A black text on a white background

Description automatically generated

SPARTA GLOBAL

FINANCIAL CHATBOT TEST PLAN

for Financial Chat BOT v1.0

1. **Introduction**

**1.1 Test Objectives**

The tests of the financial chat bot should validate both the requirements perspective and business perspective that:

* Ensure that the system provides users with simple, actionable saving tips for managing daily expenses.
* Ensure that users can access clear and detailed information on various financial products, such as savings accounts, loans, and credit cards, to make informed decisions.
* Verify that the chatbot provides advice on creating a basic budget, helping users manage their income and expenses effectively.
* Ensure the chatbot offers users practical, non-personalized advice on reducing debt, with a focus on helping users work towards becoming debt-free.
* Confirm that users receive relevant information on the benefits and risks associated with taking out loans, aiding in their decision-making process.
* Ensure the system clearly explains the differences between savings and checking accounts, helping users manage both short-term and long-term finances effectively.
* Ensure the chatbot provides clear guidance and practical steps on how to build an emergency fund, helping users prepare for unexpected financial needs.
* Confirm that the chatbot offers advice on managing daily expenses such as groceries and utilities, with the goal of helping users identify unnecessary costs and save more.
  1. **Scope of testing**

System testing of the financial chatbot will include the chatbot front end and back end API (packaged within the same project) .

1. **Approach**

**2.1 Assumptions**

* First build of the chatbot system will be ready for integration testing on 27/09/2024.
* Each build of the system will have passed unit testing before applying integration testing.
  1. **Coverage**

Test coverage will be measured by:

* Statement, Branch and Line coverage of unit tests.
* Completed report of testable system requirements and cases.
* All user requirements to be tested.
  1. **Test Tools**
* Intellij Test Coverage for testing coverage of unit tests.
* Rest Assured for testing API functionality.
* Playwright/Selenium for automated web testing framework.
* Trello for documentation of defects found
  1. **Test Type**

The following types of testing will be performed during testing of this product:

* Functional testing, by performing test cases based on testable requirements (achieved through unit and integration testing)
* Functional testing, by performing test cases based on business functions (achieved through unit and integration testing)
* Regression testing, to ensure that any changes to the application do not introduce new defects
* Confirmation testing, to ensure that detected defects are resolved.
  1. **Test Data**

Test data will be provided through AWS EC2 hosted database containing queries and expected responses.

Data sources:

* FAQ table
* Response table

1. **Plan**
   1. **Test Team**

The test team for this project will comprise of Christopher Rayner as test lead, and any other of the development team who can contribute time towards testing.

* 1. **Major Tasks and Deliverables**

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **Start** | **Stop** | **Deliverables** |
| Test case design | 25/09/2024 | 25/09/2024 | Test cases |
| Build System environment | 25/09/2024 | 25/09/2024 | Test environment ready to be populated with build and test data |
| Test case data | 25/09/2024 | 27/09/2024 | Database for system and tests |
| Build System delivered for testing | 25/09/2024 | 27/09/2024 | First build of the system |
| Build 1 test execution | 29/09/2024 | 29/09/2024 | First build tested |
| Build 1 Test Summary Report | 29/09/2024 | 30/09/2024 | Summary report of tests performed on build 1. |

* 1. **Test Environment**

Test cases will be executed on the development server hosted on an AWS EC2 instance. With test data coming from a separate EC2 instance.

1. **Features to be Tested**

**4.1 Build 1**

**4.1.1 Chatbot functionality**

* Verify chatbot provides daily money-saving strategies
* Verify relevance of tips based on user interaction
* Validate information given for savings accounts, loans etc
* System can compare different financial products when prompted
* Verify chatbot suggests ways to track and balance income and expenses
* Ensure that chatbot provides general debt reduction tips
* Verify information on different types of loans
* Validate information provided on types of loans is accurate
* Ensure chatbot discusses pros and cons of loans clearly and neutrally
* Verify chatbot provides advice on cutting down on unnecessary costs
* Validate that common financial terminology is recognised

**4.1.2 Chatbot conversational flow**

* Verify chatbots understanding of various ways users might phrase their questions
* Test for clear, concise and helpful responses
* Ensure chatbot can handle incomplete or unclear queries by asking follow up questions
* Ensure chatbot has default response to direct towards frequently asked questions.

**4.1.3 Data accuracy**

* Check the correctness of information provided about financial products and services.
* Ensure chatbots advice is based on current financial best practices.

**4.1.4 User interface and Navigation**

* Verify that buttons, links and chatbot form is functional.
* Ensure users can start, pause and reset conversations easily.
* Ensure that the chatbot is easily accessible for all users.

**4.1.5 Security and data privacy**

* Validate that sensitive information is not stored unnecessarily
* Ensure compliance with data protection policies (GDPR)
* Ensure that information provided by the chatbot does not break any data protection policies.

1. **Features Not to be Tested**

**5.1 Authorisation & Authentication**

Authorisation and authentication is out of scope for this project and will be a potential future update to link with the web application hosting the chatbot.

1. **Testing Procedures**

**6.1 Test Execution**

For each requirement, business process, or system feature to be tested, the tester will execute a set of pre-defined test cases. Each test case will have a series of actions and expected results. As each action is performed, the results are evaluated. If the observed results are equal to the expected results, a checkmark is placed in the “pass” column. If the observed results are not equal to the expected results, a checkmark is placed in the “fail” column.

* 1. **Pass/Fail Criteria**

To pass the system integration test, the following criteria must be met:

* Chatbot functionality works as intended.
* Chatbot provides valid and accurate information.
* Chatbot answers the users request.
* Chatbot provides link to FAQ if unable to provide information on user request.
* Chatbot provides link to FAQ if user request falls under FAQ’s.
* Chatbot confirms user question if unsure of the users request.
* Chatbot complies with data protection policies (GDPR).
* The system is accessible to end users.

**6.3 Defect Management**

All defects found during testing are to be documented on Trello, with a unique identifier, title, brief description of the failure, steps to reproduce the failure, priority level, severity level, date of discovery, testing environment and the author of the defect report.

Severity levels will be as follows:

1 – Critical Business objectives are impacted

2 – High Defects which prove detrimental to the system

3 – Medium Defects which provide invalid information

4 - Low Defects that are aesthetic in nature ( e.g. chatbot window moves from one side of the screen to the other)

Priority levels will be as follows:

1 – High High likelihood of the failure occurring. (e.g. any user asking the chatbot how can I save money?)

2 – Medium Low likelihood of the failure occurring. (e.g. trying to ask the chatbot the same question twice in a row)

3 – Low Very low likelihood of the failure occurring (e.g. opening and closing the chatbot 10 times in 5 seconds)

**7. Risks and Contingencies**

This section describes the system or project risks and the contingency plans that should take effect if the project experiences problems.

|  |  |
| --- | --- |
| * Chatbot functionality - | Risk level moderate to high. Should a problem occur user may be provided with incorrect information which may influence their decisions into taking out a financial product. |
| * User accessibility - | Risk level low to moderate. Should a problem occur end user may be dissatisfied with their experience of the chatbot and seek alternate options. |
| * Data compliance - | Risk level moderate to high. Should a problem occur major fines may be issued for not complying with legal data protection laws. |
| * Test Team members joining and leaving - | Risk level low – Should a team member join they will need to be informed of the progress of the project and the current stage the project is at. Should a member leave assessing the amount of work remaining and sharing it among remaining members will be required. |