Test Automation Process

Agenda

- Challenges in manual testing & Need For Test Automation
- Scope & priority of Test Automation – jConnect & jConfigure Application
- Repeated manual test activities Vs Test effort
- Approach, Tools & Technologies planned
- Advantages of Cucumber over other automation tools
- POC for Automation - Approach and Plan
- POC – Plan & Schedule
- Standards Expected/planned for Automation
- Plan for Detailed test preparation and Execution

Current Challenges & Need for Automation

Frequent Deployments & repeated Stability check

Privilege based testing for smoke & workflow

Shorter release cycles & repeated checking across environments

Multi-browser or multi-platform test support

Resource dependencies & manual errors during quick support

Huge effort for complete manual regression

Scope & priority of Test Automation

Smoke Test –Stability check based on different user roles

End to End Workflow for different user roles– Workflow & integration between different modules

Multi browser support – For identified key features

Integration Test – Integration between jConnect & external systems

Regression Test

Repeated manual test activities Vs Test effort

Type of Testing	Effort Required (Appx. Hrs)	Current Manual Test Coverage	Environments Supported
Smoke	4	4 Different Roles under site users & Subject to ensure stability of the application based on privileges configured	QA, UAT, PRODUCTION
Workflow	8	2 roles Covering all key workflow between jConnect , jConfigure & Subject applications	QA, UAT
Integration	4	With CRIO	QA, UAT
Regression	160	Functionality of 42 key modules related to jConfigure & jConnect	QA
Mobile/Multi Browser Support	4	1 role & 1 platform/Browser	QA

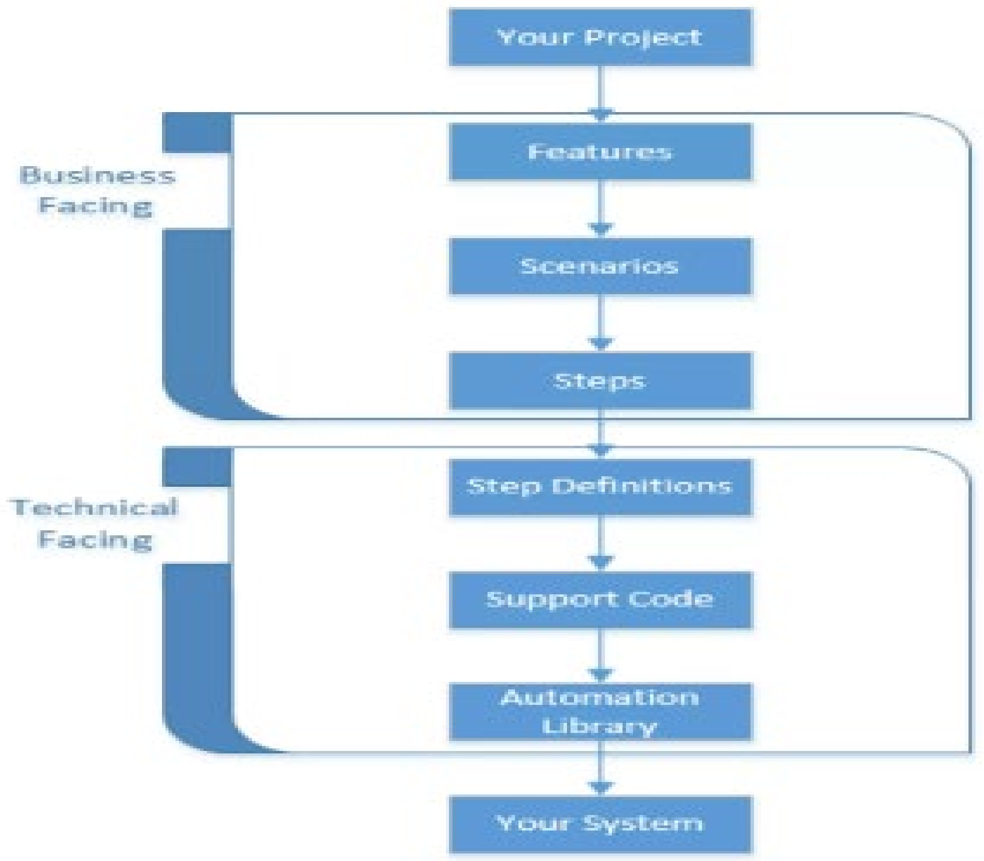
Targeted Automation Coverage

(75 – 80%)

Approach, Tools & Technologies planned

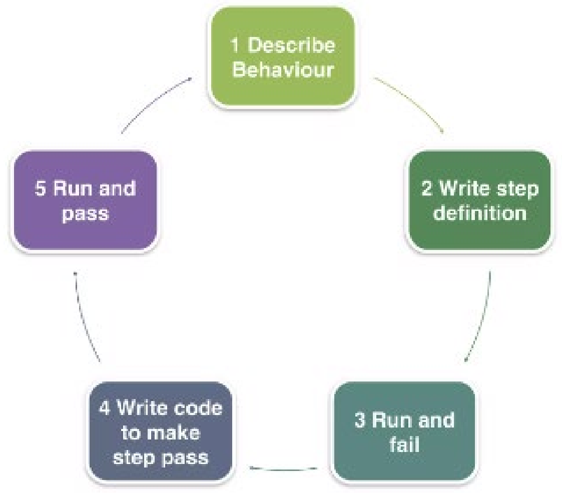
- Framework Approach - BDD (Behavioural Driven Development)
- Test Automation Tool & Framework – Cucumber, Selenium (Open Source Tool)
- Integrated Development Environment (IDE) - Intellij Idea
- Build Tool – Maven with Project Object Model (pom.xml) implementation
- Plugins & Extensions planned - Sonarlint (For Code Quality Checking),
- Extent Reports, Cucumber Reports, Spark Reports (For custom reporting)
- Source & Configuration Control – Integration with GitHub and create Branching Strategies
- Automatic build & execution –Jenkins for CI/CD & scheduled execution

