

← → ↺

localhost:8181/user_db/login.do?jsessionid=658a93d5944725f8feebe995b635ae07

🔑 🔍 ☆

🔥 🔗

☒ Auto commit 🔗

🔍

Max rows: 1000

🟢 🔴

🔧

Auto complete Off

Auto select On

?

📁 jdbc:h2:mem:testdb

📁 CASH

📁 CASHIER

📁 PRODUCT

📁 SALES

📁 INFORMATION_SCHEMA

👤 Users

📄 H2 2.1.214 (2022-06-13)

Run Run Selected Auto complete Clear SQL statement:

SELECT * FROM CASH;
SELECT * FROM CASHIER;
SELECT * FROM PRODUCT;
SELECT * FROM SALES;

SELECT * FROM CASH;

ID	FLOOR
2	56
3	34
99	50

(3 rows, 3 ms)

SELECT * FROM CASHIER;

ID	SURNAME_NAME
2	Greta
99	Bella

(2 rows, 0 ms)

SELECT * FROM PRODUCT;

ID	NAME	PRICE
2	biscuit	76
3	coke	98
4	coke	98
99	Chocolate	78

(4 rows, 0 ms)

SELECT * FROM SALES;

ID	REGISTERED_AT	CASH_ID	CASHIER_ID	PRODUCT_ID
99	2025-07-02 00:00:00	99	99	99

(1 row, 0 ms)

REST_API_TEST

07.07.2025

Hack Rush

19th May Batch

QE Full Stack domain

IBM, BCIT

Overview

This project includes API test cases for a backend system managing a department store, built with REST APIs. The test cases cover CRUD operations for departments, products, customers, and inventory through RESTful endpoints. They follow the clean layered architecture: Controller, Service, and Repository. They are designed to ensure smooth integration with frontend applications or other systems.

Deliverables

1. **API Test Cases:** Cover CRUD operations for Departments, Products, Customers, and Inventory. Ensure workflows and data persistence in the backend.
2. **Automated Test Scripts:** JUnit/TestNG scripts validate all REST API endpoints. Check HTTP responses and database state consistency.
3. **Test Documentation:** Templates document test steps, inputs, expected and actual results. Provide clear tracking of test execution outcomes.
4. **Demo Execution:** Showcase automated API testing with H2 database updates. Verify all core scenarios in a working demonstration.

Project Highlights

Frameworks components:

- **Rest Assured:**
 - To create the request and response specifications.
 - To send requests and receive responses.
- **Hamcrest:**
 - The assertion library to add verifications to RESTAssured tests.
- **JUnit:**
 - To create and execute automated test cases.
- **Pact:**
 - To create the Consumer-side tests.
 - To create Provider-side verifications.
 - To create the contract between Consumer and Provider.
 - To create the mock server to run tests against.

Agile Approach:

- Regular tracking and sprint planning done via **JIRA**.
- Code collaboration managed via **GitHub**.
- Daily standups to update teammates.

Test Coverage:

- Login
- SQLExecution
- SchemaValidation
- DepartmentCRUD
- ProductCRUD
- CustomerCRUD
- InventoryCRUD
- APIResponse
- ErrorHandling
- DataPersistence
- DatabaseCleanup

Use-case Diagram

