**CS673 Software Engineering** 

**Team 3 - Project Name**

**Software Test Document**

| Team Member | Role(s) | Signature | Date |
| --- | --- | --- | --- |
| Zuowen Tang | Security Leader | *Z.T.* | 5/12/23 |
| Alekhya Koppineni | Requirement leader | [Alekhya Koppineni](mailto:alekhya.koppineni@gmail.com) | 5/12/23 |
| Edara Umamaheswar | Team Leader | *Mahesh* | 5/15/23 |
| Adithya Prakash | Design and Implementation Leader | *Adithya Prakash* | 5/15/23 |
| Corydon Wood | Configuration Leader | *Corydon Wood* | 5/15/23 |
|  |  |  |  |
|  |  |  |  |

**Revision history**

| **Version** | **Author** | **Date** | **Change** |
| --- | --- | --- | --- |
| **1.1** | **Zuowen Tang** | **Jun 18, 2023** | **Add additional function in Summary** |
|  |  |  |  |

[Testing Summary](#_sm5odwyvuk3j)

[Manuel Tests Reports](#_pqso2mbjyzx4)

[Automated Testing Reports](#_mtfbusfb0eq3)

[Testing Metrics](#_rijyjeu2ojqa)

[References](#_15tmymhipvdv)

[Glossary](#_8n34lvocupub)

# Testing Summary

**The following features of the website will be tested:**

1. **User Registration, Login, and Profile Management**
2. **Dietary Input and Management**
3. **Diet Tracking and Reporting**
4. **Social Media Function**
5. **Data Privacy and Security**

### 1. User Registration, Login, and Profile Management

* Registering a new user account
* Logging in with an existing user account
* Recovering lost password
* Editing and saving changes in the user profile

### 2. Dietary Input and Management

* Adding, editing, and deleting food items

### 3. Diet Tracking and Reporting

* Tracking food consumption daily, weekly, and monthly
* Generating dietary reports and charts

**4. Social Media Function**

* Sending and viewing posts
* Viewing other/all users’ profiles
* Following and unfollowing other users.

### 5. Data Privacy and Security

* User data protection in transit and at rest
* Proper functioning of user consent mechanisms

**Method**

* **Unit Testing**: For testing individual components/functions of the application.
* **Integration Testing**: To test the interaction between different components of the application.
* **System Testing**: To validate the application's functionality, performance, and reliability.
* **Acceptance Testing**: To ensure the system satisfies the requirements and is ready for delivery.
* **Security Testing**: To identify any potential vulnerabilities or weaknesses.
* **Performance Testing**: To ensure the system performs well under expected and peak load.

## Pass/Fail Criteria

**A test will be considered a pass when:**

* The functionality works as expected according to the requirement specification.
* The application can handle the expected number of users and load.
* No critical security issues are found.

**A test will be considered a fail when:**

* The functionality does not work as expected.
* The application performance is not up to the mark.
* There are security issues that can lead to data leakage or unauthorized access.

*Features or components that are not developed as part of this project, such as third-party APIs or libraries, will not be tested.*

# Manual Testing Report

Here is a sample template that can be used for each test case. For system tests or acceptance tests, you may also include some screenshots.

* Test case ID, name
* New or old:
* Test items: (what do you test )
* Test priority (high/medium/low)
* Dependencies (to other test case/requirement if any):
* Preconditions: (if any)
* input data:
* Test steps:
* Postconditions:
* Expected output:
* Actual output:
* Pass or Fail:
* Bug id/link: (this should link to your github issue id)
* Additional notes:

