UiPath The Foundation of Innovation™

SDD Creation General Guidelines

Keep the SDD updated throughout the project lifecycle and involve stakeholders in the review and approval process.

- **Create code Repository.**

Topic	Recommendation	Example
Repository naming convention	{ProcessID}-{ProcessName}	coe101-process-vendor-invoice
Branch naming convention	Production code to be stored on Main branch	main
	UAT code to be stored on Develop branch	develop
	Feature branches: feature/{feature-name}	feature/dispatcher-create-report
	Bugfix branches: feature/{issueId-bugName}	bugfix/12-login-error
	Release branches: release/{version}	release/v1.0.12
Pull Request naming convention	[Version] {PR Description}	1.0.12 UAT fixes implemented
Commit message format	{StoryId}: {Description} If stories are not used, add a description of what is included in the commit.	ABC-123: Fixed login error by adding CheckAppState after inserting the credentials.
Approval guidelines	The PRs to Main must be approved by the SA. All tests are Passed.	
Folder naming convention	Create a sub-folder for each project included in the solution.	• root ○ COE101_Dispatcher ▪ Main.xaml ▪ project.json ▪ … ○ COE101_Performer ▪ Main.xaml ▪ project.json ▪ …

- **Document Creation:**
  - Use a customer template if provided and instructed by the CoE.
  - If no customer preference is known, use the default updated UiPath SDD Template.

- **Developer Collaboration:**
  - Conduct sessions with developers to explain solution design and address concerns.
  - Collaborate closely for a shared understanding of technical aspects.
- **Final SDD Review:**
  - Perform a final review and freeze the SDD; no further changes.
  - Share the frozen SDD with stakeholders during UAT boarding.
- **Framework Development:**
  - Don’t start with a framework in mind; build the framework around an optimal solution to the problem.
- **Naming Conventions:**
  - **Use Descriptive Words:** use words that clearly describe what a variable does. Avoid using abbreviations that might be confusing.
  - **Avoid Single Characters:** don't use single letters like w, a, s, d for variable names. Also, steer clear of names like index or temp.
  - **Avoid Misleading Names:** names should accurately represent what they stand for. Avoid names that could lead to confusion about the purpose of variables, arguments, or workflows.
  - **Descriptive Variable Names:** make sure names clearly express the purpose and function of a variable. Don't reuse the same variable for different logical steps.
  - **Consistency in Naming:** Stick to the decided naming convention consistently throughout your project.
  - **Align Variables and Arguments:** while only argument names are case-sensitive, for better readability, keep variables and other entities aligned with the same naming convention.
  - **Boolean Variables Clarity:** for boolean variables, use names that imply True or False. You can add prefixes like 'Is' or 'Has' (e.g., IsRed, HasRows). Always use positive names; avoid negative names like NotFound**.**
- **Workflow Analyzer Usage:**
  - Ensure developers run Workflow Analyzer periodically during development to meet required standards and decrease modification time during code review.
- **Asset Creation and Maintenance:**
  - Create assets for values that change in every environment.
  - Assign scopes/folders to assets for maintainability.
  - Establish a naming convention for multiple processes' assets.
  - Create per-robot value credentials/assets.
  - Select the correct datatype for storing values.
  - Avoid creating assets that remain static throughout the process run and in every environment.
  - Avoid creating too many assets; consider this on a case-by-case basis.