Test Plan (Inventory API)

6/29/2021

Agoda

Ankit Yadav

**Version:** 1.0

**Created:** 29/06/2021

**Last Updated:** 30/06/2021

**Status:** DRAFT

**Revision and Signoff Sheet**

**Document History** -

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Description of Change |
| 1 | 30/06/2021 | Ankit Yadav | Draft |
|  |  |  |  |

# INTRODUCTION

Application Inventory – API, its job is to serve API endpoints for fetching/upserting the

"hotel inventory data," which include rates and remaining rooms for each date. This document has clearly identified what the test deliverables will be, and what is deemed in and out of scope.

## In scope

* GET {hostName}/fetch/inventory
* POST {hostName}/update/inventory
* Performance Tests

## Out of scope

* Test Environment
* Hardware Interfaces

## Quality Objectives

* Meets full requirements including quality requirements (functional and non-functional). At the end of the project development cycle the user should find that the project has met or exceeded all their expectations.
* Ensure the Application Under Test conforms to functional and non-functional requirements
* Ensure the AUT meets the quality specifications defined by the client
* Bugs/issues are identified and fixed before go live

## Roles and Responsibilities

|  |  |
| --- | --- |
| Role | Responsibilities |
| Project Manager | 1. Acts as a primary contact for development and QA team.  2. Responsible for Project schedule and the overall success of the project. |
| QA Lead | 1. Participation in the project plan creation/update process.  2.Planning and organization of test process for the release.  3.Coordinate with QA analysts/engineers on any issues/problems encountered during testing.  4.Report progress on work assignments to the PM |
| QA | 1. Understand requirements  2. Writing and executing Test cases  3. Preparing RTM  4. Reviewing Test cases, RTM  5. Defect reporting and tracking  6. Retesting and regression testing  7. Bug Review meeting  8. Preparation of Test Data  9. Coordinate with QA Lead for any issues or problems encountered during test preparation/execution/defect handling. |

# TEST STRATEGY

## Test Principles

* Testing will be focused on meeting the business objectives, cost efficiency, and quality.
* There will be common, consistent procedures for all teams supporting testing activities.
* Testing processes will be well defined, yet flexible, with the ability to change as needed.
* Testing activities will build upon previous stages to avoid redundancy or duplication of effort.
* Testing environment and data will emulate a production environment as much as possible.
* Testing will be a repeatable, quantifiable, and measurable activity.
* Testing will be divided into distinct phases, each with clearly defined objectives and goals.
* There will be entrance and exit criteria.

## Test Scenario

**Functional Test**

**Endpoint 1: Fetch Inventory**

1. Should be able to fetch valid response from the inventory API without query parameters and success response and code should be returned.
2. Should be able to fetch valid response from the inventory API with query parameters and success response and code should be returned, when using:
   1. hotel\_id (integer)
   2. room\_id (integer)
   3. start\_date (string)
   4. end\_date (string)
3. Should be able to fetch valid response from the inventory API combining different query parameters and success response and code should be returned.
4. Should be able to fetch valid response from the inventory API combining different query parameters with OR operations and success response and code should be returned.
5. Should be able to fetch valid response from the inventory API combining different query parameters with AND operations and success response and code should be returned.
6. Should not be able to get valid response from API and bad response and code should be returned. hen provided:
   1. Incorrect hotel\_id (not present in DB)
   2. Invalid hotel\_id (string)
   3. Invalid hotel\_id (special characters)
   4. Incorrect room\_id (not present in DB)
   5. Invalid room\_id (string)
   6. Invalid room\_id (special characters)
   7. Invalid start\_date (integer)
   8. Invalid start\_date (special characters)
   9. Invalid end\_date (integer)
   10. Invalid end\_date (special characters)
7. Should not be able to fetch the inventory API combining different invalid query parameters and bad response and code should be returned.
8. Should not be able to fetch the inventory API combining different combination of valid and invalid query parameters with OR operations and bad response and code should be returned.
9. Should not be able to fetch the inventory API combining different combination of valid and invalid query parameters with AND operations and bad response and code should be returned.