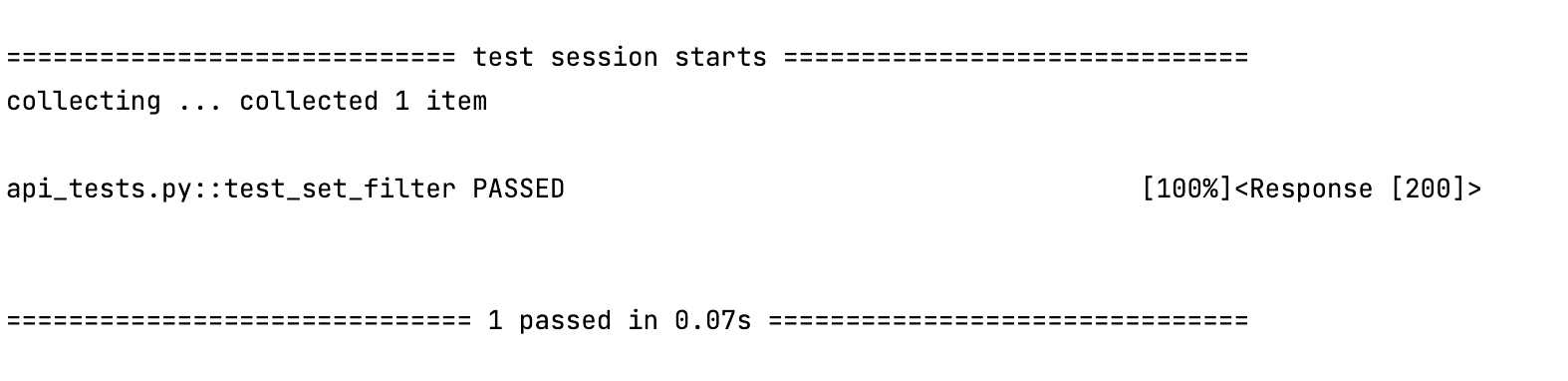
(You can use an additional spreadsheet for this section as well)

**Please check the** [**Manual Testing Report**](https://docs.google.com/spreadsheets/d/1Rp2VXKTRBPIK_enQxC9g82KAob-k72zar-PlBoH-l1c/edit#gid=0) **for testing cases.**

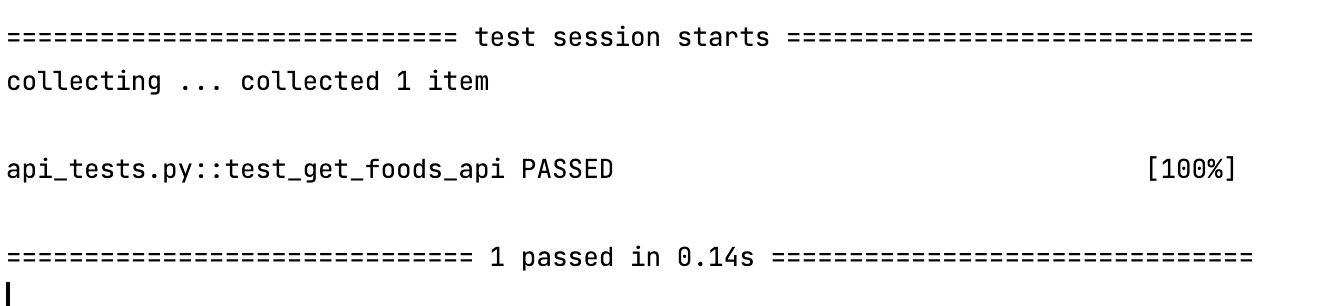
# Automated Testing Report

We Have implemented automated testing for several of the API endpoints using Pytest. For most of the user interface and database components we do not intend to implement any automated testing suites.

