**CS673 Software Engineering** 

**Team 6 - Blockbuster**

**Software Test Document**

| Team Member | Role(s) | Signature | Date |
| --- | --- | --- | --- |
| Alex Flinchum | QA Lead | *Alex Flinchum* | 09/22/2024 |
| Ricky Zheng | Configuration Leader | *Ricky Zheng* | 09/22/2024 |
| Rekik Mengstu | Requirement Leader | *Rekik Mengstu* | 09/23/2024 |
| James Zheng | Security Leader | *James Zheng* | 9/23/2024 |
| Joshua Shilts | Leader | *Joshua Shilts* | 9/23/2023 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Revision history**

| **Version** | **Author** | **Date** | **Change** |
| --- | --- | --- | --- |
| **1.0.0** | **Alex** | **9/23/2024** | **Creation** |
|  |  |  |  |

[Testing Summary](#_heading=h.gjdgxs)

[Manuel Tests Reports](#_heading=h.30j0zll)

[Automated Testing Reports](#_heading=h.1fob9te)

[Testing Metrics](#_heading=h.3znysh7)

[References](#_heading=h.2et92p0)

[Glossary](#_heading=h.tyjcwt)

# Testing Summary

In this section, you will summarize what was tested, who is involved in testing, testing techniques used, and testing result. You may have the following tests

* + Unit Testing
  + Integration testing
  + System Testing
  + Acceptance Testing
    - All Testing done this Iteration was Acceptance Testing, specifically manual testing.
    - The two test cases tested;
      * 1; Frontend | UI | Create Frontend MVP | Build the Application | Successful
      * 2; Backend | UI | Create Backend MVP | Run the Application | Successful
    - All tests had the expected outcomes.
  + Regression Testing

# Manual Testing Report

In this section, you will give a detailed description of each manual test case performed and the result. If this is a previous You shall list what are existing tests developed in the previous semester and what are new tests developed currently.

* Test case ID, name:
  + 1; Frontend | UI | Create Frontend MVP | Build the Application | Successful
* New or old:
  + New
* Test items: (what do you test )
  + This test focuses on running the frontend from the developers machine
* Test priority (high/medium/low)
  + Medium
* Dependencies (to other test case/requirement if any):
  + Being able to run the project through Npm
* Preconditions: (if any)
  + N/A
* Input Data:
  + Npm start
* Test steps:
  + 1: pull down the latest dev branch
  + 2: Open a terminal (either in your ide or from your machine)
  + 3: cd into the frontend file
  + 4: execute npm start
* Postconditions:
  + None
* Expected output:
  + The fronted should be pulled up in your browser
* Actual output:
  + The input screen is pulled up in my browser
* Pass or Fail:
  + Pass
* Bug id/link: (this should link to your github issue id)
  + None
* Additional notes:
  + N/A
  + Test case ID, name
    - 2; Backend | UI | Create Backend MVP | Run the Application | Successful
  + New or old:
    - New
  + Test items: (what do you test )
    - This test focuses on running the backend through the frontend from the developers machine
  + Test priority (high/medium/low)
    - Medium
  + Dependencies (to other test case/requirement if any):
    - Being able to run the project through Npm
  + Preconditions: (if any)
    - N/A
  + Input Data:
    - Npm start
    - Test data for inputs
  + Test steps:
    - 1: pull down the latest dev branch
    - 2: Open a terminal (either in your ide or from your machine)
    - 3: cd into the frontend file
    - 4: execute npm start
    - 5: insert date into input section
    - 6: Click the Generate button
  + Postconditions:
    - None
  + Expected output:
    - The data should be displayed in a generated text on the web page (At this time an error message will display)
  + Actual output:
    - Error retrieving data
  + Pass or Fail:
    - Pass
  + Bug id/link: (this should link to your github issue id)
    - None
  + Additional notes:
    - N/A

# Automated Testing Report

Describe briefly the automated testing you have done, including where the test code resides in your code repository, what test frameworks are used, and the screen shots or generated testing report.

* + GitHub Actions Pipeline Workflow
    - An initial foundational GitHub Actions pipeline workflow has been implemented where 3 jobs run after every commit on each project branch:
      * Docker-and-Python-Flask-Service
        + Python Flask is installed.
        + Docker is used to build and run the Python backend service.
        + Test checks if Docker image and container are created and can be manipulated and shutdown.
      * Node-Service
        + Node.js is installed along with npm.
        + npm is used to launch frontend web service.
        + Test checks for correct Node.js and npm installation and for a web browser to launch on a specific port.
      * Docker-and-Node Service
        + Node.js is installed along with npm.
        + Docker is used to build and run frontend web service.
        + Test checks if Docker image and container are created and can be manipulated and shutdown.
    - The workflow file, “current-workflow.yml”, is located in the “.github/workflows/” folder at the top level of the dev branch.

# Testing Metrics

Every test case should have it desired outcome; if the desired outcome is meet then the test will pass

There will be some test cases that look for a successful outcome and some that will check that a failed outcome is not possible.

# References

# Glossary