Short-Term Memory System (STeMS) Software Test Plan By AlphaSoft

Team Members

Awasthi, Aaditya

Blavat, Oleksiy

Bond, Matthew

Carter, Mackenzie

Mahbobi, Sayed shah

McAllister, Charlie

McCool, Max

McLaughlin, Taylor

Powers, Michael

Weaver, Daniel

Revision History

Date	Version	Description	Author
17 June 2023	1.0	Initial submission to Client	AlphaSoft
22 July 2023	2.0	Updated to match current state of project.	AlphaSoft
5 August 2023	3.0	Updated to match current state of project.	AlphaSoft

Table of Contents

Figures & Tables

1. Introduction

1.1. Purpose

This document acts as the Software Test Plan (STP) for the architectural prototype of the STeMS (Short-Term Memory System).

This document supports the following objectives:

1. List high level recommended test requirement.
2. Specify the testing strategy and any recommendations.
3. List required resources.
4. Estimate total testing effort.
5. List the test activities' deliverable components.

1.2. Project Overview

Natural Language Processing aims to combine the rules of human language with artificial intelligence and deep learning models, giving computers the ability to process large quantities of human language data. ChatGPT is one the leading generative AIs utilizing this technique in order to create written language responses to user given prompts. AlphaSoft is seeking to explore the different application areas for ChatGPT that would benefit those struggling with short-term memory loss via the Short-Term Memory System (STeMS.) Combined with the natural language processing capabilities of ChatGPT, the STeMS will give users recollection assisting tools for in person conversations.

1.3. Project Documents

This STP will be included in the documentation portion of the project deliverables. All documentation created throughout the project lifecycle is done so with the intent to assist in the understanding, implementation, and maintenance of both the mobile application and browser extension.

The following documents will be included as part of the project deliverables.

Document	**Version**	**Date**
Project Plan (PP)	4.0	5 August 2023
Software Requirements Specification (SRS)	4.0	5 August 2023
Technical Design Document (TDD)	3.0	5 August 2023
Software Test Plan (STP)	3.0	5 August 2023
Programmer Guide (PG)	2.0	5 August 2023
Deployment and Operations Guide (DOG)	2.0	5 August 2023
Software Test Report (STR)	1.0	5 August 2023
User Guide (UG)	1.0	5 August 2023
Traceability Matrix (TM)	1.0	5 August 2023

Table 1: Project Documents

1.4. Acronyms, Definitions, and Abbreviations

Throughout this STP, a variety of terms and acronyms specific to the proposed application are used. For clarity, their definitions are provided below:

Term	**Definition**
AUT	Application Under Test
QA	Quality Assurance
SRS	Software Requirements Specification
STeMS	Short-Term Memory System
STP	Software Test Plan
STR	Software Test Report
TM	Traceability Matrix. A document that traces defects and test cases back to their requirement.

Table 2: Acronyms, Definitions and Abbreviations

2. Scope

All “must have” requirements for proper application functionality will be covered in the first development phase. The first testing phase will be conducted in parallel with development efforts. By the completion of phase one, the QA team should be capable of the following:

1. Create detailed manual test cases
2. Save all manual test cases
3. Reference manual test cases during testing efforts
4. Record testing results
5. View testing results

2.1.In-Scope

The application will allow users to create and manage live conversation recordings and select how they should be processed. An additional browser extension that allows the application to use processed recordings to populate webpage forms. The following features will be covered by this STP and resulting manual test cases:

1. Recording
2. Edit Recording
3. Delete recording
4. Process recording
5. Additional browser extension that will allow the application to use processed recordings to populate webpage

2.2.Out-of-Scope

This STP covers only the testing efforts required for the front-end features of the application. Back-end feature testing will be the responsibility of the corresponding test team for Team B.

The AlphaSoft QA team will coordinate with Team B to ensure proper integration, where deemed appropriate.

