Test Document

for

Canteen Order Automation System

Version <1.0>

Prepared by

Group #: 16 **Group Name:** ***Hardwired***

Sharath Kumar V	**200916**	**sharathk20@iitk.ac.in**
Kavya Jalan	**200503**	**kavyajalan20@iitk.ac.in**
Jhaansi Reddy	**200477**	**jhaansir20@iitk.ac.in**
Ishan Bawne	**200456**	**ishanb20@iitk.ac.in**
Rashmi G R	**200772**	**rashmigr20@iitk.ac.in**
Mohd Shadab	**200593**	**mshadab20@iitk.ac.in**
Mohil	**200596**	**mohil20@iitk.ac.in**
Rathod Preet	**200775**	**preetr20@iitk.ac.in**
Harshit Kumar Tiwari	**200432**	**harshitkt20@iitk.ac.in**
Alaganji Rohan Satvik	**150082**	**arsatvik@iitk.ac.in**

Course: CS253

Mentor TA: Aman Aryan

Date: 04- 04- 2022

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Version 1.0	Sharath Kumar V Kavya Jalan Jhaansi Reddy Ishan Bawne Rashmi G R Mohd Shadab Mohil Rathod Preet Harshit Kumar Tiwari Alaganji Rohan Satvik	Testing Document for Canteen Order Automation System web application.	04/04/22

1 Introduction

We used automated testing to test the Canteen Order Automation system using Django's unit tests which uses a Python standard library module: **unittest.** In this,tests are defined in a class-based approach. This gives us abstraction. Also, the database created is automatically deleted after running the test. Also, we had minor manual testing sessions too during the implementation to test parts and subparts of the code, and to also test the application as a whole.

Testing was done by the developers in parallel with implementation. Our tests mainly focuses on the structural coverage criteria, though functional coverage has also been fulfilled to an extent.

2 Unit Testing

1. User Login

Unit Details: views.login_user
Test Owner:
Test Date: 19/03/2022
Test Results: The test was conducted to know if login is successful for the right credentials provided (correct username and password) and is redirected to the home page.In case of wrong credentials, an error page is given as response, is checked. The tests passed.
Structural Coverage: Function coverage, branch coverage, decision coverage
Additional Comments: -

2. Profile Page

Unit Details: views.profile
Test Owner:
Test Date: 19/03/2022
Test Results:This tests checks, given a user has logged in, if her/his details are shown correctly in the profile page. The test is passed.
Structural Coverage: Function coverage, statement coverage
Additional Comments: This test is not exhaustive, though the provided test implies its correct working.

3. Menu Pages

Unit Details: views.menu , models.MenuItem
Test Owner:
Test Date: 20/03/2022
Test Results:This tests checks, given a user has logged in and wants to view the menu, if the code checks if the user is accessing it as a staff or a student, and gives the respective pages. Tests passed.
Structural Coverage: Function coverage, statement coverage,conditional coverage, branch coverage
Additional Comments: -

4. Test rating of menu item

Unit Details: views.menu, models.MenuItem
Test Owner:
Test Date: 20/03/2022
Test Results:This test is for checking if the rating displayed is correct by averaging all the ratings given by buyers. Test passed.
Structural Coverage: Function coverage, statement coverage,loop coverage
Additional Comments: Not exhaustive

5. Test computing of unpaid bills

Unit Details: views.orders, models.Order
Test Owner:
Test Date: 20/03/2022
Test Results:This test is for checking if the unpaid bills being displayed in the orders page of the student displays the correct pending dues. Test passed.
Structural Coverage: Function coverage, statement coverage, loop coverage
Additional Comments: -

6. Test to check correct bill at cart

Unit Details: views.savecart, views.tocart, views.cart
Test Owner:
Test Date: 20/03/2022
Test Results:This test is for checking if the bill displayed after adding items is being correctly computed. Test passed.
Structural Coverage: Function coverage, statement coverage, loop coverage, branch coverage
Additional Comments: Not exhaustive

7. Test computing of unpaid bills

Unit Details: views.orders, models.Order
Test Owner:
Test Date: 20/03/2022
Test Results:This test is for checking if the unpaid bills being displayed in the orders page of the student displays the correct pending dues. Test passed.
Structural Coverage: Function coverage, statement coverage, loop coverage
Additional Comments: -

