IEEE 829 Test Plan

Version 1.0

Done by:

Inspiration

Royce Ang Jia Jie (Leader)

Manav Arora (Vice-Leader)

Jovan Huang Tian Chun

Clarence Hong Shi Man

Zhu Weiji

Tan Hui Zhan

Revision History

Version #	Implemented By	Revision Date	Approved By	Approval Date	Reason
0.10	Jovan Huang Tian Chun	28/03/2022	Clarence Hong	28/03/2022	Initial Document Baseline
0.20	Jovan Huang, Clarence Hong, Weiji, Manav, Clarence, Hui Zhan	07/04/2022	Jovan Huang	07/04/2022	Filled in entire document
0.30	Tan Hui Zhan	08/04/2022	Royce	08/04/2022	Check and Review document
1.0	Manav	09/04/2022	Weiji	09/04/2022	Final Document check

Content Page

Test Plan Identifier

This test plan will be known as ‘Master Test Plan for Friendstagram_V1.0’.

Introduction

This is Friendstagram’s Master Test Plan. The purpose of this plan is to ensure that the final version of our Minimum Viable Product (MVP) for Friendstagram satisfies the initial requirement specification according to our System Requirement Specifications (SRS). We will be conducting thorough testing to ensure that our MVP meets all the functional and non-functional requirements of our end users. This will help us to verify that our MVP meets all the needs of the end-users.

We will be doing 3 types of testing, unit testing, integration testing and system testing. We will be elaborating more on the details of these tests in our Approach (Strategy) section.

When we discover erroneous outputs from our tests, we will fix them by editing our source codes. To make any change requests, we will follow the change procedure as documented in the change management plan document. Our change management plan document contains the process for change requests, communication between teammates and key activities coordination.

As for constraints, we have limited time as we have only three months to finish building the MVP and all documentations. If there is any delay in the testing, the developers will need some time to change the code when trying to fix bugs. This means there will be a delay in releasing the product to end-users, hence making time constraint a serious concern.

Test Items (Functions)

The Version 2.0 of Friendstagram will be tested.

In our testing, we will be using the functional requirements and workflow which have been documented as follows:

- D02_Use_Case_Model_V2.0, Version 2.0
    - Users can register for an account
        - Users need to be able to fill in their email, password and re-enter password in the input text boxes

        - Users need to be able to click on the ‘Sign up’ button to start creation of account
        - Users need to be able to pick hall location, personal interests and fill in their bio to finish account creation
    - Users can log into their accounts
        - Users need to be able to fill in their email and password in the input text boxes
        - Users need to be able to click the ‘Sign in’ button to authenticate account and log in successfully
    - Users can log out of their accounts
        - Users need to be able to click the ‘Logout’ button to successfully logout of their accounts
    - Users can find friends
        - Users need to be able to click the ‘Find Friends’ button to get matched with a list of new friends
        - Users need to be able to see the new friends with their list of interest in the cards
    - Users can view recommended friend profile
        - Users need to be able to click on ‘View Profile’ of the matched friend to view details, such as email, bio, hall and interests of the specific friend profile
    - Users can view personal profile
        - Users need to be able to click on ‘My Profile’ to view their personal details such as email, bio, hall and interest
        - Users need to be able to see their matched friend history in their personal profile
    - Users can edit personal profile
        - Users need to be able to click on the ‘Edit’ button to start editing their personal profile

        - Users need to be able to update their profile by selecting a new hall location and personal interests and writing a new bio.

- D03_SystemRequirementSpecification_V1.0, Version 1.0
    - Functional Requirements
        - The web application must allow the user to register for an account
        - The web application must allow the user to login to have access to the features
        - The web application must allow the user to logout from their account on the platform.
        - The web application must recommend the top k most similar profiles based on the user personal information like hall residences, interests, contact information, etc.
        - The web application must allow the user to view recommended profiles listing their interests, contact information, etc.
        - The web application must allow the user to view their own profile that reflects the user's interests, hall residences information, etc.
        - The web application must show a history of matched friends in their personal profile page.
        - The web application must allow the user to edit/modify/change their interests, etc in their profile.
    - Input Requirements
        - In order to register for an account, the user must input their email address and password
        - The user must input their 'About Me', interests and hall accommodation information before they can get recommendations on new potential friends.
    - Output Requirements
        - The system must display a list of recommended friends based on their

common interests and profile characteristics.

        - The system must display the profile of the user and history of his/her matched friends.
        - The system must display the profile of the potential friend chosen by the user which includes their interests, hall residence and contact information.

