



Put your ORDS API to the Acid Test

NLOUG Database Cloud & Developer Day
November, 13th 2025 (13:45 - 14:35 | Aarde & Lucht)



Robert Marz
DATABEE
Die IT-Architekten

Primary Role

Senior Technical Architect
with database centric view of the world





Eine starke Community für Oracle Datenbank Benutzer

Ziele: Wissensvermittlung für Entwickler und DBAs

Angebote: MeetUps, Podcasts, Live Events

Projekte: Veranstaltung zum Kennenlernen

Kontakt: info@ora2know.de



ora2know

The German Oracle Database Community



The Oracle ACE Program

400+ technical experts helping peers globally



- The Oracle ACE Program recognizes and rewards community members for their technical and community contributions to the Oracle community
- 3 membership levels: Director, Pro, and Associate
- Nominate yourself or a colleague at ace.oracle.com/nominate
- Learn more at ace.oracle.com

✉ aceprogram_ww@oracle.com

Facebook.com/OracleACEs

@oracleace

Oracle ACE Program Group



SYMPOSIUM⁴²

Created by the community, to support the community

Sharing of reliable knowledge

Supporting the various user groups and individuals



@sym_42



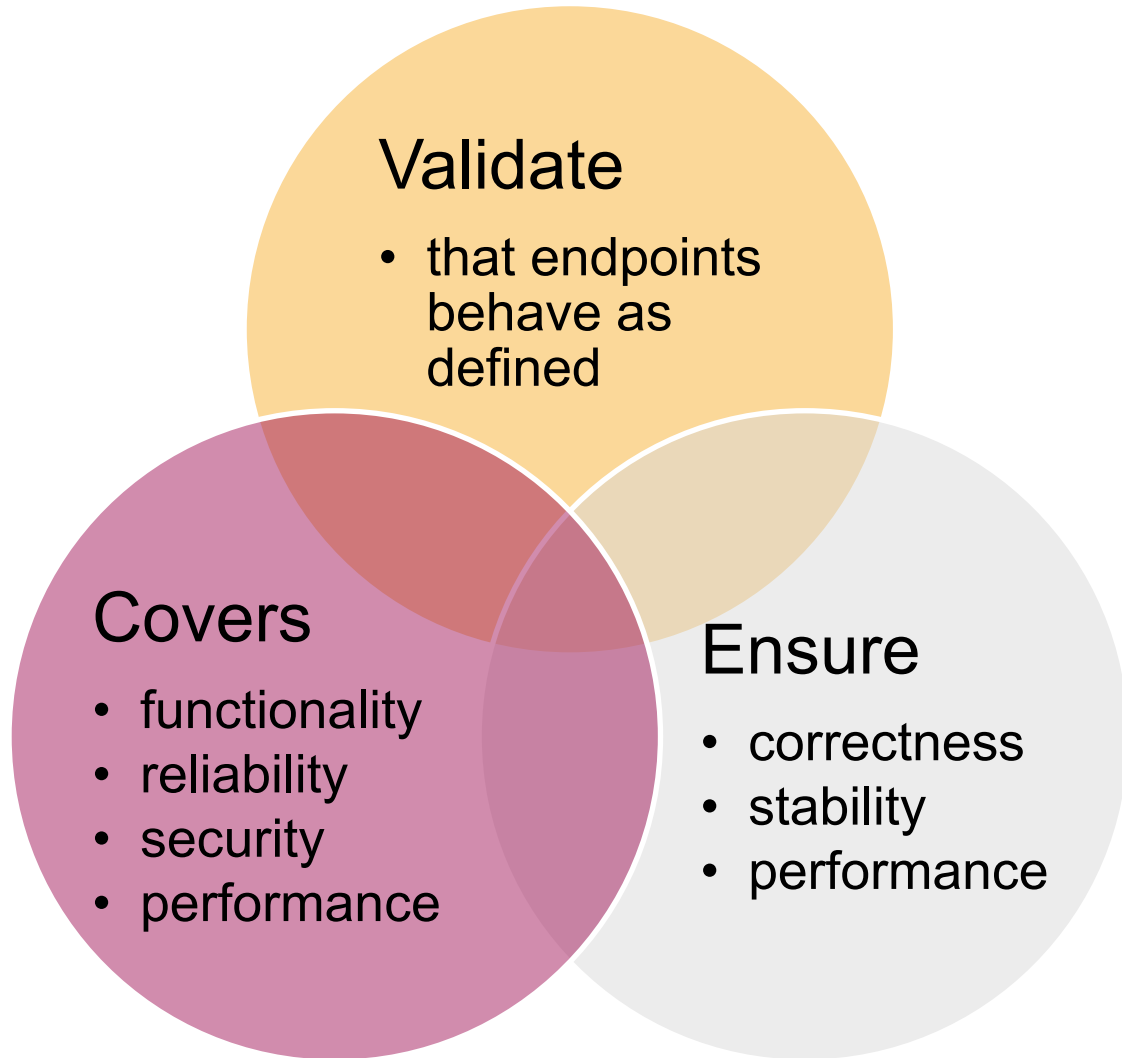
<https://sym42.org/>



API Testing

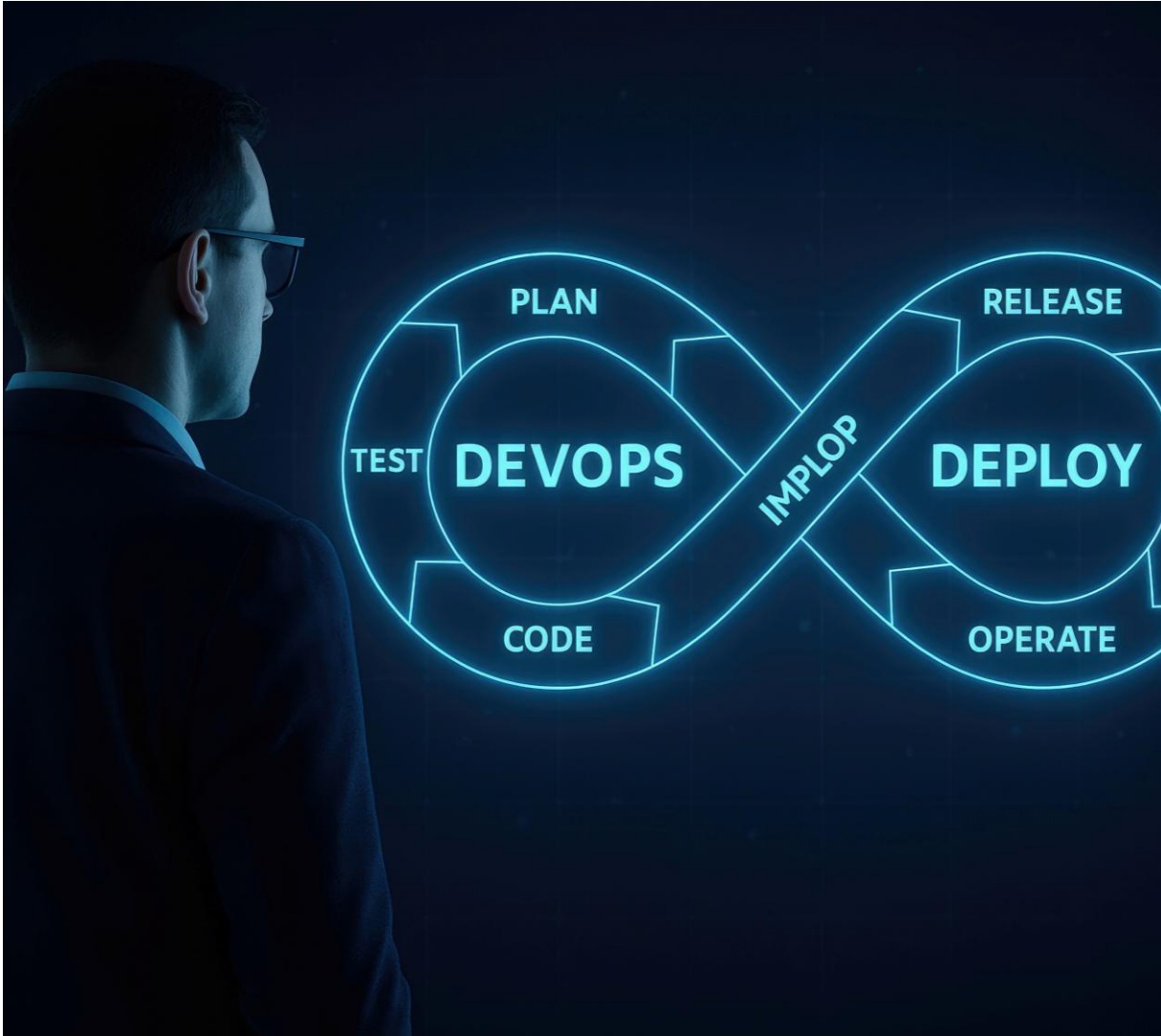


What Is API Testing?





Why Test APIs?



Early detection of logic or integration errors

Confident deployments across environments

Automation-ready (CI/CD)

Supports regression prevention



Tools & Context (Preview)



REST clients

- Postman
- Insomnia

Unit tests

- utPLSQL

Load tests

- JMeter
- K6

Contract validation

- JSON Schema

A man with a beard and long hair, wearing a striped shirt and a dark vest, is working on a computer motherboard. He is using a pair of orange tweezers to handle a component on the board. The desk is cluttered with various tools and hardware, including a keyboard, a mouse, a screwdriver, a multimeter, and several hard drives. A large orange and red banner is overlaid on the image, containing the text "Unit Testing".

