# API Automation Framework - Detailed Explanation

## 1. Introduction

This document explains the architecture, execution flow, theoretical design, and detailed layer-by-layer structure of the API Automation Framework built using Node.js, Cucumber (BDD), Supertest, and AJV schema validation.

## 2. Architectural Design

The framework follows a layered architecture combined with Behavior Driven Development (BDD). Layers include: - Feature Layer (Business readable scenarios) - Step Definition Layer (Glue code) - API Service Layer (GET, POST, PUT, DELETE classes) - Core Engine Layer (ApiMethods.js) - Transport Layer (Supertest HTTP client)

## 3. Execution Flow

1. npm run spec command is executed. 2. package.json triggers cucumber-js. 3. cucumber.js loads feature files and step definitions. 4. Feature file steps are matched with step definitions. 5. Step definitions call API layer classes. 6. API classes use ApiMethods request engine. 7. Supertest sends HTTP request to server. 8. Response is validated (status + schema). 9. Report is generated in JSON and optionally converted to HTML.

## 4. Core Components

package.json: Defines project metadata, scripts, and dependencies. cucumber.js: Configures step definitions, feature paths, reporting formats, and tag filtering. ApiMethods.js: Central HTTP engine handling: - Request building - Header management - Token handling - Status validation - Schema validation - Dynamic payload building API Layer (GET, POST, PUT, DELETE): Encapsulates business-level API operations. Step Definitions: Connect Gherkin language steps to JavaScript logic. Feature Files: Describe business workflows using Gherkin syntax.

## 5. World Object (State Management)

Cucumber provides a World object per scenario. It allows sharing data (like bookingId) between steps. This ensures scenario-level state isolation.

## 6. Validation Strategy

Two levels of validation: 1. HTTP Status Code Validation 2. JSON Schema Validation using AJV This ensures both contract correctness and data integrity.

## 7. Advantages of Framework

- Separation of Concerns - Reusability - Maintainability - Centralized HTTP engine - Token abstraction - Scalable structure - CI/CD ready

## 8. Complete Flow Summary

Feature → Step Definition → API Class → ApiMethods Engine → Supertest → API Server → Response Validation → Report Generation