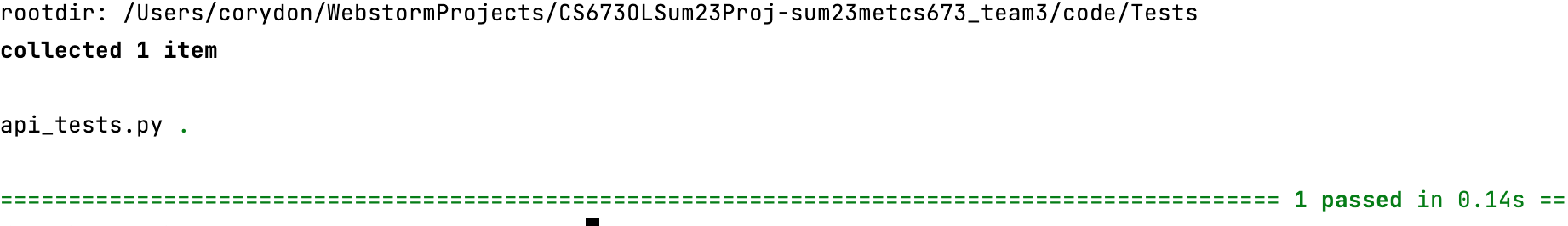
1. **Test Set Filter API**

* Old
* Test items: foods/set-filter API Endpoint
* Medium
* Input: "filter": "week"
* Expected output: 200 status
* Actual output: 200 status
* Pass rate 99%

1. **Test GET Foods API**

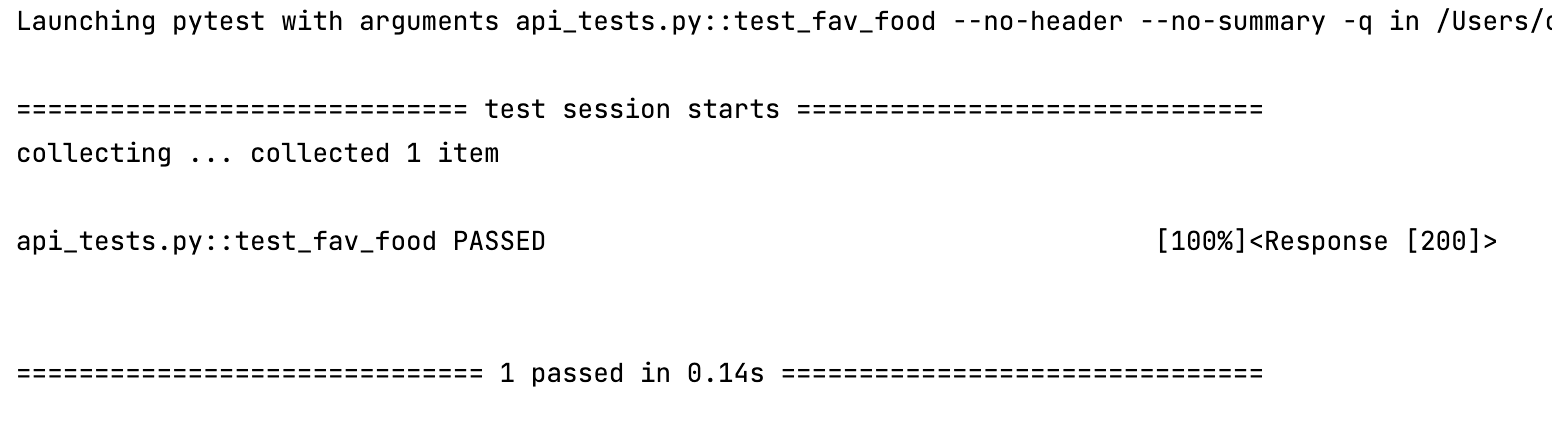
* Old
* Test items: foods/get-food-names API Endpoint
* High
* Input: none GET Request
* Expected output: 200 status, list of foods, first food item is pear, list greater than 0 length
* Actual output: 200 status, list of foods, first food item is pear, list greater than 0 length 
* Pass rate 100%

