Software Testing

Made by : Hana Ahmed Nabhan

Introduction Software Testing

Definition:

- Software testing is the process of evaluating a system or its components to determine whether it meets specified requirements.

Purpose:

1. Verify that the software performs as expected under various conditions.
2. Identify defects to ensure quality and reliability.

Fundamentals of Testing

- **Defects, Errors, and Failures**

Error: A mistake made by a developer, often due to misunderstanding or oversight.

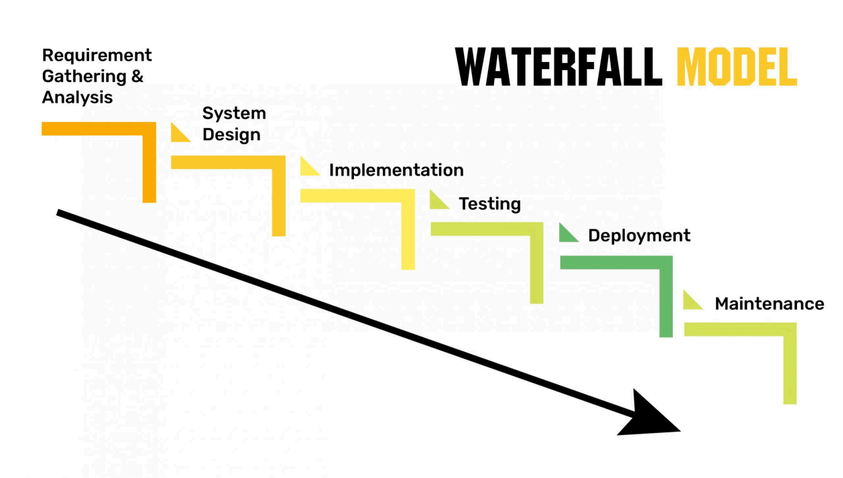
Defect (Bug): A flaw in the software resulting from an error, causing deviation from expected behavior.

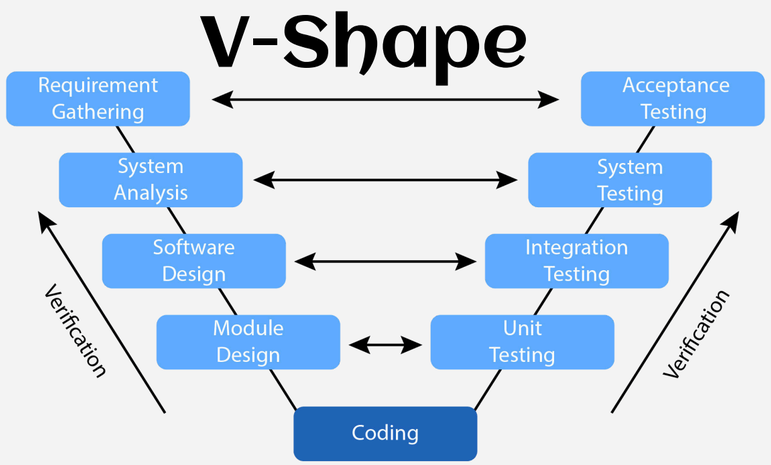
Failure: The consequence of a defect when the software does not perform as intended during execution.

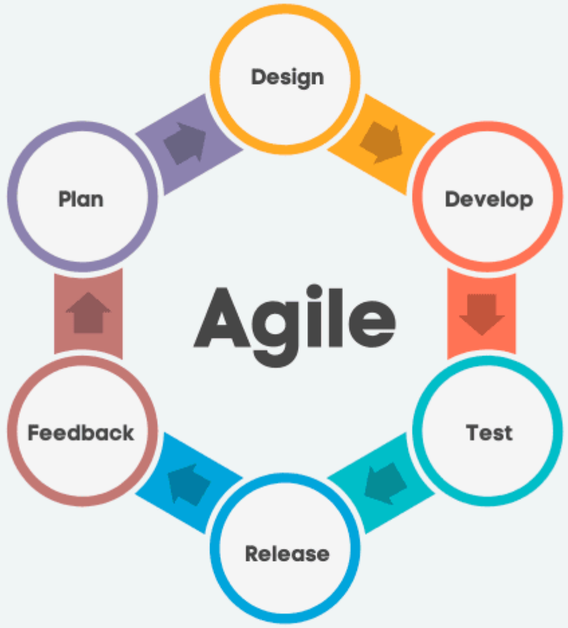
- **Impact:**Errors lead to defects, which, if undetected, cause failures. Failures can result in poor user experience, increased costs, and potential system harm.

