Agile Test Strategy

Introduction to Agile

* Iterative approach to software development
* Highly collaborative
* Continuous design improvement as the guiding principles

Guiding Standards

* Shared responsibility

*Everyone is equally responsible for the quality*

* Data Management

*How data is managed for testing purposes*

* Test Management

*How to look after test cases and test codes.*

* Test Automation

*To achieve automation for unit testing, regression testing wherever feasible*

Requirements strategy

*In the spirit of Agile, the task items will need to re-prioritize and at any time, the team will implement the highest priority work tasks.*

Quality and Test Objectives

* Accuracy

*The features and functions work completely 100%*

* Integrity

*Sensitive information like login username/password need to be encrypted*

*Data displayed in the UI need to match those from the backend data sources.*

* Maintainability

*Need to be easier to add new features and fix defects and need to have high percentage of unit tests*

* Availability

*Target to be 99.9%*

* Inter-operability and compatibility

*Need to work across different platforms and environment for example if this is web based app, it need to be able to work well across different browser. If this is a mobile app, it need to be able to work on different iOS devices due to different screen resolutions and different versions of OS (iOS 4.x, iOS 5.x, etc) and different manufacturers(if applicable).*

* Performance

Measure of response time and throughput.

Test Scope

*Identify the types of testing required for the product*

* (Automated) Unit Testing

This can be created and done by the developer to unit test the code changes.

* Integration Testing

*To test the interface between different microservices (components) and ensure they work well together*

*Stubs may be used as substitutes of components*

* (Automated) Feature and functional testing

To validate against user requirements and feedback and ensure a happy flow

* Data conversion testing

*To test the data is converted correctly from the sources to the target*

* System testing

*To test the entire system as a black box testing. This is where regression and retesting is performed and to execute the actual scenarios. This is done in the staging server which is very much similar to the production server where the software would actually be deployed.*

* Security testing

*Testing for any unauthorized access and data privacy issue*

*Testing of unapproved access to secure pages should not be permitted*

*Restricted files should not be downloadable without permitted access*

*Ensure data integrity is not compromised.*

* Environmental testing

*The test it works on different supported platforms, OS, browsers, physical devices*

* (Automated) Performance and Availability testing

*This includes load testing where multiple virtual users concurrently using the system and the measure the response time and throughput.*

* (Automated) Regression testing

*This ensures all the previously functionalities are still working and all the previously closed bugs are still closed.*

* Acceptance testing

*This test against the test criteria, an condition that the product must be satisfied to be accepted by a customer, user and stakeholders.*

Test Design Strategy

* Use of device emulators

*During development of new app features and functionalities and for cost-effectiveness*

* Use of physical device

*In the final stage, the physical device need to be available to test the real life scenarios which includes factors like different battery states, multiple networks (Wifi or 4G or network density (limited connectivity, no connectivity, metered connectivity)*

* Use of Beta Testers

*Internal employees*

* Specification-based technique

*Interoperability between different components or microservices*

*Enhancement to unit testing*

* Experience-based technique

*Employed when there is no specification or inadequate specification and timeline is tight.*

Test Environment Strategy

* Development environment

*This is used for unit testing, functional testing*

* Staging environment

*This is used for automated testing and integration testing, system testing, regression testing and exploratory testing as well production verification testing*

* Production environment (optional)

Test execution strategy

* Agile testing must be iterative
* Testers cannot be rely on having complete specification due to changing requirements.
* Testers should be flexible
* They need to work closely with developers. Log jira ticket with clear, concise, sufficient information with appropriate screenshots for developers to work on the issues
* Focus on exploratory testing
* Focus on what and not how to test
* Define when to continue or stop testing before delivery to the customer.

Test automation strategy

*Use a planned approach*

*It may be written by the developer or tester and anyone in the team can contribute.  
Aim to automate stable and high priority test cases first*

*Always work towards to increase the quality of the test automation code to ensure ROI on the test automation effort.*

*Aim to implement codeless automation to help release apps quickly*

DevTest workflow strategy

*Define behaviors in ubiquitous language >> feature files (scrum team)*

*Write feature files in Gherkin syntax (testers)*

*Write code (Developers)*

*Create step definitions (Automation engineer)*

*Automation project/continuous testing*

*What BDD brings into the equation?*

*There is collaboration between the team and everyone is talking about the testcases and they do that before they start work so that actually driving the design and everyone is on the same page and it’s written in a natural language that everyone can understand so that we can share this information very easily and no need to understand code and everyone can participate in this process.*

*Testers write and maintain use case (feature files) using Gherkin*

*Automation engineer to write and maintain step definitions.*

*BDD and codeless testing impact on testing speed and quality*

*Higher test maintainability – reusable ‘step definitions’ simplify test modification and debugging: update once and impact entire test suite*

*Using natural language to write tests make it easy to identify test scenario and ensure coverage*

*Testers with domain expert can quickly wite tests with no need to code.*

*Example of a BDD and cucumber based test*

*Feature file (written in Gherkin “steps”)*

*Feature: Login to Eribank*

*Correct login credentials open application home page presenting the account balance*

*Scenario: Login with valid credentials*

*Given Eribank application is open*

*When I enter username as “company”*

*And I enter password as “company”*

*And I clicks login button*

*Then I see the application home page*

*And I see the account page*

*Scenario: Login with in-valid credentials*

*Given Eribank application is open*

*When I enter username as “company”*

*And I clicks login button*

*Then I see wrong username or password page*

*Step definition (Created in code (Java)*

*@When(“I enter username as \”(.\*)\\*$”)*

*public void i\_enter\_username\_as(String incorrect) throws Throwable {*

*Driver.findElement(By.xpath(“//\*[@id=’usernameTextField’]”)).sendKeys(incorrect);*

*}*

*The scenarios is divided into 3 parts -Given, When , Then And*

*Given is the basic assumption*

*When is the condition what need to happen*

*And then is the expected result*

*Reference: http://www.automationtestinghub.com/cucumber-bdd-with-selenium-introduction/*

Test data management strategy

*Test a subset of the production dataset and customer sensitive data should not be used in the testing environment.*

Test Management

*Currently use of MS excel on sharepoints and Jira to track the test cases and test.*

Risk Management

*The risk in the daily standup meeting will be logged immediately and someone will be made an owner of each risk, analyse the risk and take appropriate actions.*

Defect Management

*Jira is used to perform defect classification, defect life cycle.*