2.3.Risk Mitigation

Risk Area	Risk	Prob.	Impact	Mitigation Plan
Schedule	The testing timeline is limited. The test cannot be prolonged past the UAT scheduled start date if the start of the testing is delayed because of design work.	High	High	The preparation duties and early communication with concerned parties can be managed by the testing team. Although not as much as recommended by best practices, some buffer has been included to the timeline for contingencies.
Resources	Not enough resources, resources onboarding too late the process takes about 15 days.	Medium	High	Holidays and vacation time have been budgeted for and included in the timetable; variations from the budget could result in testing delays.
Defects	Defects are detected late in the cycle or at a late point of the cycle; late Defects are most frequently the result of unclear requirements and take a long time to fix.	Medium	High	Defect management plan is in place to ensure prompt communication and fixing of issues.

Risk Area	Risk	Prob.	Impact	Mitigation Plan
Scope	completely specified scope	Medium	Medium	The scope is clearly specified, but the functional modifications are either still being made or are constantly changing.
Scope	Lack of accessibility and the absence of an Independent Test environment	Medium	High	The timetable is affected and the beginning of the test execution will be delayed as a result of the environment not being available.
Scope	Due to new issues testing delayed	Medium	High	There is a good likelihood that some "new" problems will be found during testing and that they'll develop into a problem that will take some time to fix. The overall schedule will be impacted if any issue becomes showstopper. Defect management and issue management process is in place if get any defect.

Table 3: Risk Mitigation

2.4.Quality Objective

1. Ensure that the AUT conforms the functional and non-functional requirements.
2. Ensure the AUT meets the quality specifications defined by the client.
3. All defects are documented and tracked within the Software Test Report (STR)
4. To win customers' trust by offering them a high-quality product
5. To ensure that AUT complies with the SRS.

2.5.Roles and Responsibilities

Role	Responsibilities
Test Manager	Oversees the overall testing effort for the project, gathers the necessary resources, and provides project guidan
QA Analyst	Responsible for all quality of the project to make sure product meet the requirement and work as per requirement
Developer	Will be responsible for developing the product as per requirement.
Project Manager	Oversees the overall end to end project effort and gathers the necessary help for resources and provides the guidance.

Table 4: Roles and Responsibilities

3. Testing Strategy

3.1. Test Objectives

This test plan is written by and with the intent to be used by the Alpha Soft Test Team as a means for verifying proper functionality of the STeMs application and as a part of all regression efforts. The AlphaSoft Test team is responsible for testing all front-end features of the application in accordance with the use-cases outline in the Software Requirements Specification document. Four testing levels are outlined within this test plan, including unit, functional, user acceptance, and regression testing. All test cases will be logged within a test repository tool chosen by the AlphaSoft Testing Manage based on its level of appropriateness to the development process. The test cases cover both common user functions as well as edge cases that may occur throughout the application's lifecycle.

All test case executions and results will be recorded by the conducting tester and defects will be reported to the development team for further investigation. All test results as part of regression testing prior to release will be recorded and included within the Test Report deliverables. These test cases will be conducted manually using an installed version of the STeMs application by the assigned test engineer listed in subsequent sections of this document.

3.2. Test Assumptions

All testers for the AlphaSoft team are assumed to have a standardized, and functional testing environment at their disposal. Front-end features submitted for testing are expected to be fully integrated with their back-end counterparts or with the need for test drivers specified within the testing steps. Any back-end features included as part of an item for test are assumed to have been fully tested by the responsible party before being integrated by the AlphaSoft development team. All features approved for integration are expected to have undergone both unit and functional testing. All approved releases have undergone full regression testing with defects noted within the proper documentation. User acceptance tests are assumed to have been discussed and preapproved will the appropriate parties before being submitted to this test plan.

3.3. Data Approach

All data for the verification and the validation of the STeMS application has been created for the sole purpose of testing. Any and all recordings are staged, example data created to ensure full code coverage. All members of the AlphaSoft Testing team will have the same sample set of data to ensure standardization across multiple testing environments.

All test cases will be documented and stored within a manual test case repository tool selected by the team's testing manager. All completed test cases will be documented along with any resulting defects in the appropriate reporting documents.

