Dylan Brennan
OR3 lab 4

Test Cases

1) Testing your program for all use cases is a good way to gain confidence that your program is functioning correctly, and that functions will perform correctly for users. It is also important to test your program for more extreme cases, to make sure a malicious user can't break your code or access private data.
2) Yes, we tested the program with various incorrect inputs to make sure our program would reject these cases without causing problems.
3)

Test Title	Sign up test
Pre-conditions	Username must not already be in DB

Test data	Username = test100
	Pass = test100
	Location = none

Expected result	Doesn't add test100 to database because test100 is already in DB
Test status	Fail
Bugs	Test100 was still added to the DB
Bug resolution	Fix this by first checking if user is in DB in UI before calling createUser to modify DB.
Post-conditions	Fixed bug

Test Title	Sign in test
Pre-conditions	Sign in if username and pass is right
Test data	Sign in with username = root and password = root
Expected result	Pass
Test status	Test passed with no errors

Bugs	
Bug resolution	
Post-conditions	

Test Title	CangePass test
Pre-conditions	Change pass if current password is correct
Test data	Username = dylan, old_password = password, new_password = secret
Expected result	Pass will be updates to new password
Test status	Pass
Bugs	
Bug resolution	
Post-conditions	

Test Title	Favorite Restaurant test, changes favorite restaurant of user and changes both previous and old favorite counter by -1/+1 respectively
Pre-conditions	none
Test data	User = dylan, old_restaurant = mcdonalds new_restaurant = mooya
Expected result	Dylans new favorite restaurant will be mooya, and both mooya and mcdonalds counter will be updates
Test status	Fail
Bugs	Mcdonald's counter was not updated
Bug resolution	Fixed code to first update old restaurant, and then update new restaurant.
Post-conditions	Problem was fixed

Test Title	Search restaurant test
Pre-conditions	Restaurant is in DB
Test data	Restaurant = mooya
Expected result	Will display restaurant information and 5 most recent reviews
Test status	Pass
Bugs	
Bug resolution	
Post-conditions	

Test Title	Sort reviews test
Pre-conditions	Restaurant is in DB
Test data	Restaurant = mooya
Expected result	Display 5 random sets of reviews of good, bad, & neutral
Test status	Fail
Bugs	Not all 5 reviews were displaying
Bug resolution	Fixed by correcting the if statement that added randomness
Post-conditions	none

Test Title	User Review Test
Pre-conditions	User and restaurant must exist
Test data	User = root restaurant = mooya rating = 10 comment = TESTINGGGGGGGG3213123
Expected result	Rating, review, & user should be stored in mooyas reviews
Test status	Pass

Bugs	
Bug resolution	
Post-conditions	

Test Title	Delete user test
Pre-conditions	User must be in DB and only admin can request deletion
Test data	User = test123, admin = root
Expected result	User will be removed from DB
Test status	Pass
Bugs	Reviews from user remain in DB
Bug resolution	No solution found
Post-conditions	

Test Title	Add admin test
Pre-conditions	User and admin must be in DB
Test data	User = dylan, admin = root
Expected result	User will be turned into an admin
Test status	Pass
Bugs	
Bug resolution	
Post-conditions	

Test Title	DataMine
Pre-conditions	Restaurant is in DB & admin must request a datamine

Test data	Restaurant = mooya, admin = root
Expected result	“Opinion” section for mooya in DB will be updated to positive, negative, or neutral
Test status	Pass
Bugs	
Bug resolution	
Post-conditions	