**Endpoint 2: Update Inventory**

1. Should be able to insert new record to the Update Inventory endpoint for new room of hotel.
2. Should be able to update new record to the Update Inventory endpoint for existing room of hotel.
3. Should be able to update new record to the Update Inventory endpoint for existing room of hotel by providing:
   1. remainingRoom
   2. rate
   3. remainingRoom and rate
4. Should not be able to insert new record to the Update Inventory endpoint for new room of hotel when hotel\_id is invalid.
5. Should not be able to insert new record to the Update Inventory endpoint for new room of hotel when room\_id is invalid
6. Should not be able to update the record to the Update Inventory endpoint for new room of hotel when:
   1. Incorrect hotel\_id (not present in DB)
   2. Invalid hotel\_id (string)
   3. Invalid hotel\_id (special characters)
   4. Incorrect room\_id (not present in DB)
   5. Invalid room\_id (string)
   6. Invalid room\_id (special characters)
   7. Invalid start\_date (integer)
   8. Invalid start\_date (special characters)
   9. Invalid end\_date (integer)
   10. Invalid end\_date (special characters)

**Security**

1. When person without access trying to fetch inventory API.
2. When person without access trying to insert/update inventory API.
3. When a user with admin access to the inventory, it should be allowed modification and fetching.
4. When a user with non-admin access to the inventory, it should not be allowed to do modification and only be able to fetch.
5. Hotelier should be allowed to perform fetch, insert, and update option to the inventory API.
6. FE Clients should be allowed to perform fetch, insert, and update option to the inventory API.
7. Masking of data based on user type/person.
8. Proper authorization and authentication schemes should be validated against different user types and headers
9. If API is not exposed to outside system, it should not be accessible from outside should only be privately accessible.
10. If its Public API it should be accessible authorization mechanism like JWT, OAuth
11. To do validation of encryption methodologies, and of the design of the API access control.
12. To check user rights management and authorization validation.

**Performance**

1. Number of users using a particular API
2. Concurrent users trying to access APIs
3. If query results in huge data volume
4. To check the response time with different number of items –
   1. multiple hotels
   2. multiple rooms
   3. large date range

**Integration Tests**

1. When FE Clients tries to access/fetch data via the inventory it should be reflected properly on the UI.
2. When FE Clients tries to insert/update data via the inventory it should be get reflected on DB side and should be reflected as per updated values on fetching again.
3. Queries made from the UI of FE Client should reflect appropriately by making calls from the inventory api by making use of different params supported by the endpoint.

**Regression** -   
 1. TC related to the affected areas

## Automation Execution at different levels

**Smoke** (Level 0, Level 1) – Should be executed before PR

**Sanity** - Should be executed after every merge

**Integration** Test– should be executed once code is integrated from master branch

**Automated** functional Test – On scheduled basis, On different environment from master branch

Complete Automated **Regression** Test - before building release package from master branch

## Data Approach

* In functional testing, Inventory API will contain pre-loaded test data, and which is used for testing activities.

## Scope and Levels of Testing

### Exploratory

**PURPOSE**: the purpose of this test is to make sure critical defects are removed before the next levels of testing can start.

**SCOPE**: Fetching, inserting, and updating data into the inventory tables

**TESTERS**: Testing team.

**METHOD**: this exploratory testing is carried out in the application without any test scripts and documentation

**TIMING**: at the beginning of each cycle.

### Functional Test

**PURPOSE:**  Functional testing will be performed to check the functions of inventory application. The functional testing is carried out by feeding the input and validates the output from the application.

**Scope:** Fetching, inserting, and updating data into the inventory tables.

**TESTERS**: Testing Team.

**METHOD**: The test will be performed according to Functional scripts, which are stored in any Test Management tool.

**TIMING**: after Exploratory test is completed.

### System Test

**PURPOSE**: this test focuses on validating the end to end test scenarios of inventory api.

**TESTERS**: Testing Team.

**METHOD**: The test will be performed according to end to end functional scripts, which are stored in any Test Management tool.