    - Process Requirements
        - The system must utilize SSL certificates to encrypt communications between the frontend and backend systems, i.e. data must not be modified by a malicious attacker performing a man-in-the-middle attack.
        - Data validation must be performed on both the client and server side to ensure invalid or malicious data input cannot be injected and/or forcibly stored into the Friendstagram system. Invalid data inputs must not be cached and buffers must be cleared properly. Proper error handling must also be performed for all functionalities in the Friendstagram system.
        - The Friendstagram system must be able to handle concurrent requests from multiple users without crashing. All errors must be handled gracefully without crashing the system and all errors must be displayed back to the user without disclosing the internal architecture of the Friendstagram system, e.g., backend server error codes and messages must not be displayed in its raw form back to users to prevent unintentional leakage of sensitive data of Friendstagram system. In the event of an unexpected crash, the system must be able to restart on its own and continue to provide services.

Software Risk Issues

Friendstagram requires third-party libraries and external tools in certain parts of the project. These libraries and tools are important for Friendstagram to function well and thus must be tested as well.

- Reliability of Netlify
    - Performance and reliability of the hosting service of Netlify such that it is able to handle multiple users at a time

- Reliability of Git
    - Reliability of Git to correctly record and backup the application throughout the testing lifecycle
- Reliability of PythonAnywhere
    - Performance and reliability of PythonAnywhere such that it is able to handle multiple requests at a time

Features To Be Tested

Features	Risk Level	Reason
● User can create an account ● User can choose their interests and halls ● User can find new friends through recommendations	H	This feature is critical to the functionality of the application and must be available upon the initial release of the application version 1.0.
● User can rate their interests ● User can login to an existing account ● User can log out of the current account	M	These features are not part of the main functionality of the application. They are functions that improve the user's experience. Hence, it is planned to be released in version 1.1.
● User can view/edit their personal profile ● User can view recommended friends' profile ● User can view recommendation history	M	Although this feature is not critical, it does improve the overall user experience of the application. As it is planned to be released in version 1.2, it is assigned to have medium risk.
● Bug fixes and improvements	L	This is not a critical feature and is planned to be in release version 2.0. Hence, it is assigned to have low risk.

Features Not To Be Tested

Features not to be tested	Reason

Browser Compatibility with Internet Explorer	Internet Explorer will no longer be supported from June 15, 2022. Therefore, when considering System testing for the program, Internet Explorer will not be included in the platforms we will be testing for.
Hardware interface	The hardware and their interfaces are tested by the hosting providers, and testing is focused on the software deployed to the providers. System testing will implicitly test these interfaces as well.
Database logic	The Django framework used for the backend provides a wrapper for the database logic, so testing is handled by Django maintainers. System testing will implicitly test these interfaces as well.

Approach (Strategy)

Testing Levels

A comprehensive, end-to-end testing strategy is important to assert the correctness of the program. The testing for Friendstagram will be done in multiple stages. We will begin with Test-Driven Development, where unit tests are written. Afterwards, integration tests will be mocked and conducted. Following that, system tests will be written and run.

Firstly, unit-tests will be written by the back-end developer to assert and validate the correctness of business rules and data such as user login information, the schema of user interests which were used for matching users to other users, etc. Regression tests will then be conducted to ensure that the back-end is working well, and will continue to work well even when changes have been made to the application. During development of the project, elicited requirements may change, causing the application flow to be different from what was originally planned and implemented. Comprehensive unit tests for back-end should have a minimum coverage of 75%. This allows quick changes to be made to the codebase whenever business logic is changed.

For the front-end, there is less focus on unit tests because of the structure of the frontend backend segregation and the use of pre-existing components. Because our frontend only provides an interface for users to interact with, but the true state of data is stored in the backend, there is limited use for unit tests on the front end. We might opt to test the functionality of individual buttons, but we use front-end frameworks such as Bootstrap and React which are well tested.

After all the critical defects from unit testing are resolved, integration tests may be included using the incremental integration - bottom up approach, slowly incorporating more of the interfaces between the frontend and the backend. This is crucial in making sure that the frontend and the backend work well together as a fully integrated application.

After which, system testing, functional testing and performance testing will be conducted. This helps the team to ensure that all technical, business and functional requirements have been met. The application will allow for up to one major defect as long as it does not prevent further testing and there are workarounds for the errors.

The Quality Assurance Manager will provide guidance on how APIs should be mocked and the test flow for end-to-end systems. For each CI/CD build, the data printouts, defect information and sample output from the tests will be provided to the Lead Developer for approval. This information is then passed on to the Quality Assurance Manager who will suggest and make changes to individual tests. If deemed necessary, the Quality Assurance Manager may include other tests such as stress testing or pilot testing.