8. Contact Us page

Unit Details: views.contact_us
Test Owner:
Test Date: 19/03/2022
Test Results: Test to check if the right page and details are displayed at the Contact Us page. Test Passed.
Structural Coverage: Function coverage, statement coverage
Additional Comments: -

9. Test database storing

Unit Details: models.UserExt, models.Order, models.MenuItem, models.Review
Test Owner:
Test Date: 19/03/2022
Test Results:This test checks if the database is storing the right and all the information of any model defined. Test passed.
Structural Coverage: statement coverage
Additional Comments: Not exhaustive

10. Test display names of objects in database tables

Unit Details: models.UserExt, models.Order, models.MenuItem, models.Review
Test Owner:
Test Date: 03/04/2022
Test Results: The test checks if the right names are being displayed at the admin site for any object based on our specified name format for respective models. Tests passed.
Structural Coverage: Function coverage, statement coverage
Additional Comments: -
….

3 Integration Testing

1. URLS

Module Details: urls.py, views.py
Test Owner:
Test Date: 19/03/2022 - 04/04/2022
Test Results: In this test, we tested if all the http requests renders to the correct function in views to carry out further functionality. All tests passed.
Additional Comments: -

2. Regression Testing

Module Details: urls.py, views.py, models.py
Test Owner:
Test Date: 19/03/2022 - 04/04/2022
Test Results: From the beginning we have used regression method to test the implementation. After any addition or editing of any part of code or addition of any new test, every other test case checked before is run again together. All tests passed.
Additional Comments: -

….

4 System Testing

1. **Requirement:** LogIn/registration of student/staff
Test Owner:
Test Date: 19/03/2022
Test Results: Test to check if database records details of user during registration and login occurs at providing the right login credentials. Test passed.
Additional Comments: -

2. **Requirement:** Choose menu of canteens of different halls
Test Owner:
Test Date: 19/03/2022
Test Results: Testing if the choice of hall directs to the corresponding menu of the canteen of that hall. Test passed.
Additional Comments: -

3. **Requirement:** View menu
Test Owner:
Test Date: 20/03/2022
Test Results: Test cases checked the display of menu items with the correct rating (average of all the ratings given as feedback for previous orders by the customer). Test passed.
Additional Comments: -

4. **Requirement:** Items in cart
Test Owner:
Test Date: 20/03/2022
Test Results: Test cases checked upon addition of any item, cart can be viewed. Tested the correct display of cart items and editing in place before order placement. Also the right bill amount is displayed. Test passed.
Additional Comments: More tests to be included.

5. **Requirement:** Display due bills.

Test Owner:
Test Date: 20/03/2022
Test Results: Test to compute due bills of a customer and be displayed at the orders page correctly. Test passed.
Additional Comments: -

6. **Requirement:** Admin's page

Test Owner:
Test Date: 20/03/2022
Test Results: Test cases tested that upon a user logged in is confirmed to be a staff s/he can view her/his respective menu page and update the menu.
Additional Comments: Test can be included as a part of integration testing.

7. **Requirement:** Reviews: Rating

Test Owner:
Test Date: 20/03/2022
Test Results: Test to see if a rating is collected and is contributed towards the average of the order's menu item. Passed the test.
Additional Comments: -

….

5 Conclusion

We tried to test as much as all the functionalities of the COAS implementation. The tests included in the testing of COAS were effective, and fulfilled its need. However, it can be seen that the test cases are not exhaustive. Some trivial cases have been left out on purpose, and few others owing to generation of larger number of test cases.

Testing of components on the side of the staff has not been done adequately, and is believed that more rigorous tests can be included. Testing of COAS can be improved by including more integration tests. Also, there is scope to make the tests exhaustive and include more and more coverage criteria.

As for the difficulties faced, it would be rather appropriate to say where we fall short. Exploring, knowing and learning the Django testing interface takes time, and there are still many testing features that are yet to be put into practice in this testing.

Appendix A - Group Log

Date	Duration	Meeting topic	Outcome(s)
16-03-2022	30 minutes	Getting familiar with Django test interface	-
19-03-2022	120 minutes	Deciding, running tests and correcting the code.	More on implementation.
20-03-2022	90 minutes	Including more tests for additions in the codes and using regression testing.	More on testing.
27-03-2022	60 minutes	Including tests, fixing code	-
31-03-2022	30 minutes	Test document details discussion.	
03-04-2022	60 minutes	Test document updation	
04-04-2022	15 minutes	Test document completion.	