**TIMING**: After Functional testing is done.

### Smoke Test

**PURPOSE**: this test focuses on ensuring the stability the build.

**TESTERS**: Testing Team.

**METHOD**: The test will be performed according to smoke test, which are stored in any Test Management tool and using the automation scripts.

**TIMING**: This is the first testing on the initial build and is performed on every build.

### Sanity Test

**PURPOSE**: Focuses on verifying the newly added functionalities, bugs etc. are working fine.

**TESTERS**: Testing Team.

**METHOD**: The test will be performed according to sanity scripts, which are stored in any Test Management tool and using the automation scripts.

**TIMING**: This is performed when the build is relatively stable and post regression.

### API Testing

**PURPOSE**: Focuses on verifying the functionality, reliability, performance, and security of the programming interfaces.

**TESTERS**: Testing Team.

**METHOD**: The test will be performed according to sanity scripts, which are stored in any Test Management tool and using the automation scripts.

**TIMING**: This is performed when the build is relatively stable and post regression.

### Performance Testing

**PURPOSE**: To determine the system parameters in terms of responsiveness and stability under various workload.

**TESTERS**: Testing Team.

**METHOD**: The test will be performed according to sanity scripts, which are stored in any Test Management tool and using the automation scripts.

**TIMING**: This is performed when the build is relatively stable and post regression.

## Test Deliverables

Here mention all the Test Artifacts that will be delivered during different phases of the testing lifecycle.

* Test Plan
* Test Cases
* Requirement Traceability Matrix
* Bug Reports
* Test Strategy
* Test Metrics

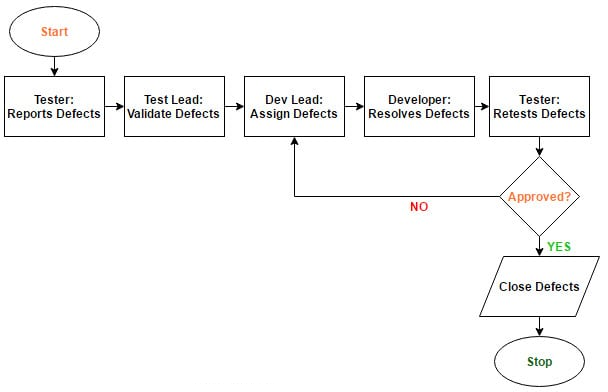
# Defect Management & Reporting

## Defect Management

|  |  |
| --- | --- |
| **Severity** | **Impact** |
| 1 (Critical) | * This bug is critical enough to crash the system, cause file corruption, or cause potential data loss * It causes an abnormal return to the operating system (crash or a system failure message appears). * It causes the application to hang and requires re-booting the system. |
| 2 (High) | * It causes a lack of vital program functionality with workaround. |
| 3 (Medium) | * This Bug will degrade the quality of the System. However there is an intelligent workaround for achieving the desired functionality - for example through another screen. * This bug prevents other areas of the product from being tested. However other areas can be independently tested. |
| 4 (Low) | * There is an insufficient or unclear error message, which has minimum impact on product use. |
| 5(Cosmetic) | * There is an insufficient or unclear error message that has no impact on product use. |

## Defect tracking & Reporting

Following flowchart depicts Defect Tracking Process:



# TEST ENVIRONMENT

The following list of software and hardware should be considered a minimum requirement:

* System – Windows 8 and above /Unix Environment
* Applications - Postman
* Test data
* Database server
* Browser – Chrome, Firefox
* Network
* Documentation required like reference documents/configuration guides/installation guides/ user manuals

**Setup of Test Server**

Every test may not be executed on a local machine. It may need establishing a test server, which can support applications.

**Network**

Network set up as per the test requirement. It includes,

* Internet setup
* LAN Wifi setup
* Private network setup

It ensures that the congestion that occurs during testing doesn't affect other members. (Developers, designers, content writers, etc.)

**Test PC setup**

API testing may require

* OS
* API Testing tool like Postman