Configuration Management/Change Control

Friendstagram shall be tested on a single release branch. Merging to the release branch shall be restricted to only the Team Lead (TL) and Project Manager (PM). To merge into the release branch, a Pull Request has to be made. A Pull Request tracks the changes made and before it can be merged, the TL/PM have to review the changes and ensure good code quality. Then, they will approve it and merge into the release branch.

Before a Pull Request can be approved and merged into the release branch, unit tests and integration tests have to be written and verified by both the development and quality assurance team. The unit tests and integration tests must be run and all the test cases must pass before the new changes can be merged into the release branch. This shall be accomplished by setting up a Continuous Integration/Continuous Delivery (CI/CD) pipeline to automatically run the unit and integration test cases.

All changes, enhancements and change requests to the Friendstagram system shall be handled according to the steps defined in the Change Management Plan document and recorded within the same document.

Test Tools

The following tools shall be used for testing:

1. Git and GitHub for version configuration management tool
2. GitHub Actions for setting up CI/CD pipeline.
3. React Testing Library and Jest for frontend testing
4. Pytest, Coverage.py for backend testing.

Meetings

Communication between project team members is critical for the success of the project. Due to conflicting timetables and hectic university schedules, the project team shall meet once every week to ensure sufficient communication and efficient collaboration between different sub-teams. During the meetings, each member shall answer the following 3 questions:

1. What have I accomplished since the last meeting?
2. What will I be doing before the next meeting?
3. What are the problems I am facing?

The above 3 questions help to ensure a sufficient flow of information between all members of the project team and encourage project members to take ownership of their tasks/deliverables, i.e. complete the tasks before the next week. The Project Leader/Managers can also intervene if a member has been stuck in a specific task for consecutive weeks. Furthermore, any problems a member is facing can be communicated to other members as soon as possible, who can pitch in to help.

Measures and Metrics

During the unit testing phase, the following metrics shall be collected by the quality assurance and the development team:

1. Defects, or unaccounted failures, and their corresponding severity
2. Cause of defects
3. Estimated time to resolve the defects

During the integration testing phase, the following metrics shall be collected by the quality assurance team and the development team:

1. Defects with integration, or unaccounted failures and their corresponding severity.
2. Cause of defects with integration
3. Estimated time to resolve the defects
4. New unit test cases to account for newly found defects and ensure they don't occur again.

Item Pass/Fail Criteria

The core functionalities of Friendstagram must work as defined in the test cases and all lower level plans should be complete. The quality assurance team should, to their best abilities, search and test for all possible modes of failure. If any modes of failure, they should then be communicated to the development team and a fix should be rolled out as soon as possible. For all releases of the Friendstagram system, there should not be any critical defects that could crash the system or render the system unavailable for users to access.

Furthermore, the test coverage should minimally cover 75% of the code to ensure sufficient coverage of all possible execution paths in the system so that only a few unexpected errors can arise. This ensures minimal disruption due to unexpected errors.

Suspension Criteria And Resumption Requirements

The following are the suspension criteria and resumption requirements for Friendstagram:

- The application cannot be loaded on a specific browser
    - Browser incompatibility issues should cause testing to be suspended
    - Changes made to fix compatibility issues may affect test cases detrimentally.
    - The resumption requirement for testing is such that the application can be loaded on all browsers.

- Netlify/PythonAnywhere is unavailable.
    - Testing should be halted immediately if either the frontend or backend servers become unresponsive.
    - The test results obtained from an unstable, unresponsive or unavailable server may not be accurate and reliable.
    - The resumption requirement for testing in such a scenario is when Netlify/PythonAnywhere becomes available again.

Test Deliverables

The following list is the list of deliverables for testing:

- Test Plan Document
- Test Cases
- Requirements Test Coverage Report
- Screen prototypes (Figma)

Remaining Testing Tasks

Since our project is a small one, we tested all the functionalities of our web application using unit testing, integration testing and system testing. In the future, we will conduct A/B testing as well so as to improve the user's experience.

Environmental Needs

The generation of test data will be done by the development team and further augmented by the quality assurance team. These data will have a different combination of potential user inputs.

After testing, the detected defects' severity and priority will be tracked using Git, a version configuration management tool issuer tracker.

To do testing on Friendstagram, we require the following all the tools under the "Test Tools" section.

Staffing and Training Needs

Preferably, one full-time tester should be assigned to the project to implement tests where required. Because of the lack of manpower, frontend and backend teams implement their own tests where required based on their own judgment and direction from the QA Manager and Lead Developer.

The integration of tests into the CI/CD pipeline will be managed by the QA Manager.

