Software Test Plan for Visulyfe (Group #4)

1. Introduction (Ryan)

Software testing is the critical process through which a software developer verifies and evaluates the product or application.

1.1. Purpose (Washika)

a. To general verification and validation process.
b. To ensure that The software product is Bug-Free and to improve the software quality.
c. To justify the level of confidence in how critical the software is to an organization.
d. To find the missing requirements, errors, and gaps in comparison to the actual requirements.
e. To evaluate and improve the quality of software products.

1.2. Scope

a. The scope of a software testing is what areas are needed to get tested on a customer's product , and what functionalities do we need to focus on.
b. The test scope describes the limitations of the testing process, that is what will be tested, how it will be tested, and the result should be the outcome in a high level of customer product quality.
c. The test scope meets a successful test system that performs incorrectly.

1.3. References

a. Visulyfe requirements documents test case: to evaluate the functionality of a particular feature.
b. Test plan: overview of all the activities.
c. Test scenario: clearly outlines of the software testing.
d. Kaggle public API dataset.
e. Datawrapper visualizer tool: used as inspiration for our project.

2. Test Strategy (Fernando)

2.1. Objective

a. The objective is to outline capabilities and faults within the web application, ensuring a clear, available, and positive user experience.

2.2. Approach

a. Combination of manual and automated tests to determine the functionality, stability, and performance of the project.

2.3. Tools

a. Splinter for easy automation test scripts for web applications.
b. Selenium for browser UI interaction for python.
c. Locust as a tool used for Load testing.
d. Pandas is a tool used for our data structures and data analysis, simplifying our data analysis workflow.

2.4. Environments

a. Strictly for testing development environments that make it easier for organization, development cycle processes, and pushing functional code to main

2.5. Entry and Exit Criteria

A. Entry:
    a. Test Graphical functionality with datasets given with testing tools formated.
    b. Test available dataset retrieval.
B. Exit:
    a. Successful execution of testing.
    b. Completion of features with accurately represented data.

3. Scope of Testing (Steven)

3.1. Functional Testing

A. Unit Testing
    a. Developers will perform unit testing on individual components of our web application to ensure they work correctly or to detect faults and defects.
B. Integrated Unit Testing
    a. Developers will collaborate to ensure that the interaction and interfaces between each integrated component work accurately, correctly, and without faults or defects.
C. Regression Testing
    a. Developers will test existing features to ensure they have not been negatively affected when new features are added or modified.

3.2 Non-Functional Testing

A. Coding Standard Testing:
    a. Developers will test as they code to ensure consistency, readability, maintainability, and efficiency across the codebase.
B. Security Testing:

    a. Developers will run tests to assess the system's ability to protect data and to identify vulnerabilities.

C. Usability Testing:
    a. After features are developed and visually integrated, UX/UI designers will assess the user interfaces against the group's standards.

D. Performance and Load Testing:
    a. Developers will conduct evaluations on the system's functionality across different conditions like scalability and response time.

4. Consideration of Infrastructure (Nick)

4.1. Hosting Environment

a. Evaluate the hosting environment for scalability and performance, considering factors such as CPU, RAM, storage, and network bandwidth.

b. Considering cloud-based hosting services (Amazon Web Services, Microsoft Azure, Google Cloud Platform) for flexibility and scalability.

4.2. Data Storage and Processing

a. Assess database requirements for storing and processing large amounts of data for visualization
b. Consider different scalable and efficient database solutions (PostgreSQL, MySQL, MongoDB) depending on the specific needs of the software
c. Implement regular data backup procedures to ensure data integrity and emergency recovery capabilities. Consider data encryption and security to protect sensitive information

5. Risks or Mitigation Plan (Geo)

5.1. Risks

A. Security:
    a. Unauthorized access to sensitive data, injection attacks and data breaches.

B. Performance:
    a. Slow responsive times,poor scalability, degraded performance under heavy loads.

C. Data Integrity:
    a. Data corruption, loss, or inconsistency due to software bugs, hardware failures,or human errors.

D. Usability:

a. Poor experience , confusing interface, and difficulties in interpreting graph data.

5.2. Mitigation

A. Security:
   a. Implementing strong authentication mechanisms, such as a multi-factor authentication (MFA) to ensure that only authorized users can access.

B. Performance:
   a. Perform load testing and stress testing to identify performance bottlenecks and scalability limitations.
   b. Optimize database queries, indexing and caching mechanisms to improve query performance and reduce server load.

C. Data Integrity:
   a. Implement data validation checks to ensure the integrity and consistency of collected data.
   b. Implement data backup and recovery procedures to prevent data loss and ensure business continuity in case of failure.

D. Usability:
   a. Design intuitive and user friendly interfaces with clear navigation, visual cues, and descriptive labels.
   b. Offer customization options to allow users to tailor graphs to their preference.

6. Resourcing (Jacob)

6.1. Team Composition

a. 1 Test Manager
b. 1 Test Engineer for functional testing
c. 1 Database Test Engineer
d. 5 Dev Test Engineers

7. Milestones and Deliverables (Jacob)

7.1 Milestones:

Sprint 24.0.1 (1/26 - 2/1)

Learn how to use and access GitHub repositories
Brainstorm ideas for the project

Sprint 24.0.2 (2/2 - 2/8)

Create project proposal
Decide which of the 2 proposals to pursue for the project

Sprint 24.0.3 (2/9 - 2/15)
    Create backlog in Github
    Created a general scrum schedule
Sprint 24.1.1 (2/16 - 2/22)
    Prepare for STP document
    Start making roadmap
Sprint 24.1.2 (2/23 - 2/29)
    Created STP document
    Updated roadmap
Sprint 24.1.3 (3/1 - 3/7)
    Create Software Requirements Document (sections 1,2,3,5,6, **skip 4**)
Sprint 24.1.4 (3/8 - 3/14)
    Setup for testing tools
Sprint 24.1.5 (3/15 - 3/21)
    Testing for Data integrity across web application
Sprint 24.1.6 (3/22 - 3/28)
    Retesting with splinter for web app communication accuracy
Sprint 24.2.1 (3/29 - 4/4)
    Testing with Selenium for UX access and functionality
Sprint 24.2.2 (4/5 - 4/11)
    Testing for user profile information safety and security
Sprint 24.2.3 (4/12 - 4/18)
    Testing with Locust for load stress tests on servers.
Sprint 24.2.4 (4/19 - 4/25)
    User testing for compatibility and comprehension
Sprint 24.2.5 (4/26 - 5/2)
    Testing with splinter, Selenium, and Locust for respective areas.
Sprint 24.2.6 (5/3 - 5/10)
    Final review testing for v1.0 release

7.2. Deliverables

a. Bug reports
b. Reports of Test Completion
c. Database Integrity and Consistency Report
d. Recommendations for User Experience