few changes to the rest of it

4 Evaluation of Test Cases

1. **Creating map object linked to a spacial database**
   Mark: 4.
   Comment: Method takes in database details and successfully returns a new GIS map object, however no checks are being performed to validate the returned object. The method does catch Exceptions, but only logs the error. No proper exception handling is being performed.

   While the method does work with correct paramaters, no unit tests were written and as such the method was not properly tested. Examples of unit tests that should have been written are as follows:

   - All parameters entered, and all parameters are correct
   - All parameters entered, some are correct and some are incorrect
   - All parameters entered, all are incorrect
   - Some parameters entered, all entered are correct
   - Some parameters entered, only some of the entered parameters are correct
   - Some parameters entered, all entered are incorrect
   - No parameters entered

2. **Getting all buildings**
   Mark: 6.
   Comment: Method takes no parameters and returns and array that contains all buildings from the database. The method does catch Exceptions, but only logs the error. No proper exception handling is being performed.

   While the method does work but no unit tests were written. Even though the method is so simplistic, unit tests should still have been written. Only two tests will have sufficed for this test:

   - The database is up and running, method returns correctly
   - The database is not up and running, method gracefully fails and correctly handles the exception.

3. **Get specific building's coordinates**
   Mark: 5.
   Comment: Method takes in a string (building name) and returns the coordinates of that building. The method does catch Exceptions, but only logs the error. No proper exception handling is being performed.

   While the method does work with correct paramaters, no unit tests were written and as such the method was not properly tested. Only two tests will have sufficed for this test:

   - The building name entered does exist in the database, coordinates are returned
   - The building name entered does not exist in the database, method gracefully fails and correctly handles the exception.