| **Criteria** | **Rest Assured** | **Postman** |
| --- | --- | --- |
| ✅ **Test Development** | Fully programmatic, reusable code | GUI for creating requests; JavaScript-based test scripts |
| 🔁 **Reusability** | High – functions, classes, and packages | Moderate – environments & scripts help reuse |
| 🔍 **Assertions** | Java-based (using Hamcrest, TestNG, JUnit) | JavaScript-based (Chai-style) |
| 📂 **Test Data Handling** | External JSON, Excel, or DB supported easily | Supports CSV/JSON (limited handling via Collection Runner) |
| 📄 **Payload Flexibility** | Full control – generate dynamic JSON programmatically | Mostly static – dynamic scripting is harder |
| ⚙️ **Environment Config** | Java config files or Maven profiles | Environment variables in collections |
| 📦 **Version Control** | Code-based – ideal for Git | JSON-based collections – possible, but not ideal for merge conflicts |
| 🔧 **CI/CD Integration** | Seamless via Maven/Gradle + Jenkins | Via Newman CLI |
| 📈 **Reporting** | Integrates with Allure, Extent, etc. | Basic CLI reports, HTML via Newman |
| 👨‍💻 **Learning Curve** | Steep for non-programmers | Easy UI, low entry barrier |
| 🔐 **Authentication Handling** | Full programmatic control (OAuth, Bearer, etc.) | Predefined Auth types + scripting |
| 🤝 **API Chaining / State Sharing** | Full control using Java objects | Global/Environment variables support chaining |
| 🧪 **Parallel Execution** | TestNG/JUnit supports it well | Not supported directly in GUI; possible via Newman CLI |
| 📤 **Mocking / Testing with Delays** | Needs external libraries | Built-in mock server support (Postman Pro) |

| **Feature** | **Allure Report** | **Extent Report** |
| --- | --- | --- |
| **Cucumber Support** | ✅ Native support for feature files, step definitions | Requires integration work |
| **Multi-env execution view** | ✅ Shows history and test trends | ❌ Basic visualizations |
| **Customization** | High (attachments, logs, screenshots, DB dumps) | Moderate |
| **CI/CD Integration** | ✅ Jenkins, GitLab, GitHub Actions supported | ✅ Jenkins (via plugins) |
| **Visuals** | Modern, interactive, and elegant | Good but less modern |
| **Parallel Test Execution Reporting** | ✅ Excellent | ❌ Can be flaky |
| **Team Visibility** | ✅ Shareable, self-hosted reports possible | ❌ Local only unless manually shared |
| **Learning Curve** | Moderate | Easy |

## 1.Basic CRUD Operations

|  |  |
| --- | --- |
| Test Case | Description |
| GET – Retrieve Resource | Validate status code, response time, and response body |
| POST – Create Resource | Validate resource creation, status 201, ID returned |
| PUT – Update Resource | Validate full update, status 200, content updated |
| PATCH – Partial Update | Validate only specified fields change |
| DELETE – Remove Resource | Validate deletion status 204, resource no longer available |
| OPTIONS – CORS / Preflight | Validate allowed methods and headers |

## 2.Positive Test Cases

|  |  |
| --- | --- |
| Test Case | Description |
| Valid Payload & Headers | Valid input should return 2xx response |
| Resource Creation | Assert new ID, timestamps, and data accuracy |
| List API | Assert pagination, total count, and content list |
| File Upload | Validate file is accepted and stored correctly |
| Chained API Call | Response from one API used in another (e.g., create → fetch) |
|  |  |

## 3.Negative Test Cases

|  |  |
| --- | --- |
| Test Case | Description |
| Missing Required Field | Should return 400 Bad Request with proper error message |
| Invalid Field Format | Test email/UUID/phone validations |
| Unauthorized Access | Access with no token or expired token returns 401 |
| Forbidden Access | User without required role returns 403 |
| Resource Not Found | Access non-existent ID returns 404 |
| Method Not Allowed | POST on read-only endpoint returns 405 |
| Invalid JSON | Malformed request body returns 400 |
| Invalid Headers | Missing/invalid Content-Type or Accept headers |

## 4. Authentication & Authorization

|  |  |
| --- | --- |
| Test Case | Description |
| No Auth Header | Should return 401 |
| Invalid Token | Should return 401 Unauthorized |
| Expired Token | Should return 401 or 403 |
| Role-based Access | Admin vs User access on restricted endpoints |
| Refresh Token Flow | Validate token renewal works correctly |

## 5. Schema & Contract Validation

|  |  |
| --- | --- |
| Test Case | Description |
| Response Schema Validation | Validate API response matches expected JSON schema |
| Data Type Validation | Field data types are as per API contract |
| Mandatory Fields | Ensure mandatory fields are never null or missing |

## 6. Performance Test Cases (Basic)

|  |  |
| --- | --- |
| Test Case | Description |
| Response Time | Ensure response time is < 1s (or defined SLA) |
| Payload Size | Validate large payloads are handled correctly |
| Rate Limit | Hitting API too frequently should return 429 Too Many Requests |

## 7. Data-Driven Test Cases

|  |  |
| --- | --- |
| Test Case | Description |
| Multiple Payloads | Test same API with different payloads (from DB/JSON) |
| Parametrized Inputs | Different combinations of query/path/body parameters |

## 8. Header & Cookie Validation

|  |  |
| --- | --- |
| Test Case | Description |
| Correct Headers Returned | Check Content-Type, Cache-Control, etc. |
| Set-Cookie Validation | Validate cookies set in login/session APIs |
| Language/Locale Headers | Accept-Language returns proper translations |

## 9. Query & Path Parameter Testing

|  |  |
| --- | --- |
| Test Case | Description |
| Pagination Params | Check /items?page=2&size=10 returns correct result |
| Sorting | /items?sort=name,asc |
| Filtering | /users?status=active&role=admin |
| Invalid Param Value | Returns proper validation error |

## 10. Edge & Boundary Testing

|  |  |
| --- | --- |
| Test Case | Description |
| Very Large Input | POST/PUT with huge payloads |
| Empty Input | Empty strings, arrays, or null fields |
| Max Length | Send data at maximum allowed lengths |
| SQL Injection / XSS | Try injecting SQL/XSS to check vulnerability handling |

## 11. Integration & Sequence Testing

|  |  |
| --- | --- |
| Test Case | Description |
| API Flow | Create user → Add to group → Assign permissions |
| Cleanup Logic | Delete or revert created data post-test |
| DB Validation | Confirm DB is updated correctly after API hits |