Framework, Tool Design



Cucumber

- Cucumber is a high-level testing framework that supports behavior driven development.
- It runs automated acceptance tests on web applications.
- Cucumber is a tool that executes plain-text functional descriptions as automated tests. The language that Cucumber understands is called Gherkin.



Cucumber

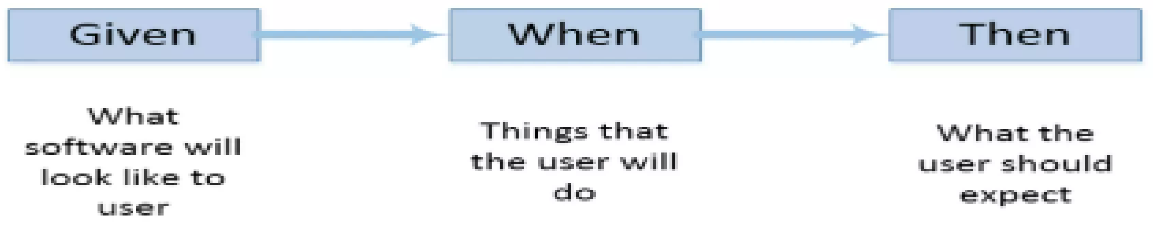
FEATURE FILE and CUCUMBER

Cucumber Nomenclature

- **Feature:** Single file, ideally describing a singl
  feature
- **Scenario:** A test case
- **Given-When-Then:** Test Preconditions, Execution and Postconditions
- **And:** Additional test constructs

Feature Introduction

Every .feature file conventionally consists of a single feature. A line starting with the keyword Feature followed by free indented text starts a feature. A feature usually contains a list of scenarios. scenarios together independent of your file and directory structure.


Given
When
Then
What software will look like to user
Things that the user will do
What the user should expect

Given The login page is opening
When I input username, password and click Login button
Then I am on the Home page

Advantages of Cucumber over other automation tools

- A well supported tool for agile methodology
- Reduces the communication gap between technical and non-technical teams and stakeholders
- Easy collaboration between Product owner, QA Engineers, Developers and Business Owners due to the common language support
- Flexibility to involve even non-technical members for script preparation with minimal training
- Simplified coding due to the support for laymen text language for script preparation
- Enhanced flexibility & Reusability of code
- Ease of maintenance and scalability
- Support for Integration with technologies like Java, python, Jython and industry standard tools, apis & plugins like Selenium, external reporting tools

POC Automation

Plan	• Identify the key scope for POC
Prepare	• Identify the steps for automation from manual test script
Design	• Prepare automation test script for the identified test steps – BDD methodology using the framework developed
Execute	• Execute the test script and identify its ability to report the gaps
Analyze	• Prepare a report based on test coverage, time taken for execution and defects identified.
Compare	• Compare & confirm the improvements noted between manual & automation execution.
Correct & stabilize	• Correct & stabilize test script with collective feedback
Update/enhance	• Update/enhance the script to suite process requirements

POC - Plan & Schedule

Automation Activity	Responsibility	Schedule (Tentative) Start Date	Schedule (Tentative) End Date	Status
Test automation Process & Approach	Management team, Raji & Pramoth	30-Aug-22	11-Nov-22	In Progress
Finalize the test tool for Automation	Raji, Pramoth	04-Nov-22	04-Nov-22	Completed
Finalize scope & priority of Test Automation	Management team, Raji & Pramoth	07-Nov-22	11-Nov-22	In Progress
POC on Automation – planning & Review	Management team, Raji & Pramoth	07-Nov-22	11-Nov-22	In Progress
Test preparation & Execution for POC Scope	Pramoth	07-Nov-22	14-Nov-22	In Progress
Internal presentation on POC scope identified	Pramoth	15-Nov-22	15-Nov-22	
Presentation to management team on the POC scope identified	Pramoth	17-Nov-22	17-Nov-22	

Standards Expected/planned for Automation

Stable and dedicated environment & Set up

Uniformity in the implementation of controls across application

Standard test data for repeated testing

Planning for full-fledged automation

- Prioritise the automation need
- Analyse manual test scripts & categorize the steps that are feasible for automation
- Identify the limitations/workaround to be followed for automation based on POC
- Get the % of automation coverage in the identified test scope
- Identify and prepare possible dedicated & reusable test data in the test environments and name them appropriately
- Derive the test schedule for test script preparation
- Repeat the Steps given in the POC section for each identified phase
- Maintain all the artefacts in a repository (GitHub) for reference
- Schedule and do a daily execution of prepared test scripts via JENKINS and keep enhancing the scripts. Adding scenarios from coverage and have a comprehensive automation test pack for Regression and Smoke Tests