3.4. Levels of Testing

Test Type	Description	Responsible Parties
Unit Testing	New feature will be tested for proper functionality before integration.	AlphaSoft Test Team
Functional Testing	Validate that all features work in relation to each other, as specified in requirements documentation	AlphaSoft Test Team
User Acceptance Testing	The Testing Manager and Business Analyst will act as Customer representatives to conduct agreed upon acceptance tests to ensure delivered product meet specified requirements	AlphaSoft Test Manager and Business Analysist

Test Type	Description	Responsible Parties
Regression Testing	Full test suite will be run against release candidates for the STeMS application to ensure end-to-end functionality	AlphaSoft Test Team

Table 5: Levels of Testing

3.5. Unit Testing

All features for the STeMS application will undergo unit testing procedures to ensure proper functionality before integration. Features have been broken into tickets based on the use cases described in the SRS document. Each ticket is to be treated as one unit for testing.

Participants:

Tester's Name	Department/ Area	Role
Aaditya Awasthi	QA	Testing Manager
Taylor McLaughlin	QA	Test Engineer
Sayed Shah Mahbobi	QA	Test Engineer
Mackenzie Carter	QA	Test Engineer

Table 6: Unit Testing Participants

3.6. Functional Testing

The integration of new features will require functional testing efforts. The functional tests are designed to ensure the proper collaboration of features and cover all possible paths within the source code.

Participants:

Tester's Name	Department/ Area	Role
Aaditya Awasthi	QA	Testing Manager
Taylor McLaughlin	QA	Test Engineer
Sayed Shah Mahbobi	QA	Test Engineer
Mackenzie Carter	QA	Test Engineer

Table 7: Functional Testing Participants

3.7. User Acceptance Testing

User Acceptance tests have been pre-approved by a customer representative in accordance with the requirements specification document. The AlphaSoft business analyst will conduct the user acceptance tests with the assistance of the Testing Manager to ensure all customer requirements have been met.

Participants:

Tester's Name	Department/ Area	Role
Aaditya Awasthi	QA	Testing Manager
Max McCool	BA	Customer Representative

Table 8: Acceptance Testing Participants

3.8. Regression Testing

Regression Testing will be conducted for every potential release candidate. The full test suite for the STeMS application will be conducted by the AlphaSoft test team to ensure the expected behavior for the application. Any defects identified during regression efforts will be documented and addressed appropriately.

Participants:

Tester's Name	Department/ Area	Role
Aaditya Awasthi	QA	Testing Manager
Taylor McLaughlin	QA	Test Engineer
Sayed Shah Mahbobi	QA	Test Engineer
Mackenzie Carter	QA	Test Engineer

Table 9: Regression Testing Participants

3.9. Automation Testing

We are going to automate critical flow to avoid any regression issues. Automation will be done using the APPIUM automation tool, which will cover all the mobile application scenarios. Any defects identified during automation will be documented and addressed appropriately.

Participants:

Tester's Name	Department/ Area	Role
Aaditya Awasthi	QA	Testing Manager
Taylor McLaughlin	QA	Test Engineer
Sayed Shah Mahbobi	QA	Test Engineer
Mackenzie Carter	QA	Test Engineer

Table 10: Automation Testing Participants

4. Execution Strategy

4.1. Entry Criteria

These are the prerequisites that must be met before testing can start. This is to ensure that the STeMS project and the test environment are prepared for testing.

Entry Criteria	Test Team	Technical Team	Notes
All necessary tools, software, and frameworks have been installed and configured properly.			
A stable software build is available and ready for testing.			
Test cases need to be created and reviewed. Test cases must also have proper steps with expected results. Any changes made must be reviewed and approved before testing can begin.			
Critical errors should be properly identified and resolved before moving on to another test case.			
Test data has been created.			
The test plan for STeMs has been reviewed and approved.			

Table 11: Entry Criteria

4.2. Exit Criteria

These are the conditions that must be met to determine when a test has been completed.

Exit Criteria	Test Team	Technical Team	Notes
All test cases have been executed and are marked as completed.			
All reported critical defects have been resolved.			
Test logs and reports are kept up to date and have a record of executed test cases and their results.			
The software has demonstrated a level of reliability and stability (ex. No crashes, fast performance)			
All test cases must be completed within the given time constraint.			
All errors and defects have been properly dated and documented.			
All test cases have a 90% pass ratio.			

Table 12: Exit Criteria

4.3. Validation and Defect Management

Validation and defect management are essential steps to ensure that testing covers all grounds, and it helps ensure that testing is effective.

