Software Test Plan for Insight Write (Group #7):

Tony Lau, Hoang Le, Miguel Gonzalez Torres, Keyvan Mahmoodzadeh Kani, Hayk Vardapetyan, Jian Verdad, Will May, Niyusha Zarnegar

1. Introduction

1.1. Purpose: This document outlines the software testing methodologies and approaches to ensure the functionality, integration, and performance of Insight Write, a web-based platform for personal journaling and data related to those journal entries, and a meditation section to allow users to meditate consistently

1.2. Scope: The testing encompasses all essential functionalities necessary for the journaling web application. This includes CRUD (Create, Read, Update, Delete) operations for journal entries, users' security and reliability when logging in to logging out of their user profile. Furthermore, we'd like to integrate YouTube videos and audio into our meditation section of the application

1.3. References:

a. Python APIs
b. PostgreSQL APIs for inserting, updating, deleting, and querying data
c. Various tool and framework documentation (specifics can be added)
d. Insight Write repository and related documentation
e. Pytest

2. Test Strategy

2.1. Objective: Provide all users a dependable and error/bug-free experience when using Insight Write while remedying any issues that arise on the user end. Furthermore, ensures that the user can easily watch videos or listen to audio(s) without lag or latency

2.2. Approach: Unit & Regressions tests that will test the basic functionality of Insight Write and various unusual edge cases that should not cause disruption in any way

2.3. Tools: The testing tools that will be utilized are:

1. Unit Testing: Testing all the individual units of the source code
    a. Pytest (Testing various functions by inserting test parameter values)

2. Integration Testing: Developers collaborate to verify the interactions and interfaces of all our application components. This will be performed as each story component is added during the middle and end of each sprint and after individual units tests have passed

3. System Testing: Development team verifies the integrity of the entire system, all the active components, and functions correctly. Testing as completed features are integrated, towards the end of the planning increment

4. Regression Testing (Ensuring that the entire application is not negatively affected by any modified or new features)

5. Security Testing: User authentication via email verification and encryption methods for journal entries, passwords, etc.

6. UX UI Testing: Responsiveness, page scaling, seamlessly inputting data from the user's point of view

7. End - to - End Testing: Towards the end of the product production, fully verify the functionality and performance of the entire software application from start to finish by simulating real-world user scenarios and replicating live-data

2.4. Environments

- Development Environment
    - Regression Testing Environment
    - Security Testing Environment
    - Accessibility Testing Environment (Cross Browser Testing)

2.5. Entry and Exit Criteria

- Entry:
    - Setting up testing environments
    - Database configuration environments
    - Updating Data Test
    - Browser Compatibility
    - User Account

- Exit:
    - Test Case Execution
    - Detect and Fixing Errors/Defects
    - Test Summary Report and Review
    - Approval

3. Scope of Testing

- 3.1. Functional Testing
    - User Interface Testing
    - User Registration and Authentication (validate users' login and sign-up process)
    - Navigation Testing (check if the website, buttons, links, etc will be navigated smoothly by the users)
    - Content Verification (ensure entries and images are displayed correctly)

- 3.2 Non-Functional Testing
    - Connecting the frontend with the backend
    - Ensuring data is requestable and storable
    - Security measures (Encryption of data such as journals, passwords, and accounts)
    - Usability Testing (Evaluating users' experience on the website)
    - Network Testing (Evaluating website's performance under different network conditions)

4. Consideration of Infrastructure

- 4.1. Server Configuration: The test will be conducted on a test server with specifications suitable for hosting the web application

- 4.2. Database: The test will be conducted on a test database with regular backups and rollback capabilities

- 4.3 Source Code and Documentation Backups: Considering backing up all documentation, source code, and other related infrastructure in case anything may be lost

5. Risks or Mitigation Plan

- 5.1. Risks

    - Login access vulnerability/leaks
    - SQL injection attacks
    - SSH exploit/hacks (Secure Socket Shell)
    - Data loss during CRUD operations. (Create, Read, Update, Delete)
    - Unauthorized access to users' journals and sensitive info

- 5.2. Mitigation
    - Regular data backups
    - Authentication and verification mechanisms
    - Journal entry encryption
    - Maintain strict access rights
        - 2 factor-authentication, robust password requirements
    - Keep external services patched and up-to-date to prevent many likely cyber attacks

6. Resourcing

- 6.1. Team Composition
    - 1 Test Manager
        a. There will be a new Test Manager every week (i.e. the Product Owner will be the Test Manager
    - 5 Test Engineers
    - 3 Database Engineers

- Note: Each sprint will have a new Test Manager and a set of Test & Database Engineers

7. Milestones and Deliverables

1) Note: We will update this section for every sprint. Sprints beyond the current one are subject to change based on the results of the current sprint

2) Points to follow for this section:
    - Should develop database integrity and consistency report
    - Must have a positive emotional impact on users
    - Ongoing development using user feedback as a basis (includes fixing bugs)