Software development lifecycle (SDLC)

structured process that guides the development of software through phases like planning, design, coding, testing, and deployment.







Seven Testing Principles

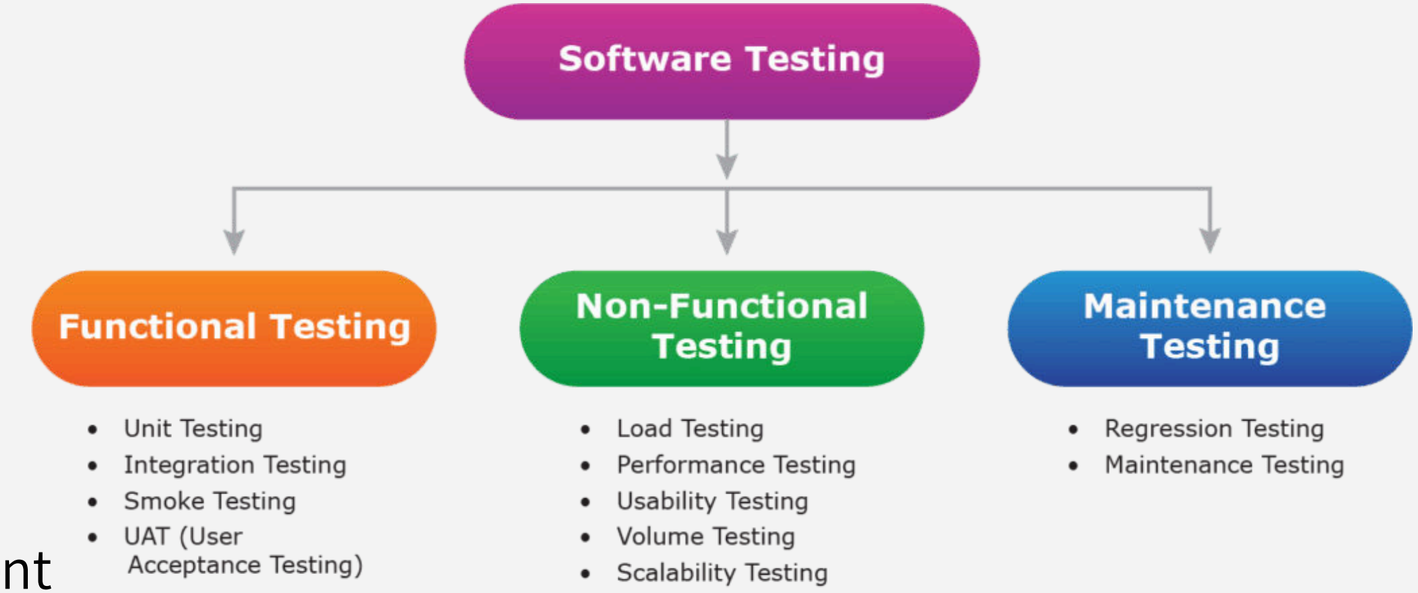
Testing Shows the Presence of Defects: Testing can find defects but cannot prove that the software is defect-free.
Exhaustive Testing is Impossible: It’s impractical to test everything, so focus on the most critical areas.
Early Testing: Start testing early in the development process to catch defects sooner.
Defect Clustering: Most defects are found in a small number of modules, so prioritize those areas.
Pesticide Paradox: Regularly update test cases to continue finding new defects.
Testing is Context-Dependent: Adapt testing strategies based on the project’s needs and risks.
Absence-of-Errors Fallacy: A defect-free product does not guarantee it meets all user needs.

Fields of Software Testing

- **Functional Testing:** Validates the software against functional requirements.

