COS301
Mini Project
Testing

Nardus van der Vyver u15012698
Minal Pramlall u13288157
Peter Rayner u14001757
Mandla Mhlongo u29630135

May 4, 2017

https://github.com/MandlaMashinini/Phase4_Testing

Contents

1 Improving On The Existing Test Output

2 Service Contract Of Each Of The Use Case Tested

1. Get building by coordinates:
   Mark: 10.
   Comment: The function correctly returns all the information on a building based on the coordinates provided.

2. Get building by name:
   Mark: 10.
   Comment: The function correctly returns all the information on a building based on the name provided.

3. Get location by room number:
   Mark: 10.
   Comment: The function correctly returns the location of a room by providing the building name and room.

4. Get routes:
   Mark: 8.
   Comment:
   The function mostly provides the correct route between a start and end point for a user.

5. Insert building:
   Mark: 10.
   Comment: The function correctly inserts a building(location) and all its details into the database.

6. Insert building room:
   Mark: 10.
   Comment: Correctly inserts a new room into an existing building of the database.

7. Update building coordinates:
   Mark: 10.
   Comment: The function correctly updates the coordinates of an existing building in the database.

8. Update building name:
   Mark: 10.
   Comment: Correctly updates an existing locations name(building name).

9. Update building room:
   Mark: 10.
   Comment: Correctly updates an existing room name inside a given building.

10. Update building room coordinates:
    Mark: 10.
    Comment: This function correctly updates the coordinates of an existing room inside a building.

11. Remove building:
    Mark: 10.

Comment: The function removes a building and all its dependencies correctly from the database.

12. Remove building room:
    Mark: 10.
    Comment: This function removes a room correctly from building it is contained in.

3 Non-Functional Requirements Tested

1. Maintainability:
   Mark: 10.
   Comment: Classes are separated well and allows easier access and retrieval to the object data.

2. Performance:
   Mark: 10.
   Comment: Recognized that Postgres is the best performing database for this system, thus has better response times.

3. Scalability:
   Mark: 10.
   Comment: Return methods allow for batch processing, returning a collection of objects at a time, as opposed to one a time.

4. Interoperability:
   Mark: 10.
   Comment:

This module runs with Java, similar to the other teams, and thus allows interchangeability with the other modules or the interfaces it can communicate with.

5. Usability:
   Mark: 10.
   Comment: Enough class accessors and mutators have been provided to allow full use of the data objects.

6. Data Integrity and Security:
   Mark: 10.
   Comment: Communication between classes leave little room for problems with data communication, GIS also has little to no interaction with any sensitive information so security measures need not be so strenuous.

7. Transparency:
   Mark: 10.
   Comment: GIS module returns the locations and as much information relating to it, which makes the need for it to be as transparent as possible, which is delivered. Class objects provide access to all the fields and the methods themselves are named to clearly describe their function.

8. Documentation:
   Mark: 10.
   Comment: All methods are well documented.

9. Reliability:
   Mark: 9.

Comment: Methods of this subsystem consistently return correct data, minus getting routes, which is - nevertheless - still mostly correct.

10. Availability:
    Mark: 10.
    Comment: Methods are easily accessible and maintain consistency with return values.