3) Reflecting on PI 23.0 (What went well, What to Improve)
    a) What went well:
        i) Conducted weekly & consistent meetings
        ii) Chosen our platforms & frameworks for the project
        iii) Backlogged all of our epics & features on GitHub

    b) What to improve:
        i) Making more efficient use of our time during meetings
        ii) Balancing documentation and the project
4) Reflecting on PI 23.1 (What went well, What to Improve)
    a) What went well:
        i) Splitting up work
        ii) Completing work and review
        iii) Consistently covering everything that needs to be discussed during meetings

    b) What to improve:
        i) More careful consideration on what to add to the product backlog

ii) Balancing schoolwork and the project

5) Reflecting on PI 23.2 (What went well, What to Improve)
    a) What went well:
        i) Great research, leading to great learning experiences
        ii) Great work on the front-end
        iii) Research

    b) What to improve:
        i) Development environment
        ii) Creating manageable tasks
        iii) Allocating more time to documentation

- 7.1 Milestones
    - Increment 24.1
        - Sprint Week 1:
            - Setting up Planning Increment (PI) 23.0 Review, 23.1 Plans, & 23.2 Forecast
            - Setting up Software Test Plan (STP)

        - Sprint Week 2:
            - Finish the Software Test Plan (STP)
        - Sprint Week 3:
            - Finish sections 1, 2, 3, 5, 6 of the Software Requirements Specification document
            - Create a presentation covering our work on the SRD.
        - Sprint Week 4:
            - Add data flow diagrams (DFDs) to sections 2.2, 2.3, and 4.1. Finish section 4.1.
        - Sprint Week 5:
            - Finish sections 1 - 5 and 8 - 11 of Software Design Document
            - Create presentation about SDD
            - Create web pages for login/sign-up, journal, and meditation sections of the website.
        - Sprint Week 6:
            - Create user input fields for login
            - Set up fonts, colors, and scheme of the website
            - Set up a PostqreSQL database
            - Journal entry creation
            - Journal entry stored in database
    - Increment 24.2
        - Sprint Week 7:
            - Update SRD and SDD to sync with Github Features and Stories
        - Sprint Week 8:
            - Update Journal entry creation
            - Create user input fields for signup and login
            - Setup designs, fonts, colors, etc.
            - Basic functional javascript for pages
            - Google calendar API integration
            - Research hosting databases on local machine
        - Sprint Week 9:
            - Another Update to Journal entry creation
            - Create user input fields for signup and login
            - Set up designs, fonts, colors, etc. for front-end
            - Verify user login credentials
            - Add google login
            - Basic functional javascript for pages
            - Google calendar API integration
            - Mini draggable calendar in journal entries
            - Add object classes
            - Railway Deployment and implementation
            - Update documentation

        - Sprint Week 10:
            - Hopefully the final update needed for Journal entry creation
            - Verify user login credentials
            - Basic functional javascript for pages
            - Google calendar API integration
            - Add object classes
            - Update documentation
            - Display weather information of an inputted location
            - Redesign web page and link pages of application together to dashboard
            - Save journal entries to database
            - Create tables and intractability with all of our models from Django to the database
            - Railway Deployment and implementation
        - Sprint Week 11:

        - Sprint Week 12:

- 7.2. Deliverables
    - Increment 24.1
        - Sprint Week 1:
            - Completed our Planning Increment (PI) plan & will utilize the Software Test Plan (STP) for testing purposes

        - Sprint Week 2:
            - STP finished. Will continue filling in the milestones and deliverables sections while the project continues.
        - Sprint Week 3:
            - Finished the sections and the presentation
        - Sprint Week 4:
            - Added data flow diagrams (DFDs) and finished section 4.1 with all members contributing.
        - Sprint Week 5:
            - Finished sections 1 - 5 and 8 - 11 of Software Design Document
            - Created the presentation that covers the SDD as well as the web pages.
            - Created web pages for login/sign-up, journal, and meditation sections of the website.
        - Sprint Week 6:
            - Created user input fields for login
            - Journal entry creation is now possible

    - Increment 24.2
        - Sprint Week 7:
            - Updated SRD and SDD to sync with Github Features and Stories
        - Sprint Week 8:
            - Updated Journal Entry Creation again
            - Created user input fields for signup and login
        - Sprint Week 9:
            - Designs, fonts, colors, etc. all set up for web pages.
            - Google authentication login possible
            - Mini draggable calendar added to journal entries
        - Sprint Week 10:
            - Updated Journal Entry Creation for hopefully the final time
            - User login credentials verified
            - Basic javascript on pages are functional
            - Object classes all completed
            - Weather information of an inputted location displayed correctly
            - Journal entries save to database
            - Railway deployment and implementation completed
        - Sprint Week 11:

-
- Sprint Week 12: