# Project name:

Improvements to user address system.

# Overview:

The overview of this project is to create a testing process for the user to change/update their address. Currently, there is only a UK form available to the user, so if they move outside the UK, they are unable to change their details. The way the details are stored and updated could also be improved by the team to a more efficient system, with less complicated processes and languages.

# Meetings:

There will be a team meeting twice a week to check the progress of each of the team members and to discuss progress or any concerns they are having with their roles. Make sure they are planned in advance, or the stakeholders will not be available due to busy schedules.

# Risks:

* If the requirements are not well-defined, complete, and accurate, it can lead to incorrect test cases and results. This can cause a delay in the project and result in additional costs.
* If the testing coverage is not very good, some defects may go undetected, which can cause problems for end-users. This can result in customer dissatisfaction and harm the company's reputation.
* If the testing team lacks the necessary resources, such as hardware, software, or lack of team members, it can lead to a delay in the testing process and cause the project to exceed its budget.
* Miscommunication between the development team and the testing team can result in misunderstandings and a lack of coordination, leading to missed deadlines, wasted resources, and an incomplete testing process.
* Over-reliance on automated testing tools without human intervention can lead to missed defects and false positives. Additionally, automation tools can require significant upfront investment and ongoing maintenance costs.
* Technical difficulties can happen during the testing process, such as software crashes or hardware failures, which can lead to a delay in the project and additional costs.
* Improper data security and privacy, measures can lead to data breaches, which can harm the company's reputation, result in fines, and cause a loss of customer trust.

# Resources and People

* Test Manager- The test manager oversees the entire testing process and is responsible for managing the testing team, creating testing plans and schedules, communicating with stakeholders, and reporting testing progress.
* Test Leads- Test leads are responsible for managing a team of testers, creating test cases and plans, identifying testing requirements, and communicating with the company.
* Testers- Testers are responsible for running test cases, identifying problems, and reporting issues to the test lead.
* Automation Engineers- Automation engineers are responsible for developing and maintaining automated test scripts and frameworks.
* Technical Writers- Technical writers are responsible for creating test documentation, including test plans, test cases, and test reports.
* Quality Assurance Analysts- These analysts are responsible for ensuring that the software meets the required quality standards and specifications.
* Developers- Developers work closely with testers to resolve defects and ensure that the software meets the desired functionality and requirements.
* Hardware and Network Resources- Hardware and network resources are necessary for testing software in various environments.
* Testing Tools- Testing tools, such as testing frameworks, test management software, and automation tools, are necessary for executing and managing the testing process.
* Training and Development Resources- Training and development resources are necessary for keeping the testing team up to date with the latest testing techniques and tools.

# Test environment and tooling:

* Use test management tools to manage the testing process, including test planning, execution, and reporting. Making use of JIRA to track issues such as bugs will help when trying to solve problems.
* Choose automation testing tools based on the application under test and the required level of test automation.
* Using load and performance testing tools to identify problems and performance issues in the application under test.
* Using defect tracking tools to manage and track issues found during testing.
* Using code analysis tools to identify potential issues in the application code, such as security vulnerabilities, and to ensure they comply with coding standards.
* Cloud-based testing environments allow you to test your software applications in a virtual environment without the need for physical hardware or infrastructure. Cloud testing services like AWS Device Farm, BrowserStack, and Sauce Labs provide a wide range of testing environments, devices, and browsers that can be accessed remotely from anywhere.
* In-house testing environments involve setting up a dedicated physical space within your organisation for testing software applications. This can be a cost-effective option if you have the necessary resources and infrastructure to set up and maintain the testing environment.
* Virtualised testing environments allow you to create multiple virtual machines on a single physical machine, which can be used for testing different software applications simultaneously. This can be a good option if you have limited hardware resources or need to test applications in different environments.
* AppDynamics is a commercial testing tool used for performance monitoring and application analytics. It allows you to monitor application performance in real-time and identify performance issues before they impact users this would be useful for the business.
* Ranorex Studio is a commercial testing tool used for web, mobile, and desktop application testing. It supports multiple scripting languages and offers a range of features for test automation, reporting, and management.
* Telerik Test Studio is a commercial testing tool used for web and desktop application testing. It supports multiple browsers and frameworks and allows you to create automated tests using a codeless test automation framework.

# In scope:

* Identifying the testable requirements of the new update under test.
* Identifying the key user scenarios to test the new update under different use cases.
* Identifying the specific test scenarios for each requirement or user scenario.
* Design test cases for each identified test scenario.
* Create or identify appropriate test data to support each test case.
* Execute the identified test cases and report problems/issues.
* Plan and execute repeat testing for each new release or version of the software.
* Testing the performance of the software under different circumstances.
* Testing the security of the new update by identifying vulnerabilities and potential attacks from hackers.
* Testing the usability of the new update by identifying user experience issues.
* Testing the software compatibility with different operating systems, devices, and browsers.
* Testing the accessibility of the new update for users with disabilities.
* Test the new update for different languages, cultures, and in different locations by using a VPN.

# Out of scope:

* If the project uses third-party software that has already been tested and approved, then testing these components again may be considered out of scope.
* User acceptance testing is typically conducted by the end-users or owners themselves to ensure that the software meets their business needs. If User acceptance testing is not part of the testing project, then it should be considered out of scope.
* If testing in the production environment is not part of the project scope, then it should be considered out of scope.

# Other considerations:

* Look to change microservices written in SCALA to something more popular and convenient. Here are some examples that could make life easier for the team. Python is known for its ease of use and its high functionality; it has many similarities to SCALA. Kotlin is a modern, statically typed language that runs on the Java Virtual Machine and is often used as an alternative to Java. It offers many of the same features as Scala, such as functional programming, immutability, and null safety, but with a simpler syntax that is easier to learn and use.
* Look into setting a company standard for integration tools so everyone knows what they are doing and using the same software throughout for integration.