

Prompt Engineering

Prompt Logs:

Logs include history of attempts to get the desired output. Specifying Role was helpful only for certain prompts wherein it didn't work for some prompts.

Prompt:

Role: Senior Automation QA Engineer

My Ask: Add a short contextual paragraph for the following bullet points

Functional testing verifies the expected success behavior. A primary success scenario is submitting a well-formed order request that includes required fields and multiple items. The service is expected to return 200 OK and publish a single enriched message to the queue. The validation also confirms that each sku and quantity from the input is preserved and that enrichment adds the required metadata fields without mutating the meaning of the original order.

Prompt:

Add a short contextual paragraph under Test Strategy section in a Test Plan document for the following bullet points

Testing will be executed across multiple layers to ensure correctness and resilience. Testing focuses on verifying correct interaction between internal components and external dependencies such as the Card Catalog service and the SQS messaging layer. Integration and end-to-end tests validate that requests flow correctly through validation, enrichment, and publishing stages, while negative and failure scenarios ensure the system behaves predictably under error conditions. Test execution is supported through isolated local environments using mocks with automated runs integrated into CI pipelines to ensure continuous quality and early defect detection.

Prompt:

Add a short contextual paragraph in 1 to 2 lines under Non-Functional section in a Test Plan document for the following bullet points

Non-functional testing is included to assess performance, resilience, and fault tolerance, ensuring the service can operate reliably in real-world production environments

Prompt:

Role: Senior Software Test Engineer

My Ask: Develop an automation framework design for API validation using Playwright in mvc concept

Prompt Improvisation:

Attempt 1:

Role : Senior QA/Automation Engineer, Ask : Give me a flowchart for below mentioned Pipeline CI Pipeline Structure
The automation test suite triggers test run for end-to-end workflows or sanity test flows in a cadence on every change to OAS in the Git. Pipeline workflow stages can be developed using AWS cdk or Jenkins. A standalone pipeline has unit tests coverage part of code commit achieving fast fail, followed by API framework, integration tests providing functionality coverage, north-bound and south-bound dependency coverage using LocalStack or mock data. Test Execution – AWS Device Farm/VM/Grid Tests execute parallelly either in local VM environment or Grid or AWS Device Farm in pipeline approval workflows. Test Environments Git Code Commit ◇ Triggers Tests in QA Env ◇ Pre-Prod Env ◇ Prod Deployment ◇ Monitoring ◇ Dashboard • Test suites (API, Integration and Regression suites) are executed in QA environment and upon successful execution, the pipeline promotes to next stage which is Pre-Prod environment. • The Test-suites (API and Integration) are executed in Pre-Prod environment with either mock, seed or dynamic data. After successful completion pipeline merges with Dev pipeline as a workflow approval and promotes Prod deployment. As part of Prod deployment and post deployment – a canary pipeline validates P0 end to end test flows every one hour once using Jenkins Job scheduler. Monitoring & Dashboard The pipeline publishes HTML test reports for every stage which is provided by Playwright's features. Artifacts - logs, traces, screen captures (in case of UI), request/response payloads, queue message bodies on failure and gate merges/releases - can be accessed via Jenkins or Cloud Watch depending on execution environment.

Role : Senior QA/Automation Engineer , Ask : Give me a flowchart for below mentioned Pipeline CI Pipeline Structure

The automation test suite triggers test run for end-to-end workflows or sanity test flows in a cadence on every change to OAS in the Git. Pipeline workflow stages can be developed using AWS cdk or Jenkins. A standalone pipeline has unit tests coverage part of code commit achieving fast fail, followed by API framework, integration tests providing functionality coverage, north-bound and south-bound dependency coverage using LocalStack or mock data. Test Execution – AWS Device Farm/VM/Grid Tests execute parallelly either in local VM environment or Grid or AWS Device Farm in pipeline approval workflows.

Test Environments

Git Code Commit → Triggers Tests in QA Env → Pre-Prod Env → Prod Deployment → Monitoring → Dashboard

- Test suites (API, Integration and Regression suites) are executed in QA environment and upon successful execution, the pipeline promotes to next stage which is Pre-Prod environment.
- The Test-suites (API and Integration) are executed in Pre-Prod environment with either mock, seed or dynamic data. After successful completion pipeline merges with Dev pipeline as a workflow approval and promotes Prod deployment. As part of Prod deployment and post deployment – a canary pipeline validates P0 end to end test flows every one hour once using Jenkins Job scheduler.

Monitoring & Dashboard

The pipeline publishes HTML test reports for every stage which is provided by Playwright's features. Artifacts - logs, traces, screen

Output:

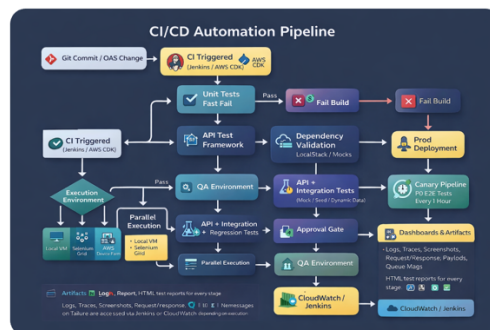
✓ CI/CD Automation Pipeline – Flowchart (Mermaid)

```
graph TD
    A[Git Commit / OAS Change] --> B[CI Triggered  
(Jenkins / AWS CDK)]
    B --> C[Unit Tests  
Fast Fail]
    C --> D[Pass]
    C --> E[API Test Framework]
    E --> F[Dependency Validation  
LocalStack / Mocks]
    F --> G[Execution Environment]
    G --> H[Local VM | Parallel Execution]
    G --> I[Selenium Grid]
    G --> J[AWS Device Farm]
    H --> K[QA Environment]
    I --> K
    J --> K
    K --> L[API + Integration + Regression Tests]
    L --> M[Approval Gate]
    M --> N[Prod Deployment]
    N --> O[Canary Pipeline  
Post-Deploy Every 1 Hour]
    O --> P[Dashboards & Artifacts]
    P --> Q[CloudWatch / Jenkins]
```