Levels of Testing:

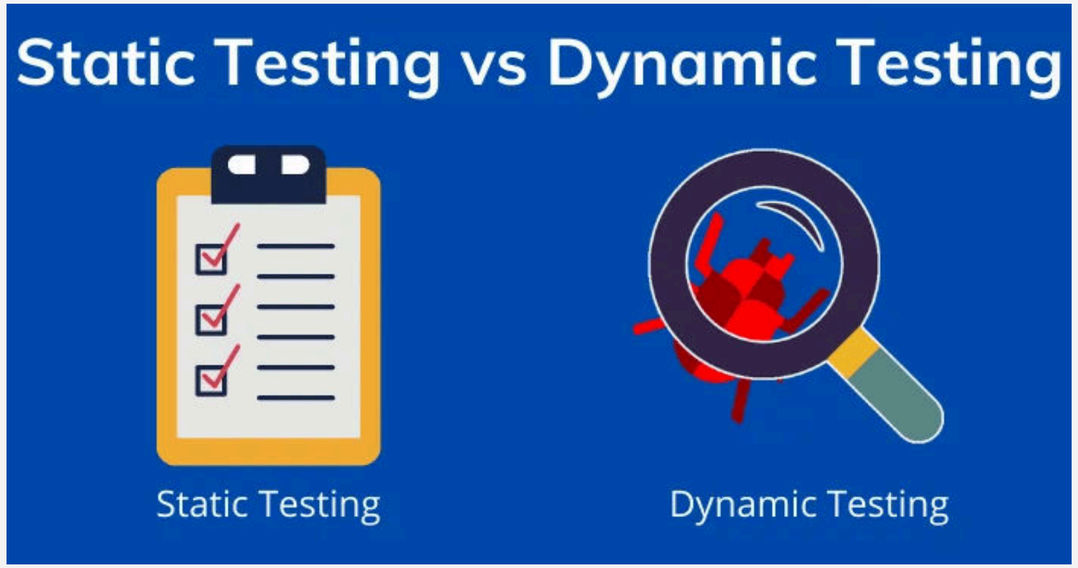
- Unit Testing: Testing individual components or functions.
- Integration Testing: Ensuring different components work together.
- System Testing: Testing the complete system as a whole.
- Acceptance Testing: Verifying the system meets business requirements.



- **Non-functional Testing**:Evaluates attributes like performance, usability, and security.

Load Testing, Stress Testing, Usability Testing, Security Testing.

Types of Software Testing



- **Static Testing:**

known as Verification testing that is performed to check the defects in software without actually executing the code of the software application.

it is performed in the early stage of development to avoid errors as it is easier to find sources of failures and it can be fixed easily.

- **Dynamic Testing:**

is a type of Software Testing that is performed to analyze the dynamic behavior of the code. It includes the testing of the software for the input values and output values that are analyzed.