Unit Testing



Why Unit Tests Matter

Verify

- PL/SQL logic independently of ORDS

Catch

- regressions early

Improve

- developer confidence
- code quality

Enable

- automated CI/CD quality gates





What is utPLSQL



Open-source testing framework for PL/SQL

Similar to JUnit, but for Oracle code

Provides assertions, setup/teardown, and reporting

Integrates with SQL Developer, SQLcl, Jenkins, GitHub Actions



Writing Tests

```
CREATE OR REPLACE PACKAGE test_api AS
  --%suite(Test API Business Logic)
  --%test(should return valid JSON response)
  PROCEDURE test_response;
END;
/
CREATE OR REPLACE PACKAGE BODY test_api AS
  PROCEDURE test_response IS
    l_result CLOB;
  BEGIN
    l_result := api_pkg.get_data;
    ut.expect(l_result).to_match('.*"status":"OK".*');
  END;
END;
/
-- To run the tests, use the following command in SQL*Plus or SQL Developer:
BEGIN
  ut.run('test_api');
END;
/
```



Unit Tests: Demo



API Tests with REST Clients

400 Bad Request

19.19 s



Visualize





REST Client Basics

REST clients

- simulate consumers of your ORDS APIs

Send

- HTTP requests

inspect

- Responses

Ideal for

- manual tests
- Prototyping
- debugging



POSTMAN



Insomnia



cURL – the mother of all REST Clients

Lightweight tool for HTTP requests

- CLI tool
- Library

Installed

- on almost every system

Perfect for

- quick tests
- automation

Forms the foundation

- for many REST tools
 - Postman
 - Newman
 - K6

```
curl -s http://localhost:8080/ords/demo/todo/ | jq
```

```
curl -s -X POST -H "Content-Type: application/json" \  
-d '{"title":"from curl"}' \  
http://localhost:8080/ords/demo/todo/ | jq
```





Scripting in REST Clients

```
// pm is the Postman object
pm.test("Status test", function () {
  pm.response.to.have.status(201);
});
// Check if response is valid JSON
pm.test("Response is valid JSON", function () {
  pm.expect(function () {
    JSON.parse(pm.response.text());
  }).not.to.throw();
});
// Stores the jobId in a collection variable
pm.collectionVariables.set("jobId", pm.response.json().jobId);

pm.sendRequest({
  url: url,
  method: 'POST',
  header: {'Content-Type': 'application/x-www-form-urlencoded'},
  body: {mode: 'raw', raw: data}
}, function(err, response) {
  // Set the environment variable for token
  // [...]
  console.log("new token: ", payload);
});
```