In order to ensure there is complete test coverage, the following points must be addressed:

1. Frontend and Backend developers must be kept updated on the requirements specification and notified of any changes to the specification

2. All developers must be familiar with the testing tools their teams are using

3. The QA Manager must keep the Lead Developer updated on the process of deploying the web application and the usage of the test suite during deployment

Responsibilities

Risks involved must be raised by all stakeholders of Friendstagram in order to ensure the timeliness and correctness of the product.

The Quality Assurance Manager will be responsible for designing and evaluating the overall testing of the product. They must coordinate the testing across the frontend and backend for integration and system testing. They must also convey the expectations on the level of detail of tests for individual components and the completion date. They must also communicate any risks to the team.

The Developers must undergo training on the test tools required for their specific roles, if they are not familiar with the tools used. The Frontend and Backend Developers are responsible for the final coverage and implementation of the frontend and backend unit test suite respectively.

The Lead Developer must coordinate with the Quality Assurance Manager to ensure the completeness of the test suite.

Ultimately, the Quality Assurance Manager has the power to make critical decisions on the execution of test cases.

Schedule

We outline the schedule, including the start and end dates for each testing milestone. In the event of slippage in our testing schedule, we will update the start and end dates accordingly to keep users and the project team informed. Testing shall be kept to the original schedule as far as possible, and it is expected that there will be no slippage up to the integration test level due to the familiarity of the developers with TDD and their testing tools.

If necessary, developers may be assigned to focus on testing in order to keep with the testing schedule. Frontend developers should preferentially be assigned to frontend testing and vice-versa to reduce the ramp-up and training time incurred.

The following table is the testing schedule for Friendstagram:

Release Version	Milestones	Testing Activities	Start Date	No.of Days	End Date
Version 1.0	Requirements Capture	Review	25/02/22	4	28/02/22
	Requirements Analysis	Review	25/02/22	4	28/02/22
	Systems Design	Review	29/01/22	5	02/02/22
	UI Design	Review	29/01/22	5	02/02/22
	Implementation	Peer Review Unit Testing	03/02/22	3	05/02/22
	Integration Testing	Review Integration Testing	06/02/22	3	11/02/22
Version 1.1	Requirements Analysis and Updating	Review	14/02/22	3	16/02/22
	Implementation	Peer Review	17/02/22	3	19/02/22

		Unit Testing			
	Integration Testing	Review Integration Testing	20/02/22	3	25/02/22
Version 1.2	Requirements Analysis and Updating	Review	28/02/22	3	02/03/22
	Implementation	Peer Review Unit Testing	03/03/22	3	05/03/22
	Integration Testing	Review Integration Testing	06/03/22	3	11/03/22
Version 2.0	Requirements Analysis and Updating	Review	14/03/22	3	16/03/22
	Implementation	Peer Review Unit Testing	17/03/22	3	19/03/22
	Integration Testing	Review Integration Testing	20/03/22	3	25/03/22

Risks and Contingencies

The following list defines the risks that the project team had identified and how to respond to them:

- Insufficient manpower during testing
    - The Project Manager will be informed about this shortage of manpower and he/she will allocate more members who are free to help out with the testing.
- Delayed delivery of software
    - The Release Manager will find out the root cause and discuss with the Project Manager on the possibility of shortening the project duration to meet the stipulated deadline first.
    - The Release Manager will also need to learn from this incident and avoid this in the future.

- Change in user requirements
    - Have frequent meetings with end-users to gain customer and stakeholder feedback as early as possible. This helps us to react as fast as possible so as to have more time to implement the changes and document everything well.

Requirements definition will be complete by 1 March, 2022, and, if the requirements change after that date, the following actions will be taken:

- The test schedule and development schedule will move out an appropriate number of days
    - This rarely occurs, as most projects tend to have fixed delivery dates
- The number of tests performed will be reduced
- The number of acceptable defects will be increased
    - These two items could lower the overall quality of the delivered product
- Resources will be added to the test team
- The test team will work overtime and this could subsequently affect team morale
- The scope of the plan may be changed
- There may be some optimization of resources
    - This should be avoided, if possible, for obvious reasons

Approvals

We will adopt a hierarchical approach for approvals so as to ensure that we have comprehensive and complete testing. Developers who write and run the unit tests are one of the first stakeholders who decide whether a test case passes or fails. The Lead Developer must approve the overall suite of unit tests for each component. System level tests must also be designed and evaluated by the Lead Developer. Afterwards, testing must be approved by both the Quality Assurance Manager and the Project Manager. This ensures complete coverage in both the technical and business aspects of the application.

Glossary

QA: Quality Assurance

References

Below is the list of documents that support this test plan:

1. D03_SystemRequirementSpecification_V1.0
2. D05_ProjectPlan_V1.0
3. D06_RiskManagementPlan_V1.0
4. D09_ChangeManagementPlan_V1.0