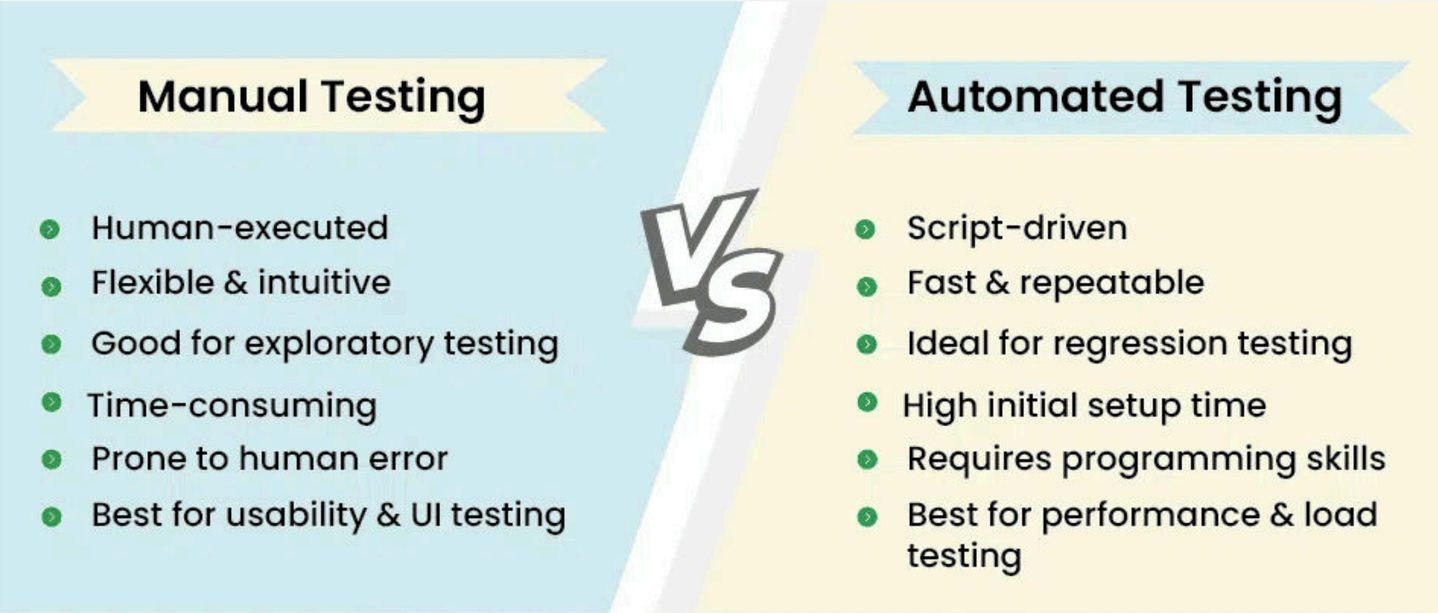
The purpose of dynamic testing is to confirm that the software product works in conformance with the business requirements.

Difference Between Manual and Automated Testing

Definition :

Manual testing : the test cases are executed by the human tester.

Automation Testing: the test cases are executed by the software tools.



API TESTING WITH POSTMAN

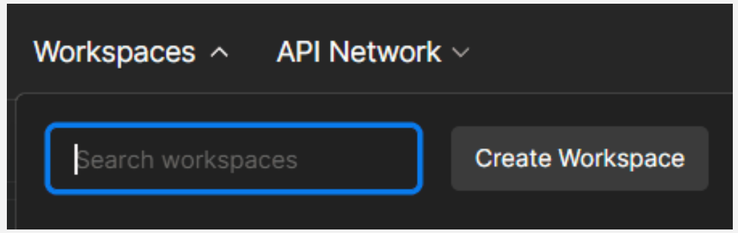
API Testing involves validating the functionality, performance, and security of Application Programming Interfaces (APIs).
it is used for both manual and automated testing , but it is used more for manual testing

What is an API Request?

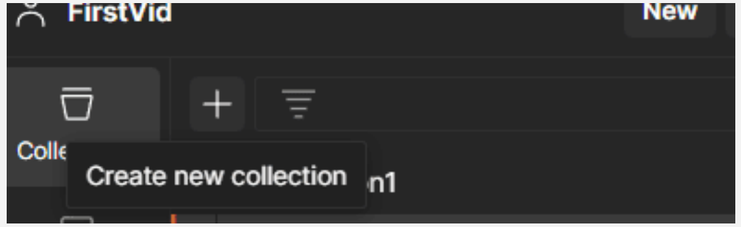
Definition: An API request is a message sent from a client (e.g., Postman or a RestAssured script) to a server to request data or perform an action.

STEPS:

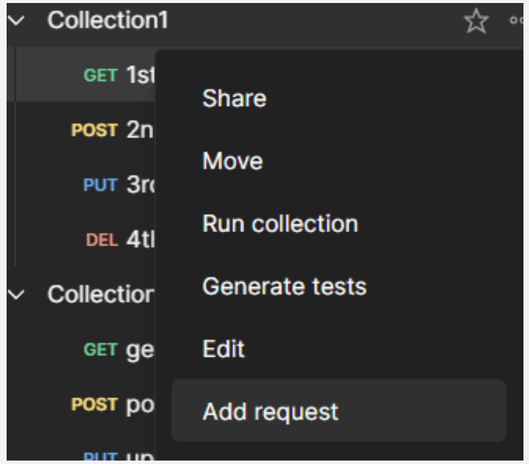
- Create a New Workspace.