4.4. Validation

All test cases must be validated in the following ways:

- Coverage - Test cases must be checked to ensure that the test cases cover all functionality and interaction within the application.
- Verify Expected Results – Check that all expected results align with the expected behavior of the product.
- Review – All QA and the development team have reviewed the test cases and have given feedback.

4.5. Defect Management

During testing all defects should be tracked in a spreadsheet with the following information:

- Defect ID
- Description
- Defect Found Date

It is the responsibility of the tester to perform these steps:

1. Record the defect
2. Inform the development team of the error
3. Re-test if the issue is potentially resolved
4. Update the log if the defect has been resolved

In addition to tracking defects in a spreadsheet, the QA team is also required to track defects in Trello's sprint board. The Trello card will contain the Defect ID and a description of the defect.

Defects found during testing should be categorized as below:

Severity	**Impact**
Critical	These are defects that block all functionality of the product or cause significant damage to the product and need to be addressed immediately.
High/Major	These are defects that affect the major functionality of a product. Major functionality is not operating as it should.
Medium/Moderate	These defects cause functionality to operate slightly differently than expected or impacts a m
Low	These defects have almost no impact on the functionality of the product (ex. Cosmetic glitches and grammatical errors).

Table 13: Defect Severity Levels

5. Environment Requirements

5.1. Test Environments

The testing environment should include the latest version of the STeMS application installed and configured properly. The test environment should have all the necessary tools, software, and frameworks required for testing. A stable software build of the STeMS application should be available and ready for testing. Sufficient test data should be created to cover various scenarios and ensure comprehensive testing. The test environment should be representative of the production environment to accurately simulate real-world conditions.

5.2. Security Requirements

The test environment should adhere to security protocols and measures to protect sensitive data and prevent unauthorized access. Proper access controls and user permissions should be in place to restrict access to the test environment. Any security vulnerabilities identified during testing should be promptly addressed and resolved to ensure the integrity of the application and data.

5.3. Testing Tools

Specify the testing tools that will be used for test case management, defect tracking, and reporting. These tools should be installed and configured in the test environment.

The chosen test repository tool should support the documentation and storage of test cases, test results, and defects. Any additional testing tools or frameworks required for specific types of testing, such as automation testing or performance testing, should be installed and available in the test environment.

Automation tools include Appium for testing and GitHub for storing the tests.

6. Dependencies

6.1. Test-Item Availability

The testing team is dependent on the availability of the STeMS application and its associated features for testing. The development team should ensure that all the necessary components, such as recording, editing, deleting, and processing recordings, are available and accessible to the testing team.

6.2. Testing-Resource Availability

The testing team relies on having adequate resources, including testing tools, testing environments, and skilled testers. It is crucial to ensure that the required testing resources are available and properly set up before testing can begin.

It is important to address these dependencies effectively to ensure a smooth and efficient testing process for the STeMS application.

6.3. Deadlines

The testing effort is dependent on meeting specific deadlines. The start of testing should not be delayed due to design work or other factors. The testing team should coordinate with the development team and other stakeholders to ensure that the testing timeline aligns with the project schedule and that any potential delays are mitigated.

6.4. Integration Testing

There is a dependency on integration testing between the front-end and back-end teams (Team A and Team B). Both teams need to collaborate and ensure proper coordination to conduct seamless integration testing, which includes testing the interaction and communication between the front-end and back-end components.

6.5. Test-Case Creation and Review

The testing team relies on creating and reviewing test cases before testing can begin. Test cases should be comprehensive, covering all the required functionality and interaction within the application. Test cases must be reviewed and approved by the relevant stakeholders to ensure accuracy and effectiveness.

6.6. Test Data

The testing team requires test data to verify and validate the STeMS application. The availability of suitable test data, including staged an example data, is essential to ensure full code coverage and standardization across multiple testing environments.

6.7. Defect Management

There is a dependency on proper defect management. The testing team should track and document all defects found during testing, including their description, priority, severity, associated test cases, date found, assigned developer, fix date, and additional comments. Effective communication with the development team and retesting of resolved defects are also crucial dependencies.

6.8. Security Requirements

The test environment should meet the specified security requirements to ensure the confidentiality, integrity, and availability of the STeMS application and its associated data. The testing team should have access to a secure environment that aligns with the project's security standards.