**Test Document**

The application should satisfy the following test cases:

**Map Editor**:

1. Test to check if a map validation with no start point will fail.
2. Test to check if a map validation with no end point will fail.
3. Test to check if a map validation without a continuous path between start and end point will fail.
4. Test to check if a map validation with an orphan path will fail.
5. Test to check if a map validation with 2 or more start points will fail.
6. Test to check if a map validation with 2 or more end points will fail.
7. Test to check if a map validation with duplicate path between start and end point will pass.
8. Test to check if a map validation with only start and end point will fail.
9. Test to check if a map validation with branches in the path between start and end point will pass.
10. Test to check if a path object can be successfully added to the map.
11. Test to check if a map validation with multiple path between start and end point will generate the shortest path between the start and end point.
12. Test to check whether a map read from file is same as the one previously written to it.

**Main Game**:

1. Test to check if the map model object is created correctly from the map in file.
2. Test to check if a tower can be successfully added to the scenery.
3. Test to check if a tower can be successfully deleted from the map.
4. Test to check if critter exist at a particular location in map.
5. Test to check if when the game data model account balance is updated, the corresponding update is made on the account balance in view.
6. Test to read a map from file and check if it is read properly.
7. Test to check if SingleGameController is implementing Singleton design pattern.
8. Test to sell a tower and get the account balance updated with refund value.
9. Test to fail to upgrade a tower with low account balance.
10. Test to upgrade a tower successfully with enough account balance.
11. Test to check if the strongest critter is selected by the tower strategy.
12. Test to check if the weakest critter is selected by the tower strategy.
13. Test to check if the nearest critter to tower is selected by the tower strategy.
14. Test to check if the factory design is implemented properly and polymorphism is achieved.