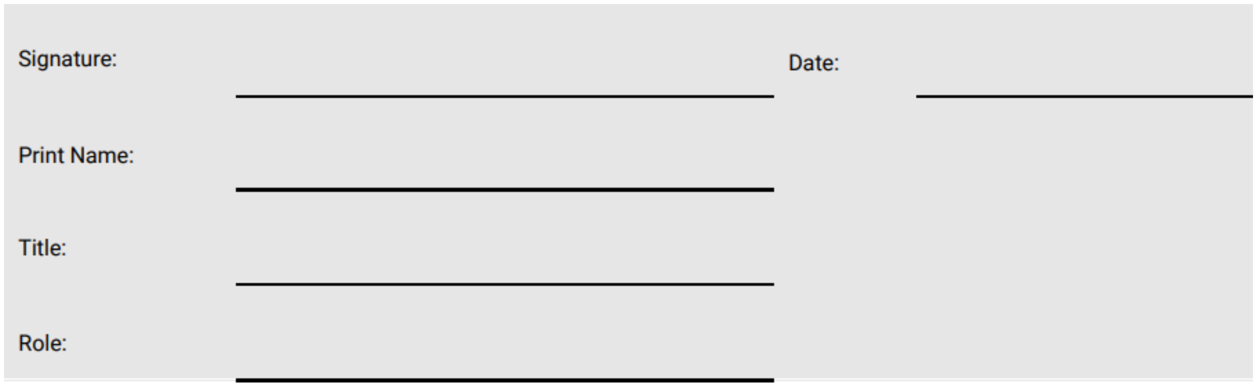
Revision History

Date	Version	Details	Author
04.03.2021	1.0	Unit Testing User Acceptance Testing Performance Testing Load Testing	Md Shahjahan Miah(Hasan)

Approvals

The undersigned acknowledge that they have reviewed the Master Test Plan and agree with the information presented within this document. Changes to this plan will be coordinated with, and approved by, the undersigned, or their designated representatives.



Signature:		Date:	
Print Name:			
Title:			
Role:			

Artifact: Master Test Plan

Eatilink URL Shortener master test plan is a document that describes the objectives, scope, approach, and focus of a software testing effort. The process of preparing a test plan is a useful way to think through the efforts needed to validate the acceptability of a software product. The completed document will help people outside the test group understand the 'why' and 'how' of product validation. It should be thorough enough to be useful but not so overly detailed that no one outside the test group will read it.

Domains: Testing Domains

Roles	**Responsible** • SIT(QA) Tester • UAT Tester	**Modified By:** • SIT (QA) Tester • UAT Tester • Test Lead
Tasks	**Input To** • Unit Test Cases • User Acceptance Test	**Output From** • Determine Resource needs • Develop Project test Plan • Plan Test Environment and Determine Data Needs

Environments: Testing Environments

Tools

Installation and deployment file is explained it’s tools, installation and deployment documentation.

Setup

Installation and deployment file is explained it’s setup, installation and deployment documentation.

Integrations and Intersystem Interfaces

The following tabular contents will list down the various Interfaces/Applications involved in the Integration Testing of Link Shortener Project and also contains the individual point of contact that will be used for coordinating any Integration Testing.

System ID	Application/Functional Area	Testing Responsibility
Application Gateway	N/A	N/A
Identity Server	N/A	N/A
Link Shortener Service	Link Shortener	Eatigo Recruitment Team

Test Types

Unit Testing

Purpose	This preliminary test is performed by the development team for testing of individual configuration, custom programs and/or technical services (e.g. Link Shortener Service) to ensure that they function according to the detailed technical specification.Unit test is a white box test and should test all possible flows. Both positive and negative conditions should be tested.
Development Phase	Development and Testing
Test Scope	All configurations, code validation, memory testing, integration, code complexity, etc.
Test Environment	Development Environment
Test Data	Manual data created by developers
Interface Requirements	N/A
Role	Developer
Entry Criteria	• Formal reviews for process models, functional spes and technical specifications have been completed • All Inspection related defects have been corrected • All documentation and design of the architecture must be made available • Development of the component is complete and compiles without error • All Unit test cases are documented
Exit Criteria	• All Unit test cases completed successfully • All source code is unit tested • No outstanding critical defects • All outstanding defects are entered into the defect tracker • All test results have been documented

Link Shortener Service

API 1 Shorten

Serial No	Test Procedures	JSON Input	JSON Output	Type	Status
API 1.1	Verify if Token Authentication is invalid including Bearer Token, Claim is JWT, Header Algorithm is HMAC SHA256 and JWS Payload includes Token expiration date and user identity.	`{ "original_url":"https://eatigo.com/th/bangkok/en" }`	`{ "error_code":"INVALID_ACCESS_TOKEN", "data":{ "field":"Token", "message":"Specify Valid Access Token" } }`	Negative	PASS
API 1.2	Verify if Token Authentication is valid including Bearer Token, Claim is JWT, Header Algorithm is HMAC SHA256 and JWS Payload includes Token expiration date and user identity.	`{ "original_url":"https://eatigo.com/th/bangkok/en" }`	`{ "original_url":"https://eatigo.com/th/bangkok/en", "short_url":"https://eati.go/jU" }`	Positive	PASS
API 1.3	Verify if Token Authentication is invalid, it will return 401 Unauthorized Http Response	`{ "original_url":"https://eatigo.com/th/bangkok/en" }`	`401 Unauthorized`	Negative	PASS
API 1.4	Verify if the request body original url is empty, it will return specify valid url with error code EMPTY_ORIGINAL_URL	`{ "original_url":"" }`	`{ "error_code":"EMPTY_ORIGINAL_URL", "data":{ "field":"original_url", "message":"Specify Valid Url" } }`	Negative	PASS
API 1.5	Verify if the request body original url is invalid, it will return unable to shorten that link, it is not a valid url with error code INVALID_URL	`{ "original_url":"hasan" }`	`{ "error_code":"INVALID_URL", "data":{ "field":"original_url", "message":"Unable  to shorten that link. It is not a valid url." } }`	Negative	PASS

