

Project Planning & Management

For

Parabank

1. Project Description

Project Idea:

ParaBank is a demo online banking application provided by Parasoft. It simulates real-world online banking services such as account management, fund transfers, bill payments, loan requests, and profile updates.

Brief Description:

This project focuses on designing and executing a testing strategy for ParaBank. The objective is to validate that the system works according to its functional requirements, ensure the user experience is consistent across roles (Guest, Customer, Manager)

User Roles

1. Guest:

- Can browse public pages, recover login info, and register as a new customer.

2. Customer:

- Can log in, view account details, transfer funds, pay bills, request loans, update profile, and view transactions.

3. Manager (Admin role for testing scope):

- Can create, update, or delete customer accounts, approve/reject loans, deposit or withdraw funds, and view all transactions.

Core Functional Modules

- Login & Registration
- Account Overview & Transaction History
- Transfer Funds
- Bill Payment

- **Request Loan**
- **Deposit / Withdraw (Manager)**
- **Delete Account / Customer**
- **Update Contact Info**
- **Public Pages (Home, About Us, Services, etc.)**
- **Change Password**
- **Forgot Login Info**

Project Planning & Management

For
RESTFUL-BOOKER (API Testing Project)

1. Project Description

Project Idea:

Restful-Booker is a publicly available RESTful API designed for practicing API testing. It provides endpoints to create, retrieve, update, and delete hotel bookings. The system supports authentication, booking operations, filtering, and basic health checks.

Brief Description:

This project focuses on designing and executing a complete API testing strategy for the Restful-Booker application.

The objective is to validate that all API endpoints behave according to their functional requirements, ensure proper authentication flow, verify CRUD operations for bookings, and assess error handling for invalid inputs, missing fields, or improper HTTP methods.

The project includes both **manual testing (Postman)** and **automation testing (REST Assured / Java / TestNG)**.

Automation covers the same functional scope but prioritizes high-value, repeatable API scenarios.

Core Functional Modules

1. Authentication

- Create Token (POST /auth)
- Required for updating or deleting bookings
- Validates correct credentials and token generation

2. Booking Management

- **Create Booking** (POST /booking)
Validates creation with full JSON body, date formats, price, deposit status.

- **Get All Booking IDs** (GET /booking)
Ensures API returns a list of IDs and supports filtering.
- **Get Booking by ID** (GET /booking/{id})
Returns booking details including name, dates, price, deposit status.
- **Update Booking (FULL Update)** (PUT /booking/{id})
Requires authentication. Validates all fields updated.
- **Partial Update** (PATCH /booking/{id})
Updates only requested fields (e.g., firstname, price).
- **Delete Booking** (DELETE /booking/{id})
Requires authentication. Ensures correct 201/204 deletion response.

3. Booking Filters

- Filter by Firstname
- Filter by Lastname
- Filter by Check-in / Check-out
- Combination filters
Used to validate that search endpoints return accurate booking IDs.

4. API Health Check

- /ping endpoint to verify service availability.

User Roles (API Context)

Restful-Booker does not have UI-based roles like ParaBank, but the API effectively operates with:

1. Unauthenticated User

- Can create bookings
- Can get booking IDs
- Can get booking by ID
- Cannot update or delete bookings

2. Authenticated User (Token Required)

- Can create token
 - Can update bookings (PUT / PATCH)
 - Can delete bookings
-

Testing Scope

Manual Testing (Postman)

Covers:

- Auth token creation
- CRUD operations
- Input validation
- Error codes (400, 403, 404, 405)
- Boundary value testing (dates, price, invalid IDs)
- Negative scenarios (wrong payloads, missing fields)
- Filtering and data extraction
- Health check endpoint

Automation Testing

Automated using: **REST Assured + Java + TestNG**

Covers:

- Create token
- Create booking
- Get booking IDs
- Get single booking
- Put & Patch update
- Delete booking
- Assertions for status codes and JSON responses
- Data-driven tests for positive & negative scenarios

Key API Validations

- Response codes (200, 201, 400, 404, 403, 405)
- JSON schema validation
- Required fields handling
- Token authorization
- Data consistency after update/delete
- Filtering logic
- Error responses for wrong HTTP methods

Summary

This project ensures end-to-end API testing coverage for Restful-Booker using both manual and automated testing techniques. The testing validates all functional modules such as authentication, booking creation, retrieval, filtering, updating, and deletion—ensuring the reliability, accuracy, and security of the system.