- Create a New Collection.



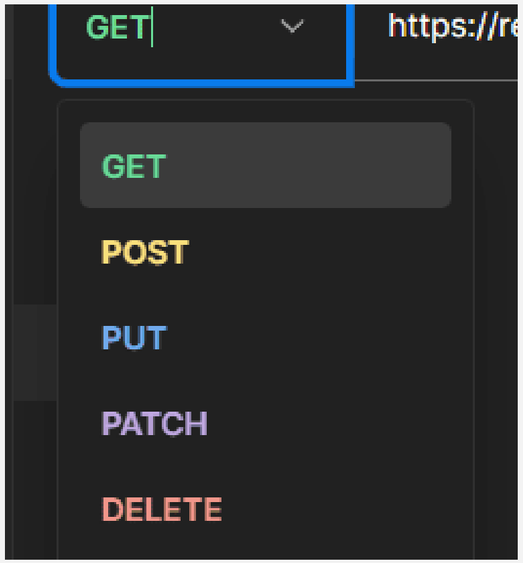
- Add a Request



API TESTING WITH POSTMAN

- Setup request

HTTP Method: Specifies the action (e.g., GET, POST, PUT, DELETE).
Endpoint: The URL where the request is sent.
Headers: Additional information (e.g., authentication tokens, content type).
Body: Data sent with the request (used in methods like POST or PUT).
Common HTTP Methods:
GET: Retrieve data from the server.
POST: Send data to the server to create something new. it needs a body
PUT - PATCH: Update existing data on the server. it needs a body
DELETE: Remove data from the server.



HTTP STATUS CODE

HTTP Status Codes

Level 200

200: OK
201: Created
202: Accepted
203: Non-Authoritative Information
204: No content

Level 400

400: Bad Request
401: Unauthorized
403: Forbidden
404: Not Found
409: Conflict

Level 500

500: Internal Server Error
501: Not Implemented
502: Bad Gateway
503: Service Unavailable
504: Gateway Timeout
599: Network Timeout

API TESTING WITH POSTMAN

- Add Tests :

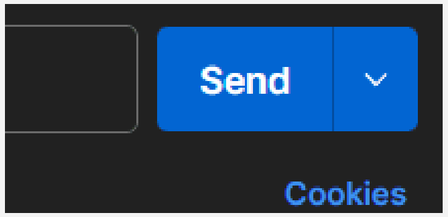
Params Authorization Headers (6) Body Scripts Settings

Pre-request

Post-response

```
//status code
pm.test("Status code is 200", function () {
    pm.response.to.have.status(200);
});
pm.test("success",()=>{
pm.expect(pm.response.code).to.be.oneOf([200,201])
});
//headers
pm.test("Status code is 200", function () {
    pm.response.to.have.header("Content-Type");
});
pm.test("Content-Type is UTF-8", function () {

    pm.expect( pm.response.headers.get('Content-Type')).to.eql('application/json');
});
```

