Test Plan (Katalon Cura)

# **Test Plan – Katalon Cura Make Appointment API**

## **Objective**

The goal of this test plan is to ensure the quality, functionality, and reliability of the Katalon Cura Make Appointment API, which manages authentication and appointment requests for a healthcare booking system.

## **Scope**

## **Scope of Test Plan for Katalon Cura Make Appointment API:**

**1. Functional Testing:**

* Verify the correctness and functionality of Login and Appointment APIs as per documentation.
* Test scenarios for both successful and failed login, and all appointment booking operations.
* Validate authentication and authorization mechanisms (if any for protected endpoints).

**2. Data Validation Testing:**

* Ensure correct validation and rejection of bad data by the APIs.
* Test boundary values for inputs (date fields, string lengths, etc.).
* Validate accuracy of data returned in API responses.

**3. Error Handling Testing:**

* Verify appropriate error codes/messages for invalid requests.
* Check for leakage of sensitive information in errors.
* Ensure graceful handling of unexpected failures.

**4. (Optional) Performance Testing:**

* May assess API’s response times and stability under moderate loads, as scheduled.

## **Inclusions**

* Testing of Login (POST) operations to authenticate users.
* Testing of Make Appointment (POST) operations to create bookings.
* Verification of required data fields and valid/invalid input combinations.
* Validation of Content-Type and headers.
* Edge/boundary value testing for date, comment, and other relevant fields.
* Negative testing: missing parameters, incorrect formats, and unauthorized access.

## **Exclusions**

* UI-level, frontend, and non-API functional testing.
* Security, load, and scalability testing beyond basic checks.
* Payment, notification, or non-listed features.
* Extensive third-party or integration testing (unless specifically required).

## **Test Environments**

| **Name** | **Env URL** |
| --- | --- |
| QA | <https://katalon-demo-cura.herokuapp.com/> |

Supported Platforms for API access:

* Windows 10, 11 – Chrome, Firefox, Edge
* Mac OS – Safari, Chrome
* Linux – CLI tools (curl, httpie)
* (APIs tested via Postman, curl, similar platforms)
* Network: Wired/Wi-Fi, moderate bandwidth

## **Defect Reporting Procedure**

* Use JIRA (or equivalent) for tracking.
* Defects logged for deviations from requirements, tech errors, or bad UX.
* Attach API requests, responses, and steps to reproduce.
* Assign severity/priority, triage daily.
* Roles: Testers (logging), Developers (fixing), Test Lead (overseeing).
* Update stakeholders daily on defect status and progress.

## **Test Strategy**

**Step 1: Test Scenarios and Test Cases**

* Create detailed test cases using:  
  + Equivalence Partitioning, Boundary Value Analysis, State Transition, Use Case Testing, Error Guessing, Exploratory Testing
* Prioritize test cases.

**Step 2: Testing Procedure**

* Begin with smoke testing on a new build.
* If smoke fails, wait for a stable build.
* Perform in-depth execution on stable builds using designed test cases.
* Multiple testers cover supported platforms; all defects logged in JIRA.
* Daily defect status shared to management.

**Testing Types:**

* Smoke & Sanity Testing
* Functional Testing
* Regression Testing & Retesting
* Usability Testing (of API endpoints)
* End-to-End Flow Testing (Login → Book Appointment)

**Best Practices:**

* Context-driven, shift-left, and exploratory testing.
* Comprehensive end-to-end scenario coverage.

## **Test Schedule**

| **Task** | **Time Duration** | **Dates** |
| --- | --- | --- |
| Creating Test Plan | 2 days | [To Be Filled] |
| Test Case Creation | 3 days | [To Be Filled] |
| Test Case Execution | 5 days | [To Be Filled] |
| Summary Reports Submission | 1 day | [To Be Filled] |

(Two Sprints, tentative: adjust as per project calendar)

## **Test Deliverables**

* Test Plan (this document)
* Test Scenarios & Cases
* Test Execution Report(s)
* Defect Logs/Reports
* Test Closure/Summary Report

## **Entry and Exit Criteria**

## **Requirement Analysis**

**Entry:**

* API requirements/documentation are available.

**Exit:**

* All requirements reviewed and understood.

## **Test Execution**

**Entry:**

* Test cases signed-off, API endpoint available.

**Exit:**

* Test case and defect reports available.

## **Test Closure**

**Entry:**

* Test execution and defect resolution complete.

**Exit:**

* Test summary/report submitted; all artifacts archived.

## **Tools**

* Test Design/Management: Word, Excel documents
* API Testing: Postman, curl
* Bug Tracking: JIRA
* Other: Mind map, screen capture (Snipping Tool)

## **Risks and Mitigations**

| **Risk** | **Mitigation** |
| --- | --- |
| Resource availability | Backup/alternate resource planning |
| API endpoint/server downtime | Tester works on documentation/prep |
| Incomplete/ambiguous API documentation | Early/continuous discussions with dev |
| Tight deadlines for testing | Dynamic ramp-up as needed |

## **Approvals**

Team will send documents for client/stakeholder approval:

* Test Plan
* Test Cases
* Summary Reports

Testing phases move forward only after formal approval.

| **Role** | **Name** | **Signature** | **Date** |
| --- | --- | --- | --- |
| Test Lead | [Name] |  |  |
| Project Manager | [Name] |  |  |
| Client Representative | [Name] |  |  |

## **Sample Login & Appointment Test Cases**

**Login API:**

* Valid Login (positive)
* Invalid Credentials (negative)
* Blank/Missing username/password
* Incorrect Content-Type

**Make Appointment API:**

* Successful appointment with all valid params
* Missing required fields
* Invalid date format or bad field data
* Use of unsupported Content-Type

**End of Test Plan**