Category	Description	Story Level	Sprint Level	Release Level	Benefits	
Requirements	All Requirements documented and in DOORS	BP		X	- Ensures we have the documentation necessary to support SOX requirements. - Ensures we retain important decisions for future reference - Provides Production Support with critical information - Provides important information for future development / enhancement efforts	
	Detailed Tech specs will be completed for new functionality (that has never been developed before) as requested by the developer. These will be completed and reviewed before development begins. If design changes are introduced after a detailed Tech Spec has been completed, the Tech Spec should be marked as obsolete and the design changes impacting the Tech Spec summarized in Doors. At a minimum, data mapping from a user interface (including derived elements) to the data model must be completed	BP	TBD	TBD	- Provides data traceability for development and validation	
	UX Guidelines: 1. Hand-built pages – a. Reviewed by UX to ensure that the page matches the UX-produced model. - Or - b. Simple changes to existing pages (no model created), UX will be included in the drafting of the acceptance criteria. (I am not sure this needs to be included in the definition of done…) 2. Dynamically built pages a. Interim steps: Reviewed by UX to ensure that the pages follows closely the UX-produced model with UX approval of widget choice and order. b. Final Design: Reviewed by UX to ensure that the page matches the UX-produced model	BP			- Ensures a consistent customer experience	
	KPIs defined	x			- Provides the system performance levels we have committed to deliver	
Testing	Validation of story acceptance criteria using Chrome	x			- Establishes that new functionality is working as expected in the supported browser.	
	All defects on the story either needs to be fixed or decision made by product owner to add to the backlog	x			- Ensures that issues are addressed.	
	Regression testing completed successfully in the INT environment (using approved browsers, e.g.; IE10 and Chrome)		BP	x	- Verifies that new functionality does not have an impact on existing functionality. - Critical functionality is still operating as expected. - All previous code is functioning as expected in future and currently supported browsers.	
	Performance Single User Test completed in the DEV environment	x			- Allows for early insight into performance issues before promoting into higher environments.	
	Requirement traceability established (each requirement/user story is associated to a test case)	BP			- Provides detail on testing coverage, minimizes the risk of promotion without comprehensive testing. - Ensures all delivered requirements have been verified.	
	Manual Test scripts updated/created - for existing regression, new functionality and smokes as required	BP	x		- Keeps manual test cases up to date, allowing QA to more efficiently test in future releases.	
	Automated Test scripts updated/created - for existing regression, new functionality and smokes as required			BP	- Keeps automated test cases up to date, allowing QA to more efficiently test in future releases.	
	Performance Testing in cert for Load/stress - as deemed necessary in acceptance criteria (meet KPIs)			x	- Provides for a quality delivery standard on application performance. - Ensures customer performance expectations are met.	
	No severity 5 defects (no work-around)	BP	BP	x	- Provides for a quality delivery standard on application functionality.	Severity 4 should be reviewed with the product owner
Coding	**Prior to committing, all code must be approved by a tech lead/architect or approved developer (for very simple tasks)-** (Git Paradigm) The goal is that the person doing the review/approval has a sufficient understanding of the story to effectively review for quality. o Recommendation: During Sprint Planning Meeting, an Approver will be designated for that Sprint o Recommendation: If team has multiple resources that are qualified, please rotate this responsibility o Recommendation: Lead Developer/architect does the review o Recommendation: Whoever is chosen as code approver should not have so many coding tasks themselves that they cannot perform this task…their job for the sprint is to watch code quality, pair program as needed and approve all code check-ins	x			- Simulates the Git Paradigm of having someone approve all code commits - Forces code review by experienced developer which should increase quality	Tech lead or paired programming
	Unit Testing o Java Code – Full unit testing of any new or changed method o N-Cube – Create an N-Cube Test for any new or changed cube o Javascript – [TBD] o Groovy	BP			- Increases code quality by forcing the developers to exercise their designs outside of the user interface	
	Code Commits o Developer should review status of projects on our CI (Continuous Integration) server to ensure there are no errors and only commit if CI server is error free and all of the junit test, javascript tests, and N-Cube tester pass locally with synched up code. If there is a UI change, selenium tests should also be run locally to ensure they pass. Developer should then check the CI build after committing to make sure it still passes. o Overall Goal = 85%	**x**			- Tries to ensure the developers are merging and testing their code with the most up to date version of code.	
	Code Coverage o Overall coverage for uwd-ra-resources, uwd-ra-impl and N-Cube should never go down o Overall Goal = 85%		BP	BP	- Code coverage shows how much code has been exercised through the J-Unit tool. This metric, by itself, does not demonstrate the quality of the code because it does not show how thorough the tests are	
	Build Summary/Higher Environment Document oThe team that owns the build for each sprint will produce a Build Summary/Higher Enviornment Document which will be stored in Jive, detailing any issues found with the build, and the steps taken to fix them. This document will also summarize any issues found in higher environments throughout the sprint. At the end of the sprint the team that owns the build will communicate the results at the architecture review to all teams.		x		- The Build Summary/Higher Environment Document outlines any issues that may have occurred during a sprint with the build or higher environments, and what was done to resolve those issues. It helps ensures that duplicate issues aren't repeated and gives a brief summary to the next team that will be owning the builds	
	Release Summary Document oThe team that owns the build process for each sprint will also own the release and all of the deployment activities and artifacts that go along with it. At the end of each sprint, the team that owns the release will communicate any changes made to all teams at the architecture review.		x		- Ensures that our code is moved to higher environments without errors and contains all the proper artifacts.	
	Code Complexity o Class complexity should not be greater than [TBD] o Method complexity should not be greater than [TBD]		BP	BP	- Code complexity is a metric that shows how clean and simple a block of code is. The theory is that less complex code is less brittle and easier to maintain over time	
	Turning Best Practices into Required (Future Tasks) • Collect Code Coverage Metrics on javascript and N-Cube • Implement tools like Git that formalize the Technical Approving process prior to code commits				- Much of our logic is moving into javascript and N-Cube so it is critical that we exercise that code the same as our Java and Groovy code	
Documentation	Update Cube University content: o If a new feature has been added, review the existing CUBE U content and create new chapters as necessary. o If a feature / tool / architecture element has changed, review the existing CUBE U content and update chapters as necessary. o The audience for CUBE University content is primarily Developers so please consider relevance of content when adding anything new. o Please note that CubeU is not a system for tracking requirements!!!o Content that formerly resided in Jive should be marked as Deprecated and reader should be directed to Cube U. o Content that formerly resided in Jive should be marked as Deprecated and reader should be directed to Cube U. o Handling this activity via creating a new documentation User Story or task within an existing User Story is at discretion of	x			- Provides an accurate easily accessible knowledge base for all developers which reduces development time. - Treats knowledge assets with a discplined life cycle exactly as used for Code. - Forces review and curation by experienced developers which increases quality. - Inline edits of content by any developer will increase chances that errors are noted and corrected.	
	For further consideration: Code Documentation o Create README.md file in the Repository's base source directory; commit/push/PR to appropriate GitHub repository. If a README.md file already exists within the source tree, review for any impacts of changes in the Sprint and update/commit/PR accordingly to appropriate GitHub repository. o Create javadocs or groovydocs for each repository as appropriate; initially will require developers to add annotations to code.		x		- Provides "inline" documentation tied directly to the feature's code giving sufficient detail to cover any unique complexity.	

BP = Best Practice