Part 6: Testing

Title: BeerMe

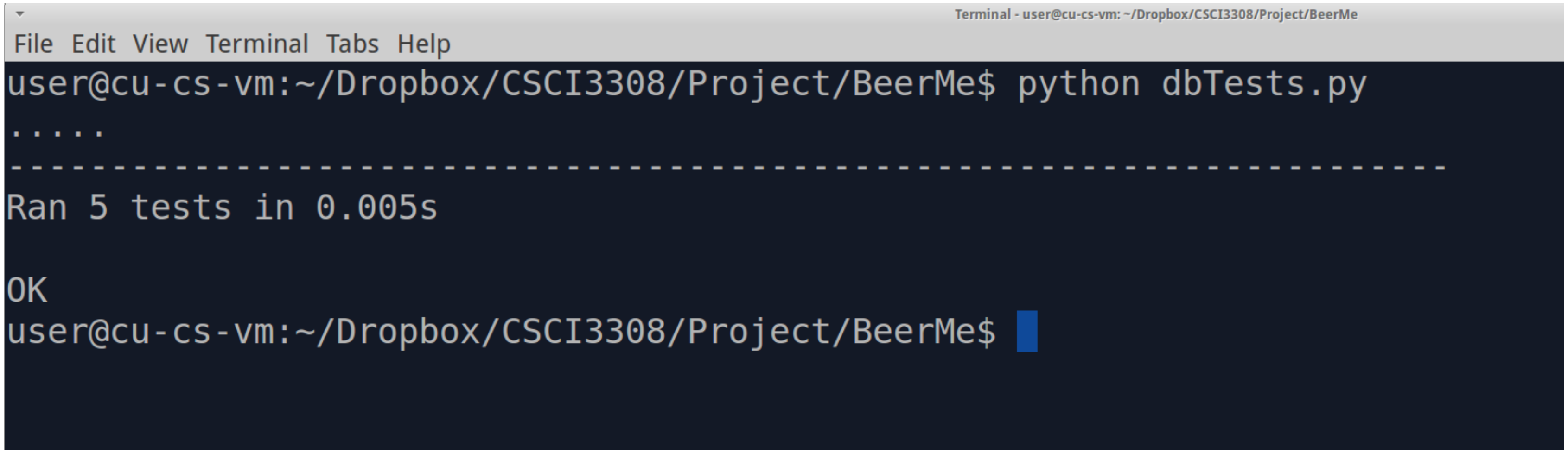
Vision Statement: Those 21 years of age and over will have access to a straightforward application that gives an instant and personal recommendation for what type of beer would best fit his/her given situation, based on his/her input

Who:

- Michelle Bray
- Jon Allured
- Spencer Wilson
- Jacob Levine
- Matthew Geckel

Automated Testing:

Our automated tests are run in Python using MySQLdb to access the database and unittest to run the tests. You may need to install MySQLdb if it is not already installed. You must import the db.sql from GitHub into a MySQL database named “BeerMeDB” and you may need to change the username and password for MySQL in the dbTests.py file on line 11 to your MySQL username and password.


Terminal - user@cu-cs-vm: ~/Dropbox/CSCI3308/Project/BeerMe
File Edit View Terminal Tabs Help

```
user@cu-cs-vm:~/Dropbox/CSCI3308/Project/BeerMe$ python dbTests.py
.....
----------------------------------------------------------------------
Ran 5 tests in 0.005s

OK
user@cu-cs-vm:~/Dropbox/CSCI3308/Project/BeerMe$ 
```

User Acceptance Testing:

Project Name: **BeerMe**						
Test Case Template						
Test Case ID: FUN_04				Test Designed By: Jon Allured		
Test Priority (low/med/high): High				Test Designed Date: 4/01/15		
Module Name: Age Verification				Test Executed By: Jon Allured		
Test Title: Age Verify Allow				Test Execution Date: 4/01/15		
Description: allow access to 21+						
Pre-Conditions: None						
Dependencies: None						
Step #:	Test Steps:	Test Data:	Expected Result	Actual Result:	Status (Pass/Fail):	Notes:
1	Navigate to ageverify.html					

2	Click on the radial button labeled ‘I am 21 or over’		User should be redirected to index.html, along with the rest of the site	User is taken to index.html	Pass	
3						
4						
5						
…						
Post Conditions: user is given access to the website						

Project Name: **BeerMe**						
Test Case Template						
Test Case ID: FUN_03				Test Designed By: Jon Allured		
Test Priority (low/med/high): Med				Test Designed Date: 4/01/15		
Module Name: Age Verification				Test Executed By: Jon Allured		
Test Title: Age Verify Deny				Test Execution Date: 4/01/15		
Description: Deny access to 20-						
Pre-Conditions: None						
Dependencies: None						
Step #:	Test Steps:	Test Data:	Expected Result:	Actual Result:	Status (Pass/Fail):	Notes:
1	Navigate to verifyage.html					

2	Click on the radial button labeled ‘I am not yet 21’		User should be directed away from index.html	User is directed to a more appropriate location	Pass	
3						
4						
5						
…						
Post Conditions: user will not be able to access the rest of the website						

Project Name: **BeerMe**						
Test Case Template						
Test Case ID: FUN_04				Test Designed By: Michelle Bray		
Test Priority (low/med/high): High				Test Designed Date: 3/31/15		
Module Name: Navigation				Test Executed By: Michelle Bray		
Test Title: Nav Bar				Test Execution Date: 4/01/15		
Description: ensure that links to all other pages work properly via the nav bar						
Pre-Conditions: All pages must exist						
Dependencies: The user has access to index.html						
Step #:	Test Steps:	Test Data:	Expected Result	Actual Result:	Status (Pass/Fail):	Notes:
1	Navigate to index.html					