- Run

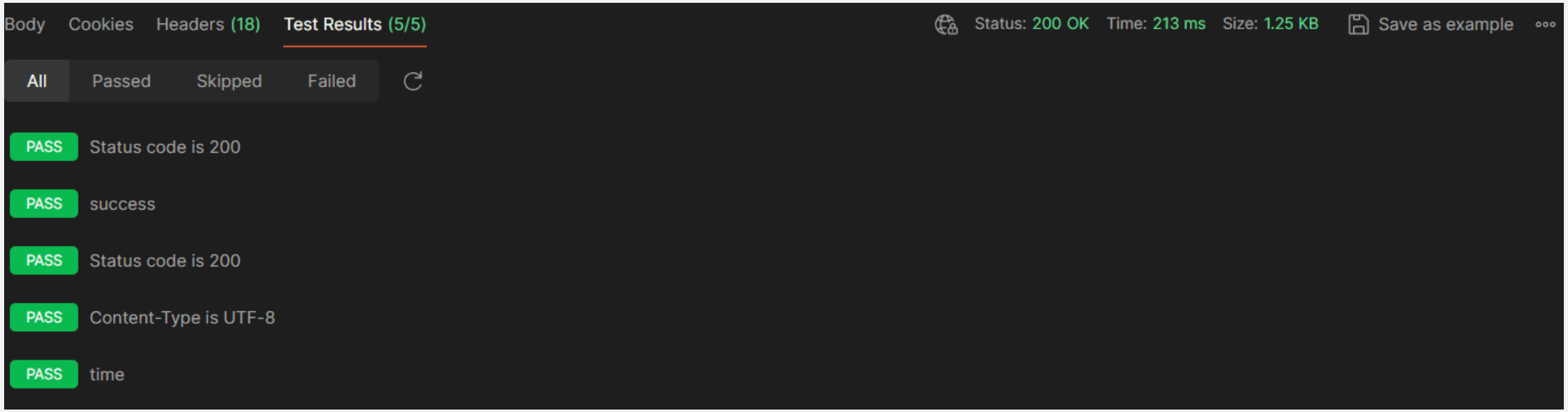


API TESTING WITH POSTMAN

- Test Results :

Status code

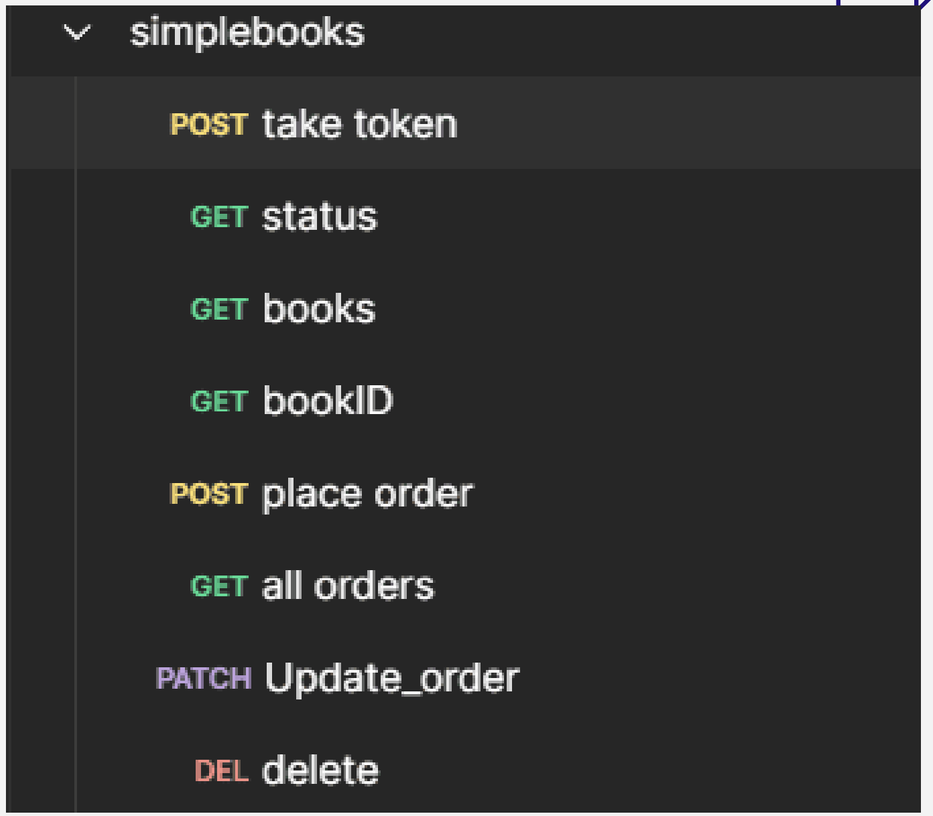
Time taken



project using postman

Simple Books API Testing Workflow in Postman

1. **Get Token (POST):** Obtain an authorization token(digital key that allows a client to access specific resources).
2. **Check Status (GET)**: Verify API status.
3. **Get All Books (GET):** List all available books.Get
4. **Book by ID (GET):** Retrieve details of a specific book.
5. **Place Order (POST):** Create a new book order.
6. **Get All Orders (GET)**: View all orders placed.
7. **Update Order (PATCH):** Modify an existing order.
8. **Delete Order (DELETE):** Remove an order by ID.