1. **Test Create Log Entry API**

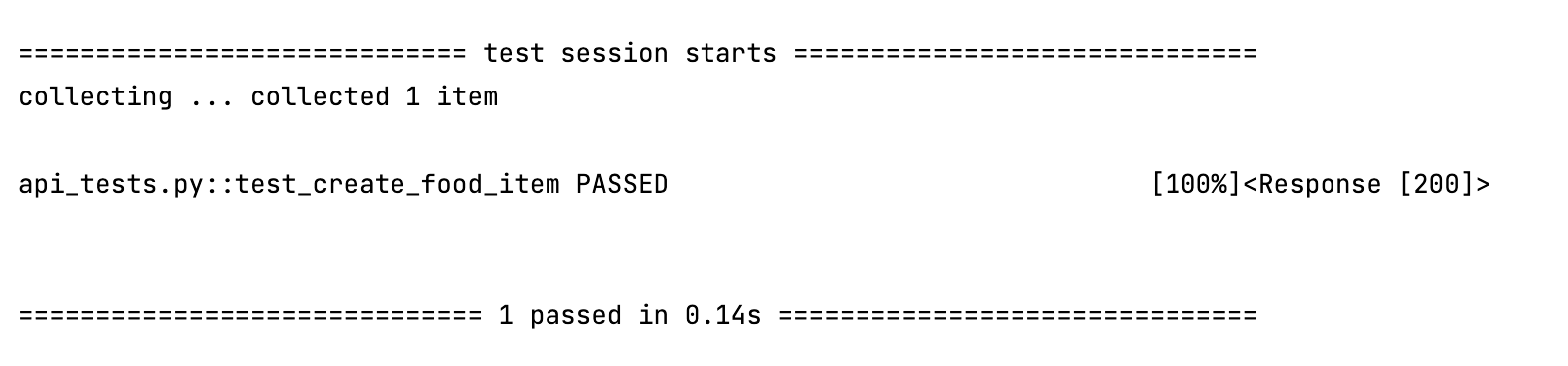
* Old
* Test items: foods/create-log-entry API Endpoint
* High
* Input: {"food\_item\_name": "apple", "num\_servings": "4"}
* Expected output: 200 status
* Actual output: 200 status,
* Pass rate 100%

1. **Test Favorite Foods API**

* New
* Test items: foods/fav-food API Endpoint
* Medium
* Input: POST request
* Expected output: 200 status, “labels” key in response data, “numbers” key in response data
* Actual output: 200 status, “labels” key in response data, “numbers” key in response
* Pass rate 100%

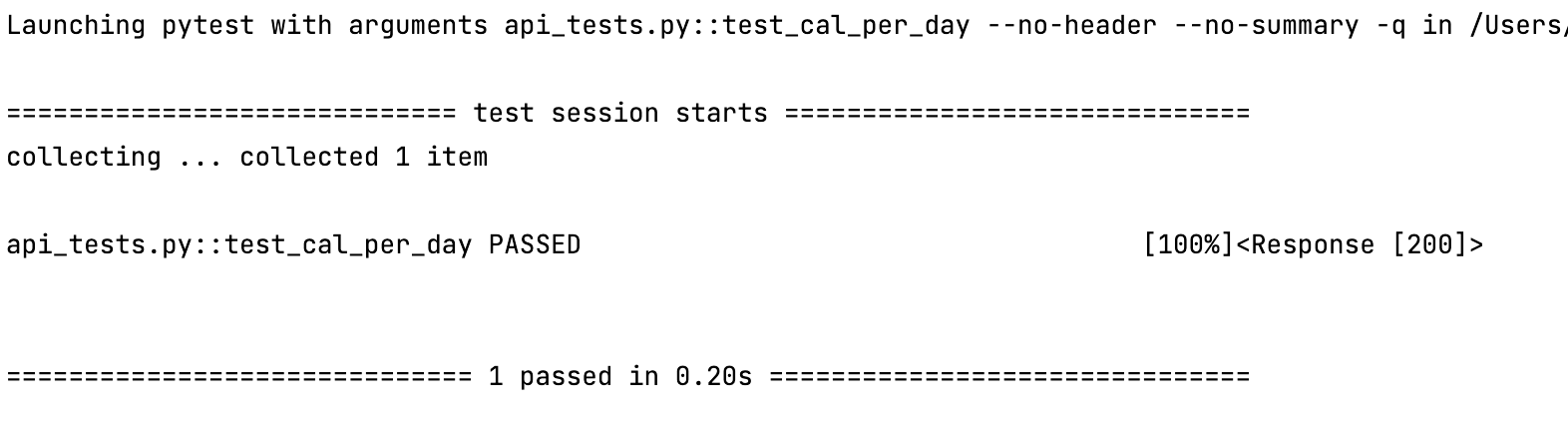


1. **Test Create Food Item API**

* Old
* Test items: foods/create-log-entry API Endpoint
* High
* Input: {"food\_item\_name": "goldfish","calories": "120"}
* Expected output: 200 status
* Actual output: 200 status,
* Pass rate 99%