Attempt 2: Create a visual diagram image

Output:

Image created • CI/CD automation pipeline flowchart



Attempt 3:

Role : Senior QA/Automation Engineer , Ask : Give me a visual flowchart diagram for below mentioned Pipeline CI to add to my Automation Strategy document "Pipeline Structure The automation test suite triggers test run for end-to-end

workflows or sanity test flows in a cadence on every change to OAS in the Git. Pipeline workflow stages can be developed using AWS cdk or Jenkins. A standalone pipeline has unit tests coverage part of code commit achieving fast fail, followed by API framework, integration tests providing functionality coverage, north-bound and south-bound dependency coverage using LocalStack or mock data. Test Execution – AWS Device Farm/VM/Grid Tests execute parallelly either in local VM environment or Grid or AWS Device Farm in pipeline approval workflows. Test Environments Git Code Commit ◇ Triggers Tests in QA Env ◇ Pre-Prod Env ◇ Prod Deployment ◇ Monitoring ◇ Dashboard • Test suites (API, Integration and Regression suites) are executed in QA environment and upon successful execution, the pipeline promotes to next stage which is Pre-Prod environment. • The Test-suites (API and Integration) are executed in Pre-Prod environment with either mock, seed or dynamic data. After successful completion pipeline merges with Dev pipeline as a workflow approval and promotes Prod deployment. As part of Prod deployment and post deployment – a canary pipeline validates P0 end to end test flows every one hour once using Jenkins Job scheduler. Monitoring & Dashboard The pipeline publishes HTML test reports for every stage which is provided by Playwright's features. Artifacts - logs, traces, screen captures (in case of UI), request/response payloads, queue message bodies on failure and gate merges/releases - can be accessed via Jenkins or Cloud Watch depending on execution environment"

ChatGPT 5.2

Role : Senior QA/Automation Engineer , Ask : Give me a visual flowchart diagram for below mentioned Pipeline CI to add to my Automation Strategy document "Pipeline Structure The automation test suite triggers test run for end-to-end workflows or sanity test flows in a cadence on every change to OAS in the Git. Pipeline workflow stages can be developed using AWS cdk or Jenkins. A standalone pipeline has unit tests coverage part of code commit achieving fast fail, followed by API framework, integration tests providing functionality coverage, north-bound and south-bound dependency coverage using LocalStack or mock data. Test Execution – AWS Device Farm/VM/Grid Tests execute parallelly either in local VM environment or Grid or AWS Device Farm in pipeline approval workflows. Test Environments Git Code Commit Triggers Tests in QA Env Pre-Prod Env Prod Deployment Monitoring Dashboard • Test suites (API, Integration and Regression suites) are executed in QA environment and upon successful execution, the pipeline promotes to next stage which is Pre-Prod environment. • The Test-suites (API and Integration) are executed in Pre-Prod environment with either mock, seed or dynamic data. After successful completion pipeline merges with Dev pipeline as a workflow approval and promotes Prod deployment. As part of Prod deployment and post deployment – a canary pipeline validates P0 end to end test flows every one hour once using Jenkins Job scheduler. Monitoring & Dashboard The pipeline publishes HTML test reports for every stage which is provided by Playwright's features. Artifacts - logs, traces, screen captures (in case of UI), request/response payloads, queue message bodies on failure and gate merges/releases - can be accessed via Jenkins or Cloud Watch depending on execution environment"

Output:

Image created • Automation pipeline CI process flow



Attempt 4:

Give me visual flowchart for CI Pipeline for below --> Git Code Commit → Triggers Tests in QA Env → Pre-Prod Env → Prod Deployment → Monitoring → Dashboard

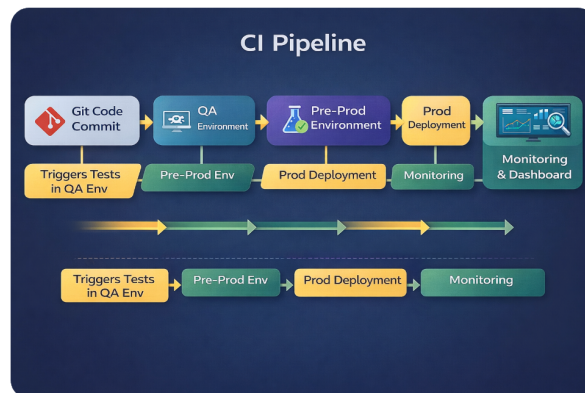
ChatGPT 5.2



Git Code Commit → Triggers Tests in QA Env → Pre-Prod Env → Prod Deployment → Monitoring → Dashboard

Give me visual flowchart for CI Pipeline for below --> Git Code Commit → Triggers Tests in QA Env → Pre-Prod Env → Prod Deployment → Monitoring → Dashboard

Image created • CI pipeline flowchart infographic



Automation Git Repo

https://github.com/anithaparamashivam/oas_e2e/tree/main

- Used GitHub Co-pilot
- Co-pilot created the project structure based on the project structure in Test Plan.
- Co-pilot helped me create Config files and dependencies
- Co-pilot helped debugging errors in Tests, Utils