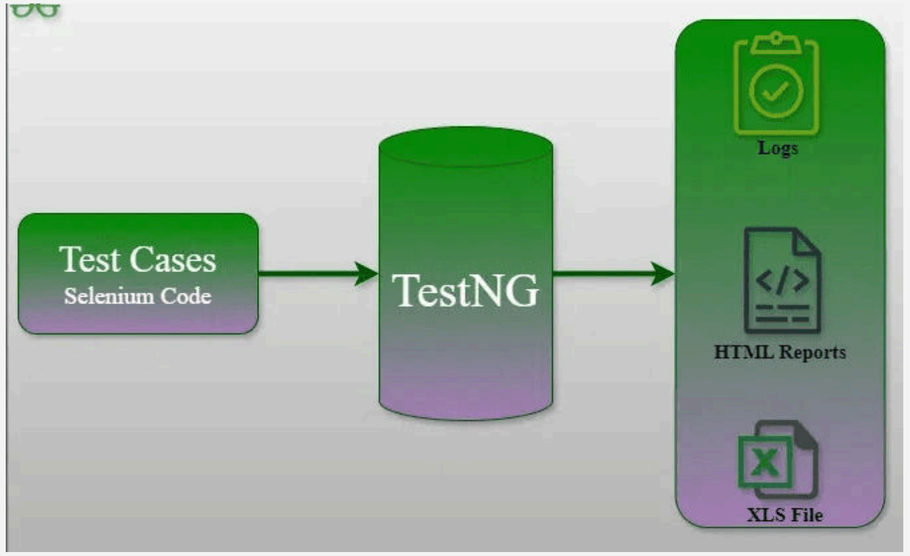
Automation

RestAssured

is a Java-based library used for testing RESTful web services(Representational State Transfer). It simplifies the process of making HTTP requests and validating responses, making it easier to automate API testing.

TestNG

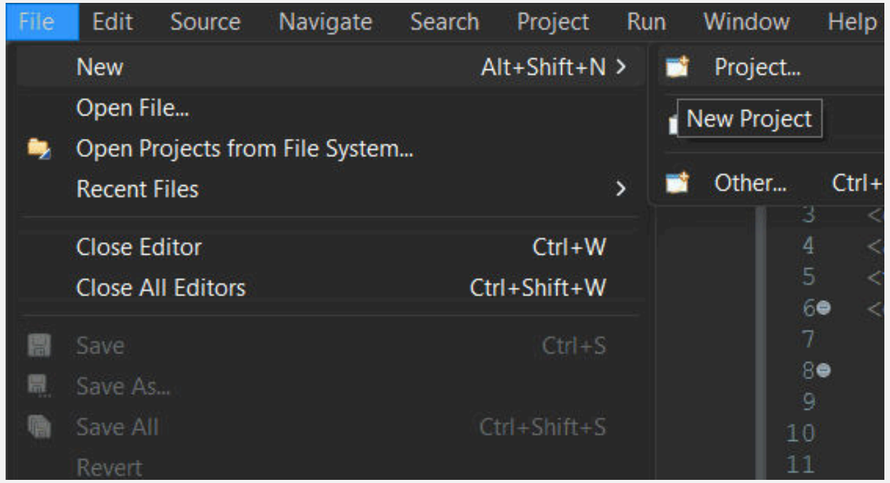
is a testing framework that provides a powerful and flexible way to create and manage automated test cases.
it has special Annotations that can be used to provide easy way for Testing



Automation

STEPS:

- Create a new Maven project in Eclipse.



- Add the required dependencies for RestAssured and TestNG to the "pom.xml"file

```
<!-- https://mvnrepository.com/artifact/
<dependency>
<groupId>io.rest-assured</groupId>
<artifactId>rest-assured</artifactId>
<version>5.5.0</version>
<scope>test</scope>
</dependency>
```

