Table of Contents

1. Scope

- Purpose of this document is to create a test cycle from initiation till closure.
- The scope of testing is limited to the EPIC requirements provided on JIRA. Review of the document will be done by Senior Stakeholders. The document contains all the testing timelines.
- This is a formal Test approach document, for details around team combination, defect reporting and other management details please refer Test Plan.

2. Test Approach

- **Testing Levels**:
  - Unit Testing will be supported by development team with acceptance criteria provided from Quality Team and Product Owners.
  - Integration Testing will be carried out by Quality team with Automation up to 80% for regression and 50% for functional changes.
  - UAT and Pre-Prod will be carried out by App Support team, QA team to support Devops/AppSupport team for environment/configuration related details.

- Roles and Responsibilities Details: To be added

- **Environment Details**:
  - QA to create environment for Automation/Manual/All types of testing.
  - QA to create Stubs/Mocks/Test Data for password change requirement.

- **Types of Testing:**
  - **Backend/API testing** to be carried out with RestAssured for Automation flow and Jmeter/Postman for functional manual flows.
  - QA to create **stubs** for any third party integration of API's.
  - **Database testing** to be carried out up to an extent where data consistency is validated across all the micro services.
  - **For UI testing**, Use Selenium/Appium depending on the technology stack.
  - **Performance Testing** to be carried on different environment (Prod Replica) than QA, Jmeter to be used for performance testing with only 10% of total users would be changing password at a time. **Tps** to come from product team.
  - **Security Testing** to be carried out by DevSec team supported by QA team
- Defect Management, triage and Reporting: To be added.

- **Functional(Technical) Responsibilities for QA:**
  - Validate the **changePassword** API against the contract/spec provided by Dev Team.
  - Create **RestAssured** Regression Pack.
  - Validate responseCodes, responseBody, headers and other related semantics.
  - Validate API with optional parameters.
  - Validate Authorization testing scenarios.
  - Integration testing multiple micro services and UI.
  - Validate endpoint with multiple configuration like queryParams, pathParams.
  - Send Mal-Formed content to validate robustness handling of the API.

High Level Test Design(Sample):

#	Test Scenario Category	Test Action Category	Test Action Description
1.	Positive Workflow – Happy Path		
	Execute API call with **valid required parameters**	Status Code Validation	1. Should return 2XX status codes. 2. ChangePassword should be a POST request with 201 status to update the password and 4xx to deny the password change.
		Payload Validation	1- JSON Validation 2- Structure of the response Body etc.
		Headers Validation	1- Content Type 2- Cache etc.
		Performance Validation	TPS as per Test Plan/Epic.
		Security Validation/OAuths	As per Plan.