Use JavaScript

- for assertions and automation

Validate responses

- status codes
- body content
- headers

Chain requests

- for end-to-end workflows



Managing Environments



Store variables

base URLs

credentials

tokens



Switch between environments easily

dev

test

prod



Use dynamic values

for authentication

or test data



Automating Test Runs



Run

- full collections of tests automatically



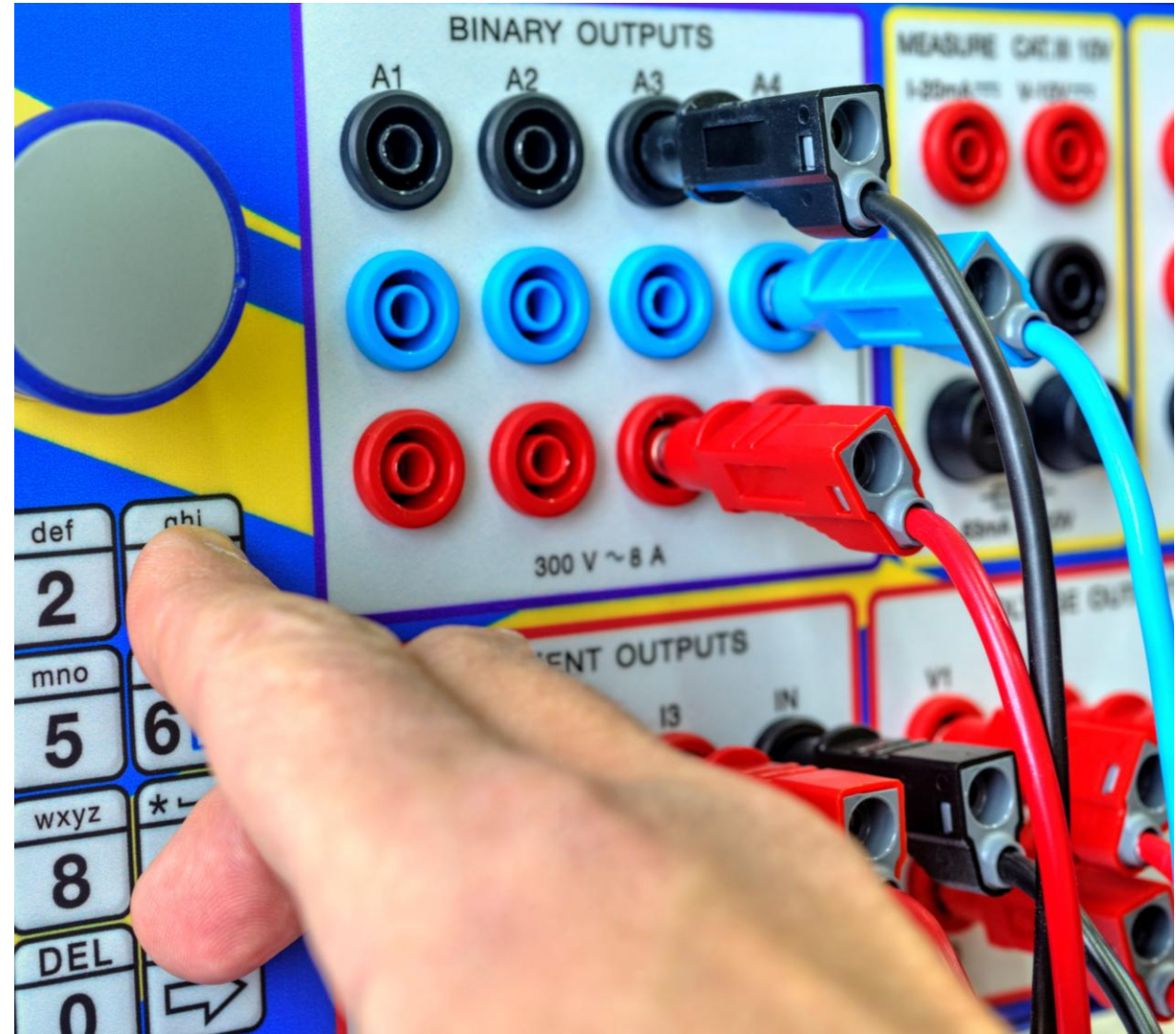
Integrate

- with Newman (Postman CLI runner) for CI/CD pipelines



Export

- results as reports (HTML, JUnit XML)