2	Click on ‘Login’		User should be redirected to login.html	User is taken to login.html	Pass	
3	Click on ‘Beer Info’		User should be redirected to beerinfo.html	User is taken to beerinfo.html	Pass	
4	Click on ‘Other Resources’		User should be redirected to resources.html	User is taken to resources.html	Pass	
5	Click on ‘About Us’		User should be redirected to about.html	User is taken to about.html	Pass	
…						
Post Conditions: Ensure nav bar is accessible on every page of the website						

Project Name: **BeerMe**		
Test Case Template		
Test Case ID: DB_01		Test Designed By: Spencer D. Wilson
Test Priority (low/med/high): High		Test Designed Date: 3/30/15
Module Name: DB		Test Executed By: Spencer D. Wilson
Test Title: DB Display		Test Execution Date: 3/31/15
Description: Testing that Database is set up such that long sentences will display in their entirety		

Pre-Conditions: The Database is constructed by sourcing our db.sql file and we are in the database						
Dependencies: MYSQL must be installed						
Step #:	Test Steps:	Test Data:	Expected Result:	Actual Result:	Status (Pass/Fail):	Notes:
1	SELECT (Beers.description) FROM Beers WHERE Beers.Id = 3;	Testing longest sentence in Beers table	A table containing: The Light Lager is generally a lighter version of a breweries premium lager, some are lower in alcohol but all are lower in calories and carbohydrates compared to other beers. Typically a high amount of cereal adjuncts like rice or corn are used to help lighten the beer as much as possible. Very low in malt flavor with a light and dry body. The hop character is low and should only balance with no signs of flavor or aroma. European versions are about half the alcohol (2.5-3.5% abv) as their regular beer yet show more flavor (some use 100% malt) then the American counterparts. For the most part this style has the least amount of flavor than any other style of beer.	A table containing: The Light Lager is generally a lighter version of a breweries premium lager, some are lower in alcohol but all are lower in calories and carbohydrates compared to other beers. Typically a high amount of cereal adjuncts like rice or corn are used to help lighten the beer as much as possible. Very low in malt flavor with a light and dry body. The hop character is low and should only balance with no signs of flavor or aroma. European versions are about half the alcohol (2.5-3.5% abv) as their regul	Fail	Must change the varchar limit for the description field of the Beers Table
2	SELECT (FoodPairing.Review) FROM FoodPairing	Testing longest sentence in	A table containing: A very crisp and refreshing beer, this	A table containing: A very crisp and refreshing beer, this	Pass	

	WHERE FoodPairing.Id = 1;	FoodPairing table	easy-to-drink pale ale is an excellent complement to a delicious burger. The heavier, meaty flavor of the burger pairs extraordinarily well with light, fresh taste of the beer. You'll have a hard time finding a better brew than this to wash down your bites of juicy burger!	easy-to-drink pale ale is an excellent complement to a delicious burger. The heavier, meaty flavor of the burger pairs extraordinarily well with light, fresh taste of the beer. You'll have a hard time finding a better brew than this to wash down your bites of juicy burger!		
3						
4						
5						
…						
Post Conditions: Must alter the db.sql file and reload the beers table						

Project Name: **BeerMe**		
Test Case Template		
Test Case ID: DB_01		Test Designed By: Spencer D. Wilson
Test Priority (low/med/high): High		Test Designed Date: 3/30/15
Module Name: DB		Test Executed By: Spencer D. Wilson
Test Title: DB Display		Test Execution Date: 4/01/15
Description: Ensure that we can produce clean output from the DB that might be utilized by our web developers for filtering beer recommendations		

based on the type of food someone wishes to consume						
Pre-Conditions: The Database is constructed by sourcing our db.sql file and we are in the database, DB_01 has not yet passed in its entirety						
Dependencies: DB_01 step #02 must Pass						
Step #:	Test Steps:	Test Data:	Expected Result:	Actual Result:	Status (Pass/Fail):	Notes:
1	SELECT concat(FoodPairing.Name, ' a ', Color.Name, ' color beer of the ', Beers.name, ' style with ', Beers.abv, ' abv pairs well with ', Food.Name, ' \n Observation: ', FoodPairing.Review) AS ' Beers That Pair Well With Red Meat include:' FROM Food, FoodPairing, Color, Beers WHERE FoodPairing.FoodId = 1 AND FoodPairing.FoodId = Food.Id AND FoodPairing.BeerId = Beers.Id AND FoodPairing.ColorId = Color.Id;	What is the output for beers pairing with red meat	A large table of information regarding the brand names reviewed, beers, general abv, color of the beer type, and the written review	Well-formatted table with the expected information	Pass	
2	SELECT concat(FoodPairing.Name, ' a ', Color.Name, ' color beer of the ', Beers.name, ' style with ', Beers.abv, ' abv pairs well with ', Food.Name, ' \n Observation: ', FoodPairing.Review) AS	What is the output for beers pairing with sea food	'There are currently no beers in our database that pair well with Sea Food'	Empty set (0.00 sec)	Fail	Know this output we could force the desired output with a conditional print statement.

	'There are currently no beers in our database that pair well with Sea Food' FROM Food, FoodPairing, Color, Beers WHERE FoodPairing.FoodId = 4 AND FoodPairing.FoodId = Food.Id AND FoodPairing.BeerId = Beers.Id AND FoodPairing.ColorId = Color.Id;					
3						
4						
5						
...						
Post Conditions: Must write code to deal with the cases for foods that we have not yet paired beer with						