

**Overview of testing team core capabilities**

**Contents**

[1. UI Automation Framework – Key Features and Infrastructure Details 3](#_Toc108532321)

[2. Implementation and Integration of APIs into Framework 4](#_Toc108532322)

[3. Release Pipeline 4](#_Toc108532323)

[4. Automation Workflow 4](#_Toc108532324)

[5. Manual Workflow 5](#_Toc108532326)

[6. Performance testing Workflow 5](#_Toc108532327)

[7. Introduction of new age automation tool 6](#_Toc108532328)

[8. QA Center of Excellence (COE) 6](#_Toc108532329)

# 1. UI Automation Framework – Key Features and Infrastructure Details

* **WebUITest** class facilitates the execution of test scripts developed in **JAVA** on different environments without any code change. It provides flexibility to add various parameters namely URL, username, password into **VM** (virtual machine) arguments, thereby ensuring portability
* The framework implements the Page Object Model(**POM**)design patternto enable binding multiple classes with same module grouped under one package to reduce complexity
* There is a provision for utilizing multiple **TestNG.xml** files which allows grouping of the test scripts by functionality and release version
* To make the test data setup easier and user friendly, the test data can be passed through various files including
  + .sql
  + .json
  + .xlsx depending on user need
* Framework has a wrapper class to perform various Selenium actions (**like click, waiting for a page to load, capturing screenshot, file interaction etc.)** which takes into consideration the process of page synchronization
* A utility class performs various operations like getting the state of Document Object Model (**DOM**) for better page synchronization, analyzing the exception caught by the test-script, reading a file. Apart from that a **JavaScript Executor** performs operations such as clicking and scrolling in the web page
* **Extent reports** are used over built-in reports that are generated through Junit and TestNG as it offers several advantages such as a pie chart representation, test stepwise report generation, adding screenshots etc., at every test step
* Internal java logging allows to configure the logging parameters. The logging framework can display different levels of logs like **INFO, ERROR** and **WARN**

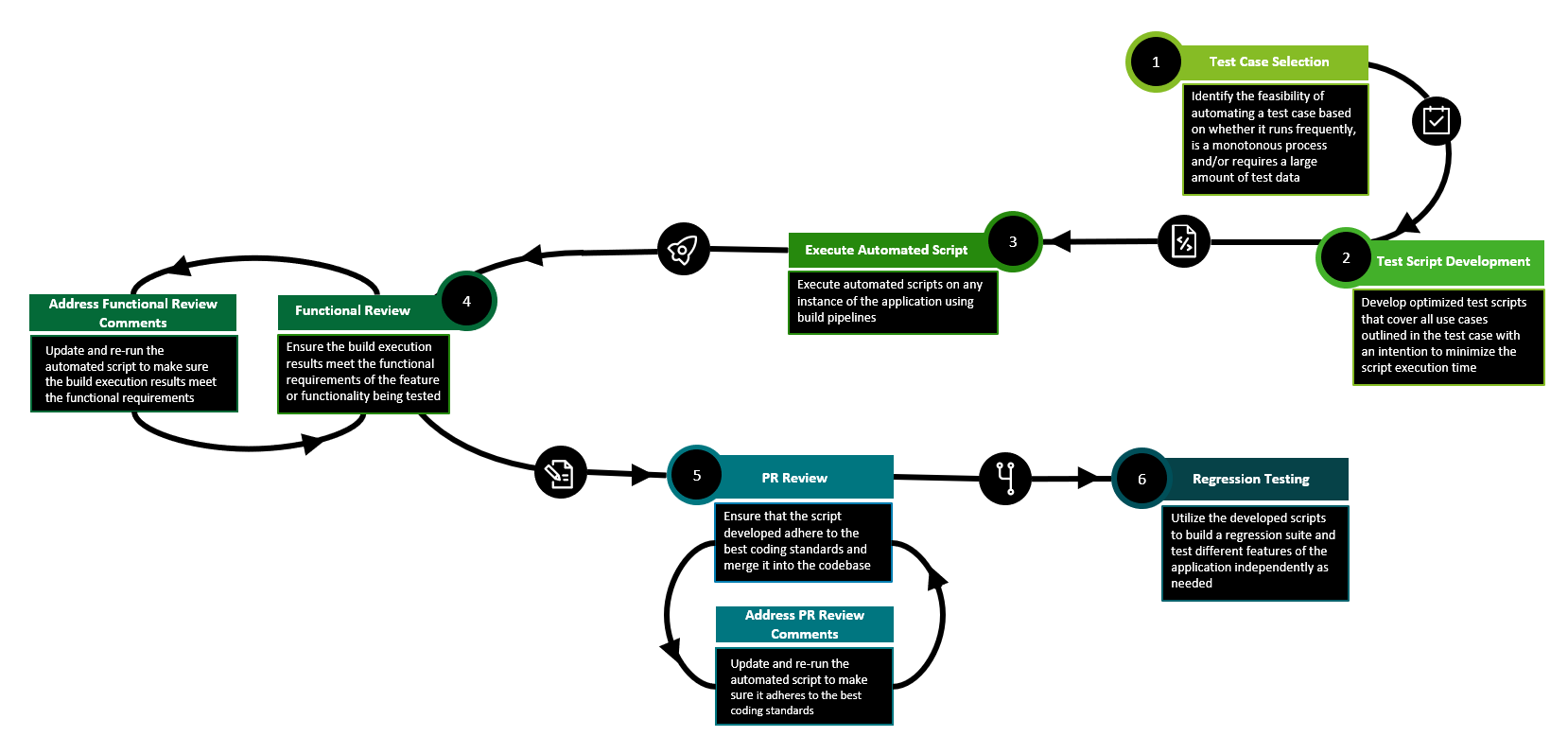
# 2. Implementation and Integration of APIs into Framework