All dependencies can be found on MVN Repository

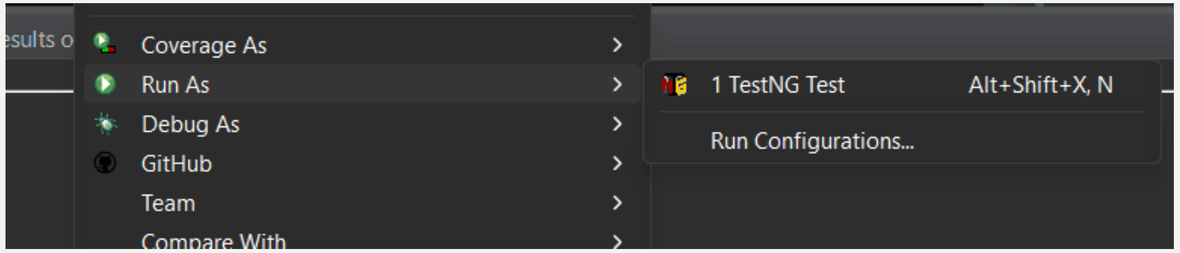
MVN REPOSITORY

Automation

- Create a Java class to write test cases.

```java
package firstproj;


import org.testng.annotations.Test;


Run All
public class httpRequest {
    int id_var;

    @Test (priority=1)
    Run | Debug
    void getusr() {
        given()
        .when()
            .get("https://reqres.in/api/users?page=2")
        .then()
            .statusCode(200)
            .body("page",equalTo(2))
            .log().all();

    }
```

- Run test cases using TestNG



Automation

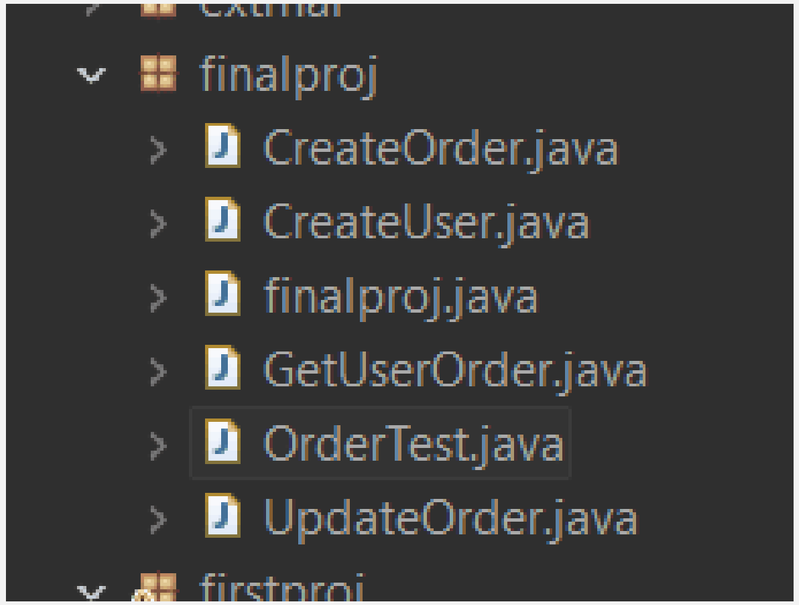
- Results

```
<terminated> httpRequest [TestNG] C:\Program Files\Java\jdk-20\bin\javaw.exe  (Aug

===============================================
Default suite
Total tests run: 4, Passes: 4, Failures: 0, Skips: 0
===============================================
```

the build can be automated and integrated with a Continuous Integration (CI) tool like jenkins.

PROJECT USING AUTOMATION

- **Base Class (finalproj)**:Common setup (Faker instance, base URI).
- **Create User Class (CreateUser) :** Generate user tokens.
- **Create Order Class (CreateOrder):** Create orders.
- **Update Order Class (UpdateOrder):** Update orders.
- **Get User Order Class (GetUserOrder):** Retrieve order details.
- **Test Class (OrderTest):** Execute end-to-end test (testing an entire application flow from start to finish).



CONCLUSION

During my software testing internship, I gained a solid foundation in testing principles, including static and dynamic testing as covered in the ISTQB Foundation Level. Additionally, I learned how to work with APIs, creating and validating requests using tools like Postman and RestAssured. This experience has deepened my understanding of testing methodologies and equipped me with practical skills that are essential for a career in software testing. I'm eager to apply these skills in future projects and continue advancing in this field.

Thanks!

ANY QUESTIONS?