Automation Test plan

# Test purpose and objective

To validate that manually tested functionality does not brake with new code commits and version deployments. To verify that target System can fulfill its goal and be easily maintained.

# Automation coverage

Project’s areas of automation are WEB and API functionalities, delivered in Sprint, which have passed manual acceptance testing and are fully described in Test cases. This also includes E2E scenarios or specific Business scenarios, constructed from multiple delivered features integration.

## Including

Any Web Elements, being part of the delivered functionality, will be verified from the perspective of their placement and available logical activities (e.g. hover, click, type etc.)

Any API endpoints, which are opened for external calls.

## Excluding

Any nonfunctional specification, such as performance or security

Any API endpoints, which are opened only for internal calls.

# Technology stack

This project will be built on Java Programming Language (version 1.8.0\_xxx), with Selenide as Selenium Wrapper, Gradle as the build tool, Cucumber for Behavior driven development, Rest Assured library for API coverage, Lombok extension for POJO creation, Allure for test reporting. Code will be shared through code repository Github.

# Testing approach

Once a Story or Scenario has passed manual testing, it can be put for automation. Automation of the functionality starts with automation engineer going through the Test cases, prepared by the Person testing the functionality manually. After resolving any questions which may create uncertainty in the automation process, Engineer gives his estimation of the time it would take to automate the functionality. Once the time is agreed with the Test lead, Engineer can start writing test scripts for the tested functionality. After all the code is prepared, it gets pushed into the Repository and is left ‘In Review’ state, until at least 2 other automation engineers review the code. As soon as review is passed, test must be merged into the Master branch, for future execution as part of Regression, E2E or Sanity suites.

# Risks

Major risk in the execution of the automated scenarios is regression. If the build version of the site does not get updated frequently OR tests are not being executed daily, there is a high chance of failures to appear inside the run, meaning that the automated suite is no longer trustworthy.

Low quality of Test cases may result in a bad quality of Automated test scripts.

Lack of Unit and Integration tests may result in frequent functionality braking, requiring constant support both from the Developers and Testing Engineers.

# Data requirement

Any data required for a specific test Scenario must be provided beforehand.

# Team members

John Doe – Test lead, responsible for approving or suggesting Automation tasks.

Jane Doe – Automation Engineer, responsible for writing WEB GUI testing scripts.

Brad Pitt – Automation Engineer, responsible for writing API testing scripts.

# Test environments

One environment, using the same build version as Test, but dedicated only to automation, with obfuscated Data, to be configured before any automation scripting work should be started.

…………………….