* An integrated **REST API** (Application Program Interface)framework allows us to reduce the execution time and remove the complexity involved in user interface (**UI**) automation.
* An integrated **Hibernate** framework that allows ease of database interaction.
* A self-developed **Flask WSGI** framework with **Python** allows us to create APIs for most used features in the application to avoid direct UI intervention

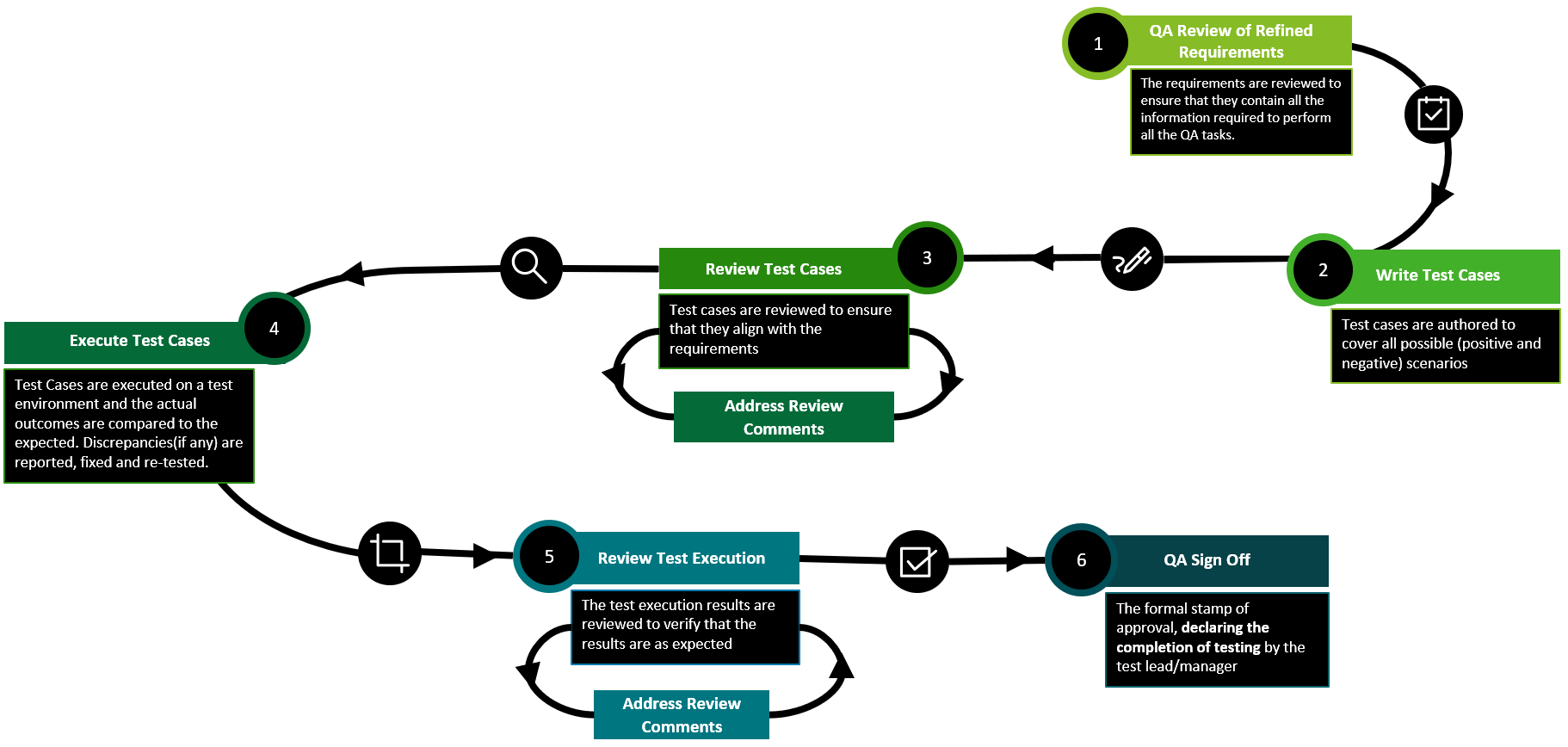
**3. Release Pipeline**

Multiple pipelines created using **CI/CD feature** of **Azure DevOps** areintegrated into the framework with the help of **.yml** files. These pipelines are used to execute test scripts on any environment as required, by updating pipeline variables. One of the pipelines is coupled to the code repository in a way that a build gets triggered as soon as a new pull request (**PR**) is created.