API Testing with Rest Clients





Stress and Load Tests



Why Stress and Load Tests

Validate performance

under real traffic

Detect bottlenecks

before users do

Ensure stability

scalability

resilience

Essential before production

deployment





Load vs Stress Testing



Load Test

Goal

- Verify normal operation under expected users

Example

- 100 users → stable responses

Stress Test

Goal

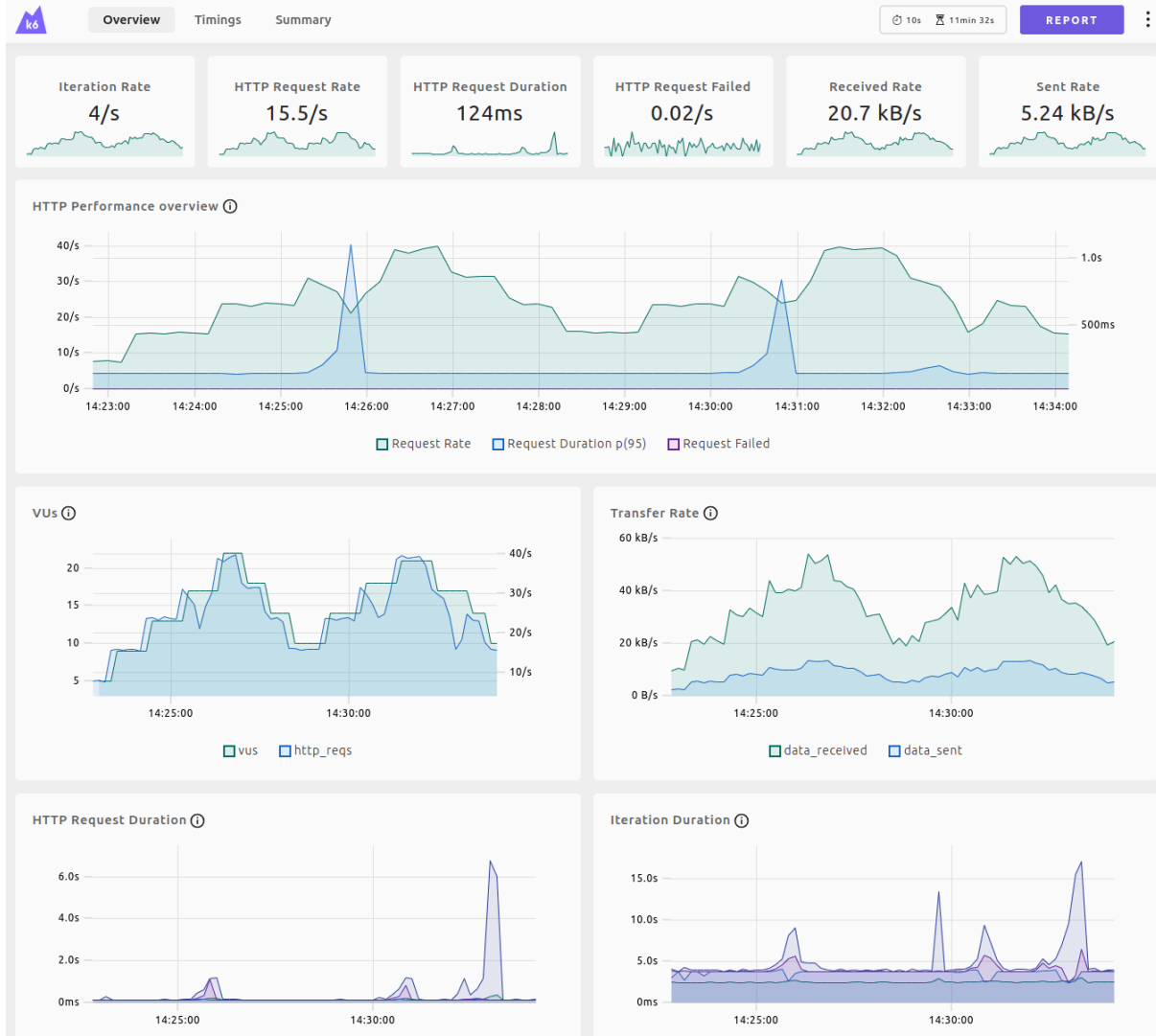
- Push beyond limits

Example

- 500 users → observe failure behavior



Grafana k6 Overview



Modern, developer-friendly load testing tool

Open Source

Scriptable Engine written in Go

Tests authored in JavaScript or TypeScript

CLI + cloud execution

Great for CI/CD integration

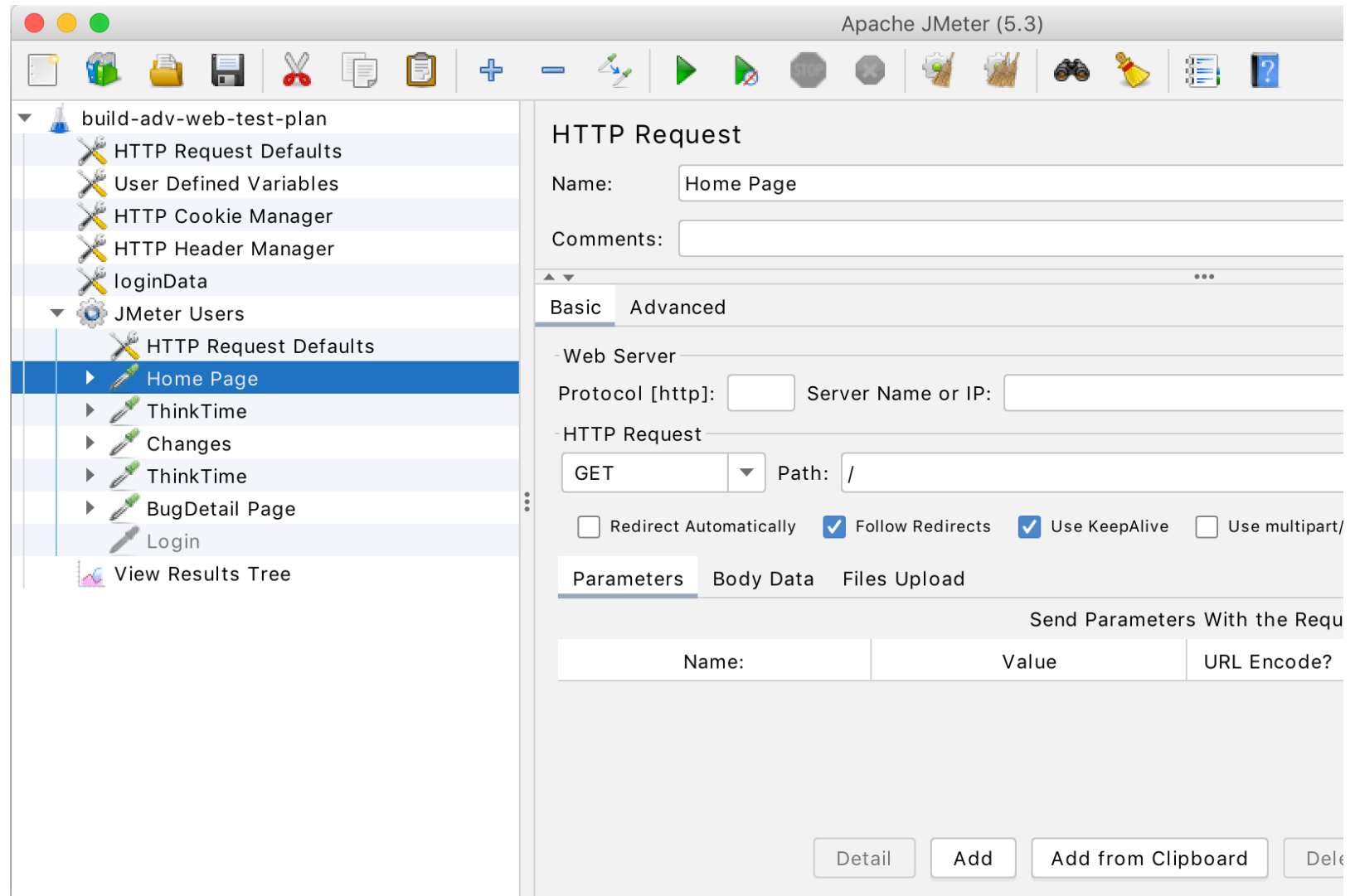
Apache JMeter Overview

GUI-based open-source load tester

Broad protocol support (HTTP, JDBC, MQ...)

