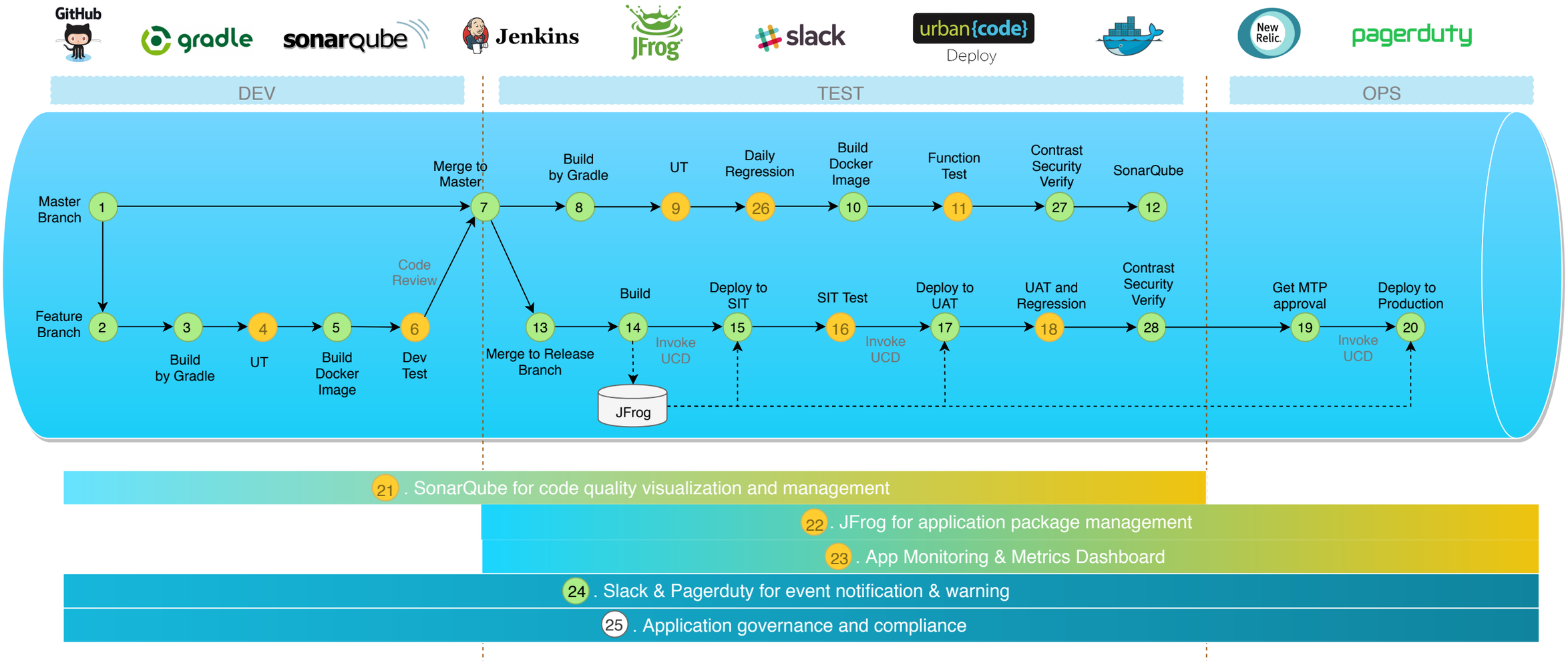
ESW ShopZ & LMS DevOps Pipeline

Ver.201805



	DEV	TEST	OPS
Phase I: Done Nov.2016 - Feb.2017 Core DevOps pipeline	1, 2, 3, 8: Migrated to Github, leverage social and agile development capability from Github Fast development workspace setup, improve productivity	16, 18: Automation test coverage is 24% for LMS 16, 18: Adopt RobotFrameWork for automation test 12: Set up SonarQube for code quality visualization	15, 17, 20: On demand automation deployment on SIT/UAT/PROD is with ONE click
Phase II: Done Mar.2017 - May.2017 E2E DevOps pipeline	6: Fast development test environment setup, improve productivity & support versioned env 5: Decouple application runtime in container for application transformation 24: All development activities status can be informed by slack channel	10: Adopt Docker for testing Functional Test 24: All test activities status can be shown in slack channel 21: Code quality can be visualized and measured by leveraging SonarQube in each iteration 22: Manage App build package by JFrog	23: Monitor pages, services, API, connection availability and performance, visualized dashboard by New Relic 24: Deploy status can be put in slack channel
Phase III: Doing Jun.2017 - Sep.2018 Improve code quality & auto-test coverage	4: New code is covered by unit test 21: Leverage SonarQube to fix technical debt and improve product quality iteratively	9: New code is covered by unit test 11: Update V18 automation test cases, increase function coverage 16: Add stub testing capability increase API & service coverage 27, 28: Add Contrast to monitor security 18, 26: Increase auto regression coverage, add capability of test case choreography	23: Monitor E2E ESW Orders by Hackathon idea - on bluemix 23: Monitor function availability (business flow, transactions) and performance by New Relic 24: Use PagerDuty for prod availability and performance warning
Phase IV: To Do Oct.2018 - Dec.2018 Governance and info management	25: Adjust process and R&R for app governance and compliance	22: Leverage JFrog to manage app lifecycle info through CI/CD	22: Leverage JFrog to deploy fully tested application on production

Legend:

- n — *Action #n has been done*
- n — *Action #n is in progress*
- n — *Action #n is not started*