**4. Automation Workflow**

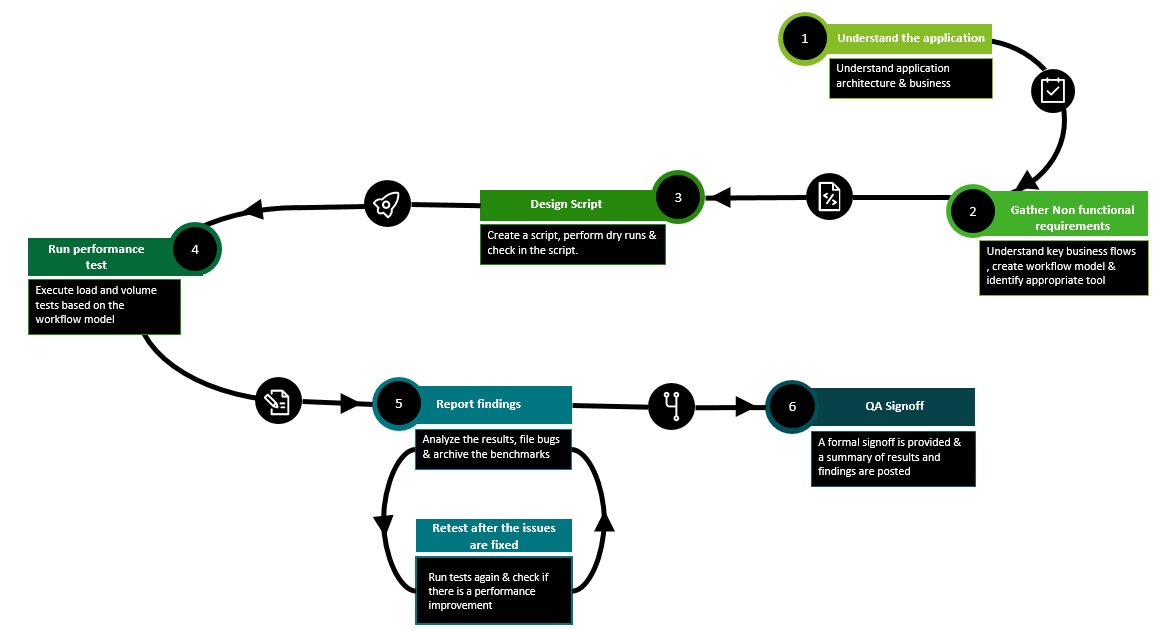
****

**5. Manual Workflow**



**6. Performance testing Workflow**

Performance testing is a key nonfunctional testing conducted by our team to identify performance bottlenecks and benchmark the system performance. Our arsenal is equipped with multiple performance testing & APM tools like JMeter, Performance center, Grafana, perfmon, etc.



**7. Introduction of new age automation tool**

We constantly strive to incorporate the latest tech stack in our project. We frequently work on research projects and POC’s in automation, performance testing tools and process improvement initiatives.

In one of our recent additions, we have introduced Playwright UI automation tool in one of the products. We have also worked on designing a robust framework to facilitate just in time script development and flakiness reduction.

**8. QA Center of Excellence (COE)**

QA COE community was established with the intent of forming a single known platform across Hybrid-Operate to apply thoroughly engineered and designed processes for Quality Assurance engagements. The community has been operating since 3+ years now and has expanded to incorporate more than 20 tech assets.

The focus of the core COE team has been to explore standard operating procedures (SOPs) and implement QA best practices across product teams, gather upskilling needs across the QA job family through anonymous surveys, conduct training sessions and certifications and speed up recruitment processes by formalizing a panel for campus and lateral recruitment in specialized areas of testing.  
  
To improve communication, QA COE has launched periodic newsletters to showcase the Hybrid Operate QA highlights, upcoming trainings, innovations & industry best practices. Quarterly All Hands-Meet (AHMs) are organized and leveraged as a platform to connect internally and understand what each tech asset team has to offer in terms of technology, people, and innovation.

The Innovation wing of the COE community works on research projects and proof of concepts (POCs) which are published in the newsletter and organizes internal “Tech Talk” sessions to share best practices and approaches internally within the tech asset teams