Test Plan: Harmony

Change Log

Version	Change Date	By	Description
1.0	2023-10-19	Alex Senden & Andrii Provozin	Initial Draft
1.1	2023-11-30	Alex Senden & Andrii Provozin	Load Testing Plan

1 Introduction

1.1 Scope

The Harmony test plan covers unit, integration, regression, acceptance, and load testing. The application's frontend, backend, and database (through integration tests) will be tested.

Harmony has 6 core features: Creating Posts, Account Management, Interacting With Posts, Interacting With Others, Interacting With Artists/Albums/Songs, and Searching for Content. These distinct features will all be individually tested to ensure the quality of the application.

Additionally, Harmony has one main non-functional feature: The system shall be able to maintain 200 concurrent user sessions, responding to 1000 requests per minute. This non-functional feature will also be individually tested to ensure the performance of the application.

1.2 Roles and Responsibilities

Name	Net ID	GitHub Username	Role
Alex Senden	sendena@myumanitoba.ca	alexsenden	Architecture Designer, Regression Test Lead
Andrii Provozin	provozia@mymanitoba.ca	Developik	Deployment Manager, Load Test Lead
Chris Baran	baranc@myumanitoba.ca	Dankgen	Database Administrator
Edden Wong	Wonge5@myumanitoba.ca	EddenWong	Frontend Design Lead
Ikram Khan Shipon	shiponik@myumanitoba.ca	moiikram	Frontend Implementation Manager

Architecture Designer: Make final decisions on application architecture, such as the layers within the frontend and backend, and key frameworks.

Database Administrator: Make important decisions regarding the database, including schema design and the database's interactions with the backend.

Deployment Manager: Oversee the development of the CD pipeline and the integration with a web hosting provider.

Frontend Design Lead: Oversee the design decisions of the frontend, including creating mock-ups and planning the user interactions in the application.

Frontend Implementation Manager: Translate the frontend designs and plans into dynamic, reusable frontend components.

Regression Test Lead: Oversee the development of unit and integration tests, and the development of the CI pipeline.

Load Test Lead: Oversee the architecture and development of load testing suite.

Note: All team members contribute to all aspects of the project, however each individual takes on an area of focus and responsibility for their role.

2 Test Methodology

2.1 Test Levels

Acceptance Testing: Acceptance tests will be conducted via end-user testing on a user story-basis according to the acceptance criteria in the user story.

Regression Testing: The CI pipeline will run all unit and integration tests, and will require them to pass before allowing changes to be pushed to the main branch.

Load Testing: Load tests will be run manually from a computer using installed JMeter or via invoking github actions pipeline that would run it.

1. **Creating Posts:**
    - Unit Tests: Test new post validation logic in the backend at the function level.
    - Integration Tests: Test the seams regarding post creation starting with the API call made by the PostModal in the frontend, through the backend, into the database.
2. **Account Management:**
    - Unit Tests: Test user registration logic and validation, login logic and validation, as well as account information modification validation at the function level.

- Integration Tests: Test the seams regarding registration, login, and account information modification starting with the API call made by the corresponding component in the frontend, through the backend, into the database.
- Load Testing: Test Login, Logout, Account page endpoints to make sure that they fullfill the basic requirement.

3. **Interact With Posts:**
    - Unit Tests: Test "Like" creation logic, comment creation validation, and post retrieval logic at the function level.
    - Integration Tests: Test the seams regarding "Like" creation, comment creation, and post retrieval starting with the API call made by the Post or PostFeed component in the frontend, through the backend, into the database.
    - Load Testing: Test Like, Post Retrieval endpoints to make sure that they fullfill the basic requirement.
4. **Interact With Others:**
    - Unit Tests: Test user retrieval logic and follow/unfollow logic at the function level.
    - Integration Tests: Test the seams regarding following and unfollowing a user, and user retrieval from the database, starting from the corresponding API calls on the profile page, through the backend and to the database.
    - Load Testing: Test user retrieval endpoint to make sure it fullfills the basic requirement.
5. **Interact With Albums/Artists/Songs**
    - Unit Tests: Test the logic for following and unfollowing artists, albums, and songs at the function level.
    - Integration Tests: Test the seams regarding following and unfollowing artists, albums and songs, as well as retrieving posts from the database that relate to a specific artist, album, or song, starting with the API call made by the artist, album, or song page, into the backend, and into the database.
    - Load Testing: Test getting information about artists, albums, and songs endpoint to make sure it fullfills the basic requirement.
6. **Searching For Content**
    - Unit Tests: Test the logic for searching for user, artist, album, and song profiles at the function level.
    - Integration Tests: Test the seams regarding the retrieval of users, artists, albums, and songs starting in the search bar component, through the backend, and through to the database.
    - Load Testing: Test searching for various content to make sure it fullfills the basic requirement.

2.2 Test Completeness

Code Coverage: The backend will have 100% code coverage.

Feature Coverage: Every feature will have at least 10 unit tests.

Integration Coverage: There will be at least 10 integration tests.

User Story Acceptance: Each user story will have 1 corresponding acceptance test.

Regression Testing: Regression tests will be run and must succeed before any change can be pushed to the main branch.

Load Testing: Load tests will cover basic user functions to test the capacity requirement of at least 20 users and 200 requests per minute concurrently.

3 Resource & Environment Needs

3.1 Testing Tools

Requirements Tracking: GitHub Issues (Acceptance Criteria is listed alongside each user story)

Bug Tracking: GitHub Issues

Unit/Integration Testing Framework & Code Coverage: Jest

Unit/Integration Testing Libraries:

Frontend: Mock Service Worker, React Testing Library

Backend: Supertest

Regression Testing Workflow Automation: GitHub Actions

Load Testing: Apache JMeter

3.2 Test Environment

Regression Testing Environment: GitHub Actions using Ubuntu 22.04

Acceptance Testing Environment: Windows 11 using Google Chrome (latest version)

Load Testing Environment: Windows 11 (if run locally) and Ubuntu 22.04 (pipeline)

4 Terms/Acronyms

TERM/ACRONYM	DEFINITION
API	Application Programming Interface
CI	Continuous Integration
CD	Continuous Delivery/Deployment