Good for complex test plans and graphs

Produces rich reports and charts



The screenshot displays the Apache JMeter (5.3) interface. On the left, a tree view shows a test plan named 'build-adv-web-test-plan' containing several elements: HTTP Request Defaults, User Defined Variables, HTTP Cookie Manager, HTTP Header Manager, loginData, JMeter Users, HTTP Request Defaults, Home Page (selected), ThinkTime, Changes, ThinkTime, BugDetail Page, and Login. Below the tree is a 'View Results Tree' button. The right pane shows the configuration for the selected 'Home Page' element, which is an HTTP Request. The 'Name' field is set to 'Home Page'. The 'Basic' tab is active, showing the 'Web Server' section with 'Protocol [http:]' and 'Server Name or IP:' fields. The 'HTTP Request' section shows 'GET' selected for the method and '/' for the path. Checkboxes for 'Redirect Automatically', 'Follow Redirects', 'Use KeepAlive', and 'Use multipart/' are visible. Below this is the 'Parameters' tab, which has a table for defining request parameters. The table has columns for 'Name:', 'Value', and 'URL Encode?'. At the bottom right, there are buttons for 'Detail', 'Add', 'Add from Clipboard', and 'Delete'.



API-Stress- and Load-Tests

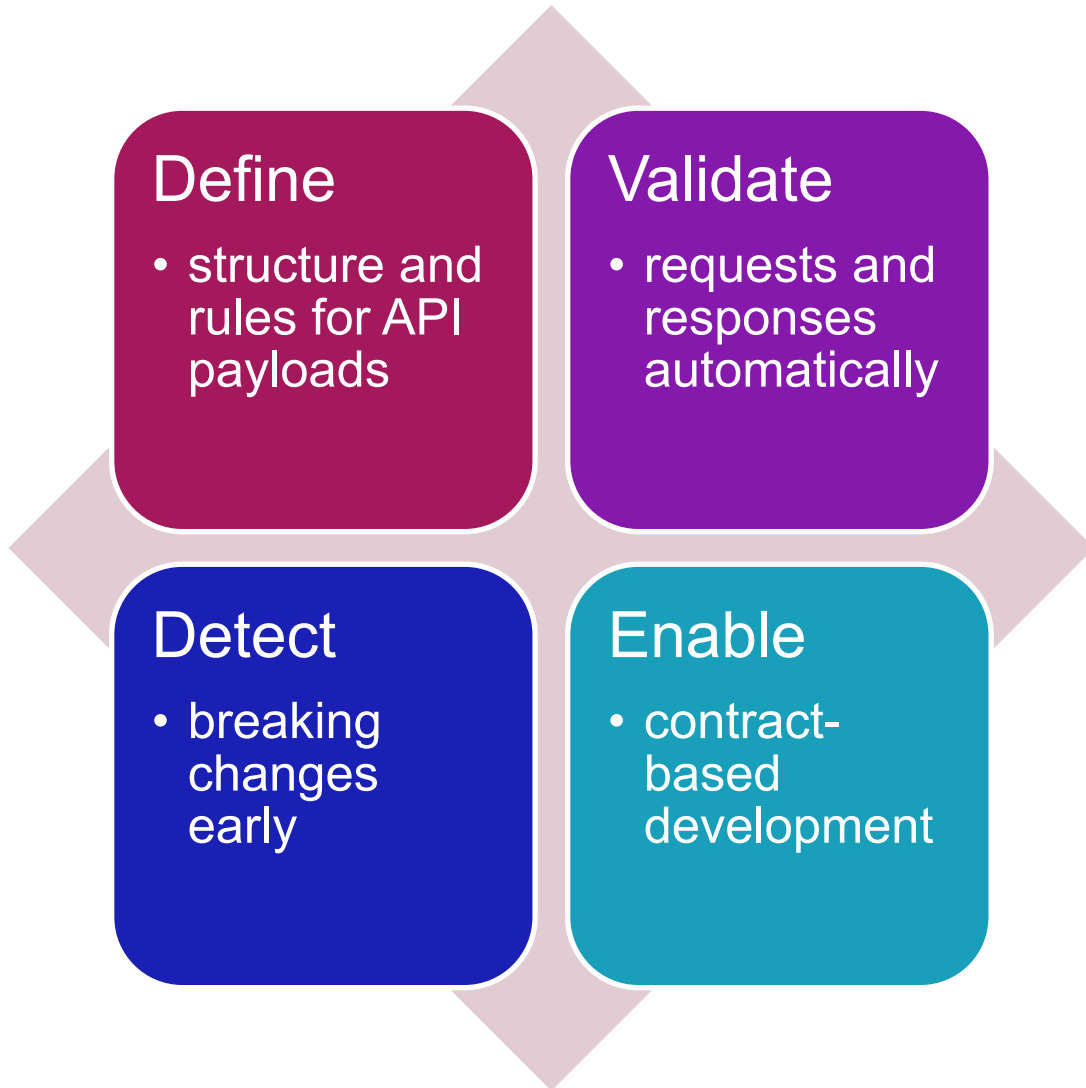


A horror-themed background image featuring Jason Voorhees from the Friday the 13th franchise. He is wearing his signature white hockey mask with multiple ventilation holes and a dark, heavy jacket. He is holding a large, curved machete in his right hand, which is raised. His left hand is extended forward, palm facing the viewer. The scene is lit with dramatic, low-key lighting, with a strong blue light on the left and a yellowish light on the right, creating a menacing atmosphere. In the background, there are faint, blurry silhouettes of people, possibly in a camp setting.