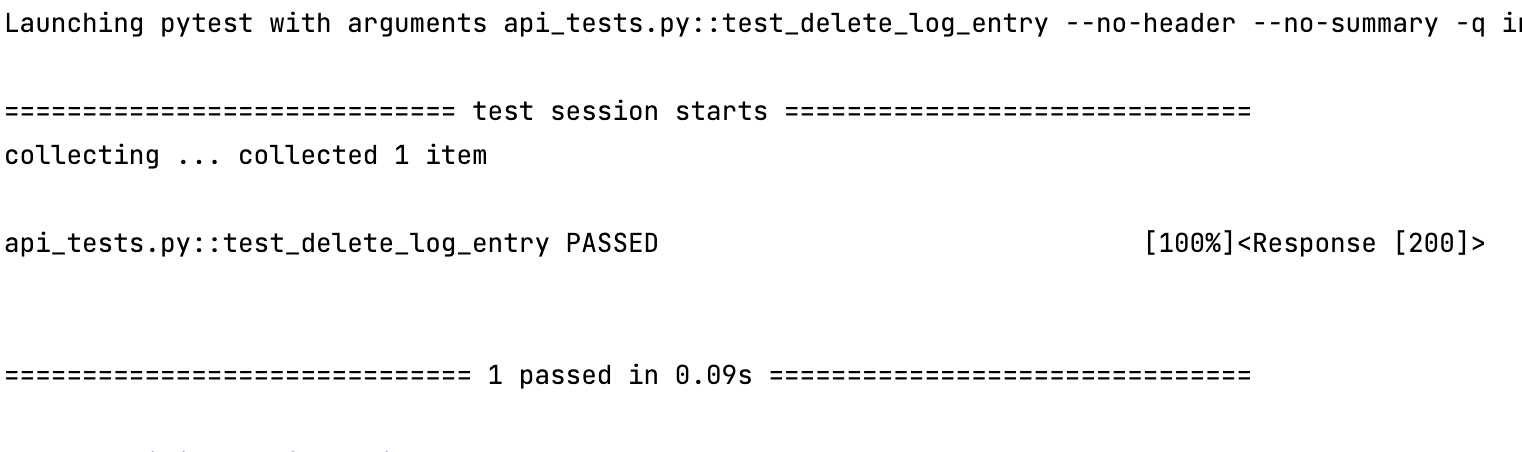
1. **Test Calories Per Day API**

* New
* Test items: foods/cal-per-day API Endpoint
* Medium
* Input: {"food\_item\_name": "goldfish","calories": "120"}
* Expected output: 200 status, “calories” key in response data
* Actual output: 200 status, “calories” key in response data
* Pass rate 100 %



1. **Test Delete Log Entry**

* New
* Test items: foods/cal-per-day API Endpoint
* High
* Input: {"log\_item\_selected":"26"}
* Expected output: 200 status, record removed from database
* Actual output: 200 status, record removed from database
* Pass rate 100 %



# Testing Metrics

In this section, you shall report any metrics used for the evaluation, e.g. # of test cases, test coverage, defects rate, etc.

| **Metrics** | **Number of** | *Pass Rate* |
| --- | --- | --- |
| Test Cases | 23 | 97% |
| Unit Tests | 10 | 99% |
| Defects | 14 | / |
| Defects fixed | 14 | / |
|  |  |  |

Test coverage for key functions *(includes unit testing and manual testing)*: **95%**

# References

<https://www.guru99.com/manual-testing.html>

<https://docs.python.org/3/library/unittest.html>

<https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Testing>

<https://docs.djangoproject.com/en/3.2/topics/testing/>

# Glossary