API 1.6	Verify if response body short url will include eati.go domain name	`{ "original_url":"https://eatigo.com/th/bangkok/en" }`	`{ "original_url":"https://eatigo.com/th/bangkok/en", "short_url":"https://eati.go/jU" }`	Positive	PASS
API 1.6	Verify if response body short url short string base62	`{ "original_url":"https://eatigo.com/th/bangkok/en" }`	`{ "original_url":"https://eatigo.com/th/bangkok/en", "short_url":"https://eati.go/jU" }`	Positive	PASS
API 1.7	Verify if all validation meets the validation rules then base62 short url will generate a unique id and save to the database.	`{ "original_url":"https://eatigo.com/th/bangkok/en" }`	`{ "original_url":"https://eatigo.com/th/bangkok/en", "short_url":"https://eati.go/jU" }`	Positive	PASS
API 1.8	Verify if response return from memory cache when it is accessed before it's time period limit	`{ "original_url":"https://eatigo.com/th/bangkok/en" }`	`{ "original_url":"https://eatigo.com/th/bangkok/en", "short_url":"https://eati.go/jU" }`	Positive	PASS
API 1.9	Verify if memory cache is auto refresh after it's time period limit.	`{ "original_url":"https://eatigo.com/th/bangkok/en" }`	`{ "original_url":"https://eatigo.com/th/bangkok/en", "short_url":"https://eati.go/jU" }`	Positive	PASS

User Acceptance Test (UAT)

Purpose	User acceptance test is performed by business users (Here applicable Eatigo Recruitment Team). The users test the complete, end-to-end business processes to verify that the implemented solution performs the intended functions and satisfies the business requirements.
Development Phase	Final Prep or Implementation
Test Scope	• UAT • Full Regression
Test Environment	Pre-Prod or Implementation
Test Data	Mock cutover or Test Data Management tool
Interface Requirements	interface connectivity required for all interfacing systems
Role	Process Team & Business Users (Here Eatigo Recruitment Team)
Entry Criteria	• The application works functionally as defined in the specifications • No outstanding “showstopper or severe” defects • All areas have had testing started on them unless pre agreed by UAT stakeholder/Test and Project managers • Entire system functioning and all new components available unless previously agreed between UAT stakeholder/Test manager and project managers • All test cases are documented and reviewed prior to the commencement of UAT
Exit Criteria	• The Acceptance Tests must be completed, with a pass rate of not less than 98%. • No outstanding “showstopper or severe” defects • Less than 5 significant defects outstanding • All Test cases have been complete • No new defects have been discovered for a week prior to Production Implementation. • All test results recorded and approved • UAT test summary report documented and approved • UAT close off meeting held.

Link Shortener Service

#	Name	Prerequisites	Description	Test Procedure	Test Result
API 1	Shorten	Invalid Authentication	Invalid Token authorization.	API 1.1	It will not generate shortened links.
		Authentication	JWT Token Authentication	API 1.2	It will generate shorten link
		Unauthorized	Invalid Token Authentication	API 1.3	It will not generate shortened links.
		Authenticated but empty original url	If the shorten api request body contains empty original url	API 1.4	It will give the response empty original ur.
		Authenticated but invalid original url	If the shorten api request body contains invalid original url	API 1.5	It will give the response invalid original url.
		Authenticated and validi original url	If the shorten api request body contains valid original url	API 1.6	It will give the response with a short url with base62 short string.
		Authenticated and valid original url	If the shorten api request body contains valid original url	API 1.7	It will give the response with a short url with base62 short string and also generate a unique id to save the database as UID.
		Return from memory cache	If any url accessed before and want to shorten again it's time limit, then it will return from the memory cache.	API 1.8	Return from memory cache.
		Auto Refreshing cache by it’s time limit	If any url accessed before and want to shorten again it's time limit, then it will return from the memory cache.	API 1.9	Return from memory cache.

Performance Testing

Intentionally Left Blank.

Load Testing

Intentionally Left Blank.

Test Deliverables

Status and Issue Reporting

Types	Gateway	Identity Server	Link Shortener Service	Remarks	Issues
Unit Testing	N/A	N/A	100%	Tested by Hasan	N/A
User Acceptance Testing	N/A	N/A	Intentionally Left Blank	Intentionally Left Blank	Intentionally Left Blank
Performance Testing	N/A	N/A	Intentionally Left Blank	Intentionally Left Blank	Intentionally Left Blank
Load Testing	N/A	N/A	Intentionally Left Blank	Intentionally Left Blank	Intentionally Left Blank