02. Automation Principles and Rules

Last updated by | Warren Duffin | Sep 21, 2022 at 9:15 PM GMT+5:30

| Introduction
This will be outlining the rules that are to be peer enforced and principles that should be used as individual guidance, this will improve but by no means limited too creating constant approach across teams. Please be aware that this document is subject to change as we learn from the ongoing project.

Principles – should be in every individuals mind when conducting automation .

Principles	Description
1: Follow BDD guidelines	It should be assumed that all BDD recommended guidelines are to be followed unless stated otherwise.
2: Notify the right people at the right time	If your one of the lucky few to get nightly notification regarding automated runs. If you notice a failure is detected please don't assume someone else is handling, ask in the automation channel.
3 : Tests should rely on as little pre-Defined data in the database as possible	This will keep manual setup to a minimum. i.e. This can be achieved by Deriving data from codes.
4: D.R.Y. (Do not Repeat yourself)	Utilize and add to the frameworks existing shared methods.
5: K.I.S.S (keep it simple, stupid)	don't over complicate the solution
6: Fail Fast	Tests should fail as fast or as close to the cause as possible aids faster debugging.
7: Y.A.G.N.I (you aren't gonna need it)	Don't add any unnecessary/unneeded functionality.
8: IDs should be used for element locators	IDs should be used when defining elements if they don't exist in the system under test, they should be added.

Rules – Explicit regulations

Rules	Description
1: Test data must be dynamically generated	Where test data can me dynamically generated you should do so e.g. test: Adding Company Personnel to Site, givens should add the company personnel in reference before adding to the site.
2: New Prerequisite data steps must be documented in the wiki	This is to stop duplication in the project.
3: Feature files must be reviewed before development	Review for feature files must be completed before coding the back end to limit rework and must include at least 1 BA. This is to ensure that feature files are readable from a technical perspective as well as a business thus creating living documentation.
4: Project must conform to IMPACT coding standards	Please refer to the IMPACT coding standard classification for any rules associated. 05. Coding Standards
5: Already established Naming conventions should be followed	I.e. JSON, IWP page object and feature files
6: Test numbering should follow the standard	10. Functional Module ID's (api pending)
7. Tags must conform to agreed standards	08. Tags and Reporting
8. capitalization in feature parameters	i.e. `When user "adds Subject" on the "Subject Overview" page` nouns in upper case adjectives in lower case and page parameter is a name so both capital