JSON Schemas

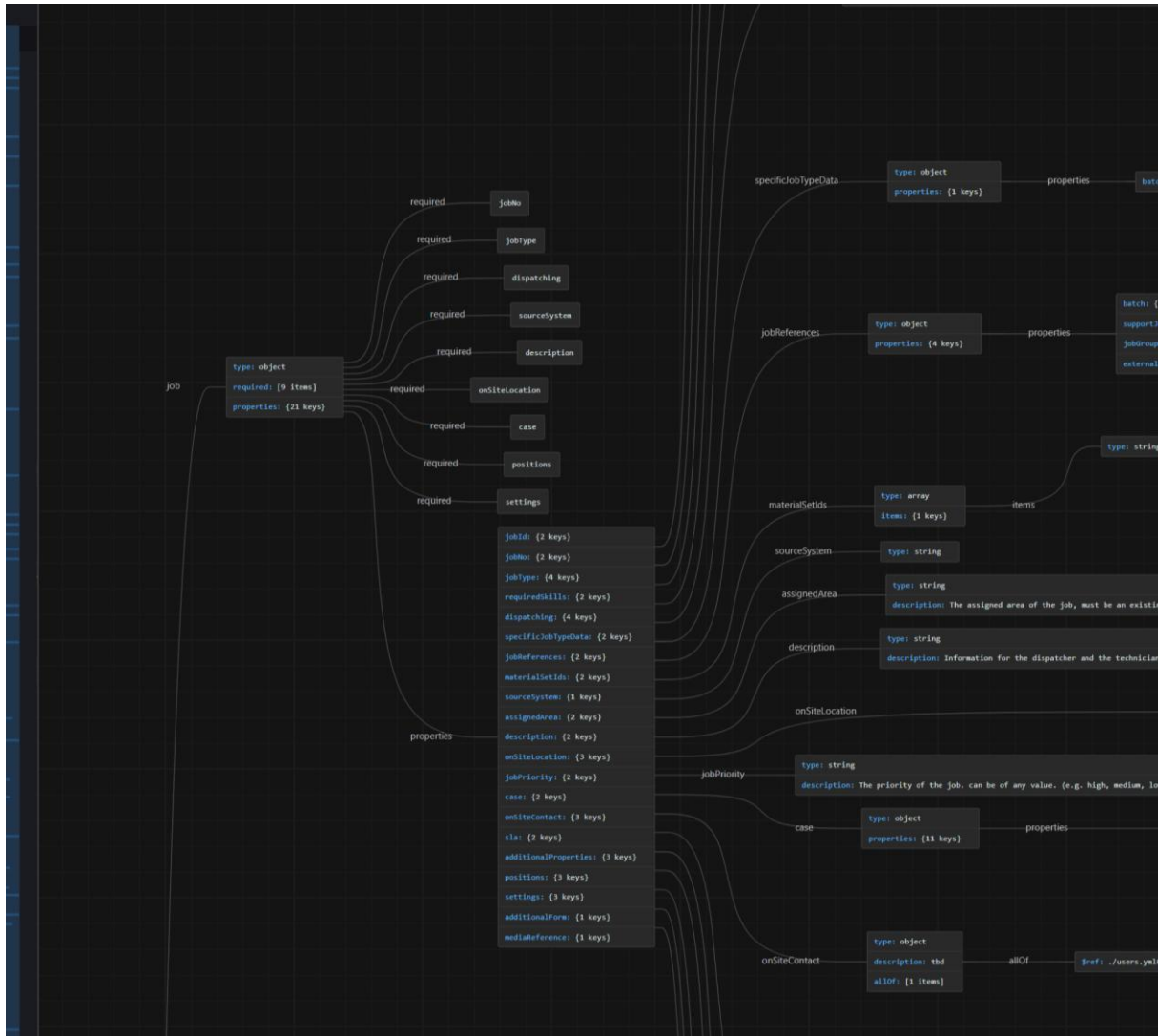


Why JSON Schema





Testing JSON Payloads



Validate incoming

- payloads against schema

Validate outgoing

- responses for consistency

Prevent

- Nulls
- type errors
- missing fields

Use schema

- as single source of truth



Automate Schema Extraction

Extract JSON Schemas

- automatically from openapi.json

Keep schema

- and API in sync

Use scripts or CI steps

- for regeneration

Commit schemas

- to repo for version control

```
#!/bin/bash
# Extract all JSON Schemas from OpenAPI spec
# Requires: jq (https://stedolan.github.io/jq/)

API_FILE="openapi.json"
OUT_DIR="./schemas"

mkdir -p "$OUT_DIR"

# Extract each schema from the OpenAPI spec
jq -r '.components.schemas | keys[]' "$API_FILE" | while read -r SCHEMA
do
    jq ".components.schemas[\"$SCHEMA\"]" "$API_FILE" > "$OUT_DIR/${SCHEMA}.json"
    echo "Extracted schema: $SCHEMA"
done

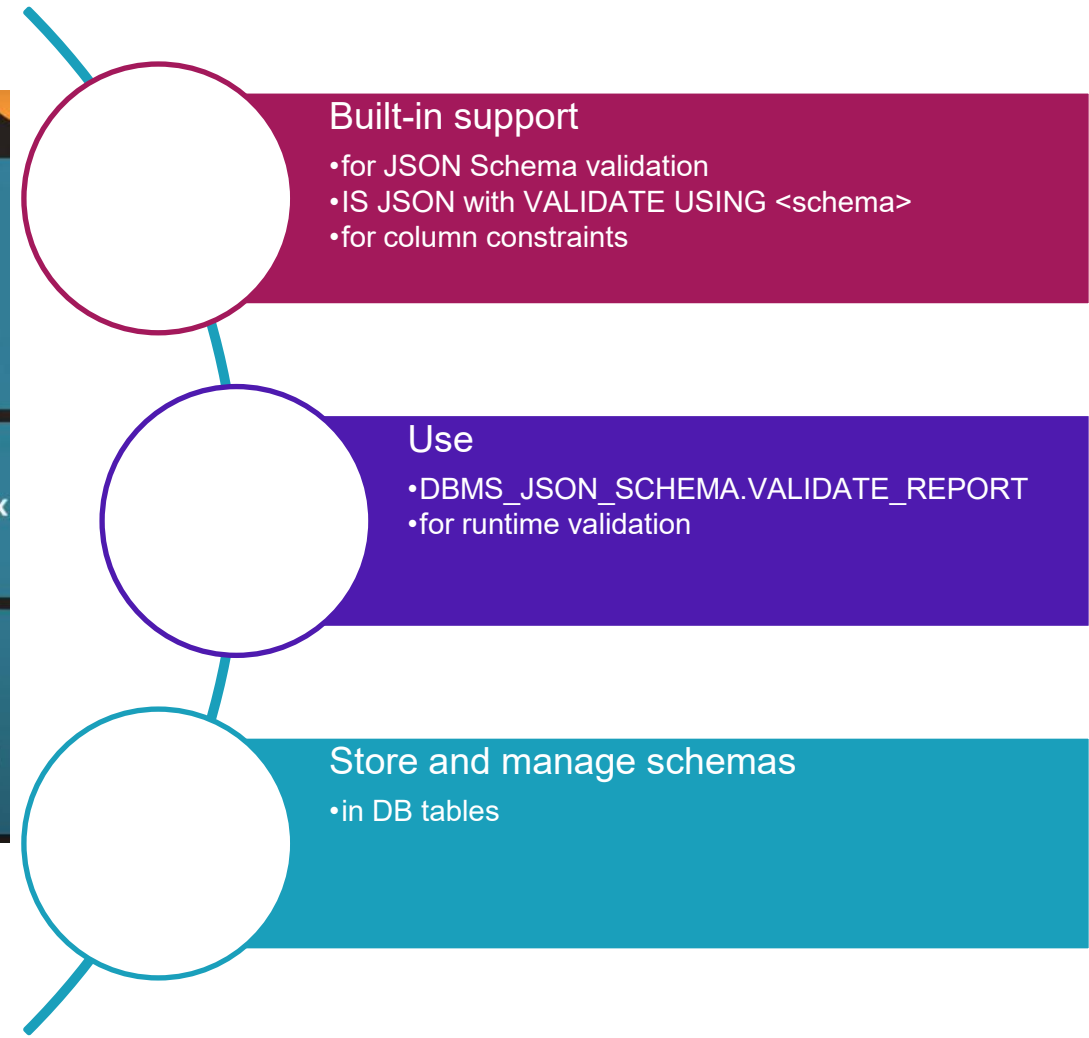
echo "✅ All schemas exported to $OUT_DIR/"
```



JSON Schema in Oracle 23ai / 26ai

Oracle Database 23ai JSON features

- JSON Type Domains
- JSON data type (since 21c)
- Augmentation for JSON Relational Duality
- AI Vector support in JSON type
- JSON Collection Tables
- JSON / Relational Duality
- JSON Schema validation and describe
- Support for MongoDB's Aggregation pipeline
- JSON in Property Graphs
- JSON Search Index Sync Auto
- Fast refreshable JSON Materialized Views
- \$sql command
- Multi-value any-type index
- PL/SQL record/collection JSON mappers
- improved Exadata smart scans
- SQL Boolean to JSON mapping
- JSON External Tables
- Indexes in MongoDB API: singleton Index, array index, TTL index, Search index





JSON Schema Validation in Azure APIM

Oracle 19c has no JSON Schema Support

Import OpenAPI spec → APIM automatically creates schema policies

Central enforcement for multiple APIs

API Management Systems like Azure APIM to the the Rescue

Validate requests/responses before reaching backend

Combine with monitoring and alerts



Azure API Management



JSON Schema Validation



Conclusion





Put your ORDS API to the Acid Test

Validate your ORDS APIs on every layer

Use **utPLSQL** for backend unit tests

Test API contracts via **REST** clients

Simulate real-world usage with **K6** or **JMeter**

Enforce structure and data integrity with **JSON Schema**

PLEASE

**DO
TRY THIS
AT HOME**