# Approvals

|  |  |  |
| --- | --- | --- |
| **Name** | **Role** | **Approval** |
| Robert Pepitone | QA Manager |  |
| Kanika Lal | QA Manager |  |
| Keling Cheng | QA Test lead |  |
| Elizabeth Garcia | QA Test Lead |  |
| Nandini Chintak | QA Test lead |  |

Contents

[Approvals 1](#_Toc479783269)

[Table of Contents 2](#_Toc479783270)

[Roles and Responsibilities 2](#_Toc479783271)

[Tools 4](#_Toc479783272)

[UFT 4](#_Toc479783273)

[ALM 4](#_Toc479783274)

[SVN 4](#_Toc479783275)

[Selenium 4](#_Toc479783276)

[Cucumber 4](#_Toc479783277)

[Regression Test Maintenance Strategy 5](#_Toc479783278)

[Long-term maintenance 5](#_Toc479783279)

[Regression Test Execution Strategy 6](#_Toc479783280)

[Environment 6](#_Toc479783281)

[Timelines 6](#_Toc479783282)

[Smoke Tests 6](#_Toc479783283)

[Automated Regression Testing 6](#_Toc479783284)

[Automated Regression Scripts Criticality 7](#_Toc479783285)

[Manual Regression Testing 7](#_Toc479783286)

[Test Evidences Maintenance 7](#_Toc479783287)

[Resource Allocation 7](#_Toc479783288)

[Defect Management 8](#_Toc479783289)

[Sign-Off Criteria 10](#_Toc479783290)

# Roles and Responsibilities

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Role** | **Areas** | **Location** |
| Robert Pepitone | QA Manager |  | Harrison |
| Kanika Lal | QA Manager |  | Harrison |
| Keling Cheng | QA Test lead | Dynamic Forms, PSD, Current Investments/Auto Rebalance, Services, FE, DCMA | Harrison |
| Elizabeth Garcia | QA Test Lead | Loans and Withdrawals P3 | Harrison |
| Nandini Chintak | QA Test lead | Withdrawals DDOL, Mobile, Remittance, PV | Harrison |
| Beatriz Bergamaschi | e-Core (Brazil) Team Lead |  | Porto Alegre, Brazil |
| Danilo Guimaraes | QA Analyst | Selenium | Porto Alegre, Brazil |
| Andre Saez | QA Analyst |  | Porto Alegre, Brazil |
| Camila Bulmini | QA Analyst |  | Porto Alegre, Brazil |
| Cesar Pedroso | QA Analyst |  | Porto Alegre, Brazil |
| Francine Bergmann | QA Analyst |  | Porto Alegre, Brazil |
| Guilherme Heck | QA Analyst |  | Porto Alegre, Brazil |
| Leandro Amavisca | QA Analyst |  | Porto Alegre, Brazil |
| Marcia Cardoso | QA Analyst |  | Porto Alegre, Brazil |
| Marcus Kobel | QA Analyst |  | Porto Alegre, Brazil |
| Marin Gechev | QA Analyst |  | Porto Alegre, Brazil |
| Nathalia Mandelli | QA Analyst |  | Porto Alegre, Brazil |
| Rafael Mendonça | QA Analyst |  | Porto Alegre, Brazil |
| Raoni Domingues | QA Analyst |  | Porto Alegre, Brazil |
| Roberto Teixeira | QA Analyst |  | Porto Alegre, Brazil |
| Victor Silva | QA Analyst |  | Porto Alegre, Brazil |
| Michael Bayley | QA Test Lead, Data Specialist | Accting | Harrison |
| Iryna Kobzeva | QA Test Lead | Statements | Harrison |
| Annmarie Santucci | QA Test Lead | EDS | Harrison |
| Lynn Pettipaw |  |  |  |
| Amro |  |  |  |
| Brian Gillete |  |  |  |
| Angelo |  |  |  |
| Rajesh |  |  |  |
| Lisa Onorato |  |  |  |
| Adil |  |  |  |
| James |  |  |  |
| Hans |  |  |  |
| Helen |  |  |  |
| Prashant |  |  |  |
| Samir |  |  |  |
|  |  |  |  |

# Tools

Here are the tools involved in the testing process covered by this test plan:

### UFT

UFT stands for Unified Functional Testing. It is HP’s test automation suit that is able to automate many different kinds of application. (Web and Desktop applications, for example)

### ALM

ALM stands for Application Lifecycle Management. It is HP’s test management application that provides functionality to manage the UFT scripts and the test results. It is the default test results repository for UFT scripts execution.

### SVN

SVN is an abbreviation for Subversion, which is Apache’s software versioning control system. It is used to keep the history of changes in the code of a software and allows to backtrack to previous versions if needed. It also allows many different developers to work in the same code at once without affecting the stable version of it. It is used to control the code of the Selenium automation scripts.

### Selenium

Selenium is a web browser automation library that can be used for different purposes, including testing automation.

### Cucumber

Cucumber is a BDD process integration tool, in which people is allowed to create documentation for an application using a language called Gherkin. This documentation is created using a language similar to natural English, and includes test scenarios that can later be automated and executed straight from the documentation.

# Applications

Here is the list of applications covered by this test plan:

## DDOL

## Paris III

## PSD

# Test Environments

## Test

The test environment contains the application code that is up-to-date with the sprint deliveries. All integrated manual functional testing is done in the Test environment as this is updated as soon as the developer finishes a development ticket. The code in this environment is considered unstable as this is where the code to be tested is deployed.

## Regression / ITECH

The ITECH environment is the stable code environment. After the code is tested in Test, it is moved to the ITECH environment on-demand for regression testing. This environment has a database refresh process that can restore the data to its original state and is meant to be used by the automation scripts.

Application URLs

|  |  |  |  |
| --- | --- | --- | --- |
| Application | Environment | URL | Database |
| DDOL | Regression (ITECH) | http://162.123.113.25/ddol/login/login.htm | PAREGR |
| Paris III | Regression (ITECH) | http://crasdiatst01/parisIII/login/logIn.jsp | PAREGR |
| PSD (itech login) | Regression (ITECH) | http://162.123.113.25/ddol/login/login.htm | PAREGR |
| PSD (participant login) | Regression (ITECH) | <http://crasdiatst01:81/PSDirect/signin.jsp> | PAREGR |
| DDOL | Test | https://ddoltest.divinvest.com | DIVRDIVT |
| Paris III | Test | http://paris3.test.divinvest.com/parisIII/login/logIn.jsp | DIVRDIVT |
| PSD | Test | https://psdtest.divinvest.com/PSDirect/signin.jsp | DIVRDIVT |

# Regression Test Execution Cycle

The full regression cycle is based on a 4-week cycle, as follows:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Week 1 | | | | | Week 2 | | | | | Week 3 | | | | | Week 4 | | | | |
|  | 1 | 2 | 3 | 4 | 5 | 1 | 2 | 3 | 4 | 5 | 1 | 2 | 3 | 4 | 5 | 1 | 2 | 3 | 4 | 5 |
| Weekly Patch Regression Tests |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Release Regression Execution |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Move code from Test to ITECH |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Sprint 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Sprint 2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

A patch is applied in Production every Wednesday and the full release is deployed to Production every end of month. The regression is executed partially every Wednesday and the full regression set is executed by the end of the cycles, after the code is moved to the ITECH environment.

The sprint timeline is usually 2 weeks and begin on a Monday and end on a Friday. This timeline may change accordingly to the teams.

The regression execution timeline may change accordingly to the ability of the teams to have all the environment setup done on time and on resource allocation. Reducing the amount of time or resources available for executing the Regression scripts may impact on:

* Finishing the regression execution after the expected date;
* Not executing the full scope of the regression scripts;
* Reporting issues in the Test /Regression environment that may affect the Production environment after the code has been moved to production.

# Regression Test Execution Strategy

## Pre-Conditions

Here is a list of pre-conditions that need to be met so that the Regression scripts can be executed:

* Code must have been moved from Test to ITECH;
* ITECH database must have been refreshed.
* Changes in ITECH database schema have been applied.
* Links in regression ITECH must be pointing to the correct environment;
* Smoke tests for each application must have been executed and issues communicated.

## Smoke Tests

Smoke tests are going to be executed across different areas of DDOL and P3 with the following scope:

* **Paris3**
  + Transactions:
    - Process a Loan Request
    - Process a Withdrawal Request
    - Process a Transfer Request
    - Remittance Detail
    - Remittance Reversal
  + CTI
  + Participant/Maintenance
  + Participant/New Enrollment (create new participant)
  + Plan Setup
  + Reports
    - Fulfillment Options
    - Investment Mix
* **DDOL**
  + Transactions:
    - Create a Loan Request
    - Create a Transfer Request
    - Process a Rebalance
    - Subscribe to an Auto-Rebalance
    - Create a Withdrawal Request
  + Enrollment Path
  + E-Documents
  + Account Balance
  + On Track
  + Activity Summary
  + Contributions
  + Allocations
  + Current Allocations
  + Portfolio Express
* **PSD**
  + For a given case:
    - Open all tabs
    - Generate a report
    - Open generated report
    - Create new PSD user
  + Participant Information / Deferral

## Automated Regression Testing

After the smoke tests are done the execution of the automated test scripts will begin. There are scripts automated using Selenium and UFT. Here is the approach for each of them:

* **UFT:**
  + UFT scripts are executed locally in the QA analyst machine.
  + Results must be stored in ALM.
    - ALM URL is <http://hnyasxfree02:8080/qcbin/start_a.jsp>
    - Project/Domain: DEFAULT/ITECH
* **Selenium:**
  + Selenium scripts are going to be executed remotely using Jenkins. The following projects are going to be used:
    - **ddol\_psd\_paris3**: the main script repository that uses Cucumber for test writing and execution.
    - **ddol**: the alternative Selenium automation using TestNG for DDOL.
    - **paris3**: the alternative Selenium automation using TestNG for Paris3.
  + The projects mentioned above can be found at the SVN server at <http://crasswvc01.us.aegon.com/svn/com.transamerica.tbs.test/selenium>
    - Access to the SVN folder is provided as described in the following document: <http://confluence.transamerica.com:8090/display/QA/1.1.+Requesting+SVN+Access>
  + In case there are issues with the Jenkins infrastructure, the Selenium scripts can be executed locally by a QA Analyst.
  + Reporting results are going to be stored at S:\Selenium\Reports
    - Jenkins jobs should copy the results to this folder automatically.
    - In case of local script executions the test must copy the results manually to this folder.
    - Please contact Danilo Guimarães if you need access to this folder.

## Automated Regression Scripts Criticality

In cases where not all the scripts can be executed, execution will move on accordingly to the criticality of the automated test scripts listed below:

### DDOL

|  |  |  |
| --- | --- | --- |
| Script Name | Area | Criticality |
| Account Balance |  |  |
| Account Summary |  |  |
| Allocations |  |  |
| Auto-Rebalance |  |  |
| Beneficiary |  |  |
| ContribEstimator |  |  |
| Contributions |  |  |
| CurrentInvestments |  |  |
| Custom Portfolios |  |  |
| E-Delivery |  |  |
| E-Documents |  |  |
| Investment Mix |  |  |
| IRA Rollover |  |  |
| Loan Activity |  |  |
| Loans |  |  |
| Login\_Beneficiary\_OnTrack |  |  |
| ManagedAdvice\_Ontrack |  |  |
| Message Center |  |  |
| Mobile |  |  |
| Multiple Browser - Chrome |  |  |
| Multiple Browser - Firefox |  |  |
| NQDC |  |  |
| OnTrack |  |  |
| Password Reset |  |  |
| Sanity Check |  |  |
| Savings Calculator |  |  |
| Activity Summary |  |  |
| Telephone Check |  |  |
| Transfers |  |  |
| Transfers SPL |  |  |
| Activity Details |  |  |
| Username and Password Management |  |  |
| Account Overview |  |  |
| TA\_DB\_before |  |  |
| TA\_DB\_after |  |  |
| TA\_reset\_data |  |  |
| ddol\_DMA\_Linux\_main |  |  |
| ddol\_withdrawal\_infoOnly\_linux\_main |  |  |
| ddol\_PX\_linux\_main |  |  |

### Paris3

|  |  |  |
| --- | --- | --- |
| Script Name | Area | Criticality |
| Outsourcing Eligibility | Eligibility |  |
| Batch Run | Participant |  |
| E-Documents | Participant |  |
| EM\_NewEnrollment | Participant |  |
| Maintenance | Participant |  |
| New Account | Participant |  |
| NewEnrollment | Participant |  |
| NDQC | Participant |  |
| Rehire | Participant |  |
| 001 – Portfolio Transfer Allocation | Plan Setup |  |
| 009 – Conversions | Plan Setup |  |
| 016 – Funds | Plan Setup |  |
| 020 – Outsourcing | Plan Setup |  |
| 027 – Report Output | Plan Setup |  |
| 031 - Transfers | Plan Setup |  |
| 035 - Information Sharing | Plan Setup |  |
| Custom Portfolios | Plan Setup |  |
| p3\_plan\_main\_part1 | Plan Setup |  |
| p3\_plan\_main\_part2 | Plan Setup |  |
| p3\_plan\_main\_part3 | Plan Setup |  |
| p3\_plan\_main\_part4 | Plan Setup |  |
| Plan Setup Rehire | Plan Setup |  |
| Reporting | Reporting |  |
| ACH\_Remittance\_Funding | Transactions |  |
| Loans | Transactions |  |
| Remittance Details | Transactions |  |
| Reversals | Transactions |  |
| Transfers | Transactions |  |
| Withdrawals | Transactions |  |

### PSD

|  |  |  |
| --- | --- | --- |
| Script Name | Area | Criticality |
| General Testing | General Testing |  |
| Custom Report | Reporting |  |
| UserMaintenance | User Maintenance |  |
| Loans and Withdrawals | Reporting |  |
| Participant Information | Reporting |  |
| Plan Balances | Reporting |  |
| Plan Reports | Reporting |  |
| Transaction Reports | Reporting |  |

# Manual Regression Testing

Manual regression testing is expected to happen under the following conditions:

* Smoke tests;
* Defect retesting;
* An automated script is broken and it won’t be fixed within the regression timeline.

# Regression Test Maintenance Strategy

Regression scripts are going to be maintained/fixed as the resource allocation allow it to be done between regression execution cycles. Maintenance will be done accordingly to notes taken in the last regression execution results.

Maintenance will be done accordingly to the criticality of the automation scripts as it has been mentioned before.

## Long-term maintenance

* Scripts may be removed from scope due to changes in the application. Those scripts are going to be marked as “Deprecated” in ALM or will be flagged with @ignore @deprecated in Cucumber feature files.
* As the coverage increases, there will be too many scripts to maintain the data and code up-to-date.
  + In this case, the criticality of the scripts will have to be reviewed and less critical scripts won’t be maintained and/or executed anymore.
  + The amount of scenarios executed per script may be lowered on less critical scripts.

# Test Evidences Maintenance

This section describes the policy for test evidence retention.

For each test evidence stored, the following information is considered the minimum to make sure the test results are traceable:

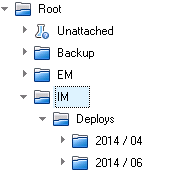
* Application
* Test Name
* Test Data
* Execution Date
* Responsible Tester

## UFT Test Executions

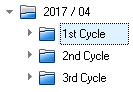
UFT test execution evidences are stored in ALM. For the scope of this test plan, the project and instance used are the following ones:

* ALM URL: <http://hnyasxfree02:8080/qcbin/start_a.jsp>
* Project/Domain: DEFAULT/ITECH

A new folder is going to be created under **[Test Lab]Root/IM/Deploys/** with the name as **yyyy / mm** as seen in the picture below:



Inside this folder, a new folder named as **1st/2nd/3rd/4th Cycle** has to be created accordingly to how many executions have been done so far. Example:



The cycle folder must have the test sets as needed for the regression execution.

## Selenium Test Executions

Selenium test execution evidences must be stored in the following url:

S:\Selenium\Reports

## Manual Test Executions

## Evidence Retention Policy

Test evidences have to be kept for a minimum of 2 years after the test execution. They can be deleted later to free up disk space if needed.

# Resource Allocation

Here is the average resource allocation for regression execution per regression cycle:

* xxxxxxx – 100%, x days
* xxxxxxx – 100% , x days
* xxxxxxx – 100% , x days

Regression execution will be done even if those requirements are not met.

# Defect Management

Defects are managed in Jira in the project Information Technology (IT). Before the defect is logged, the tester needs to check if the issue can be reproduced in Test environment and add this information to the ticket.

There are two main types of defects to be used:

* **Testing Issue**: To be used in case the defect can be reproduced in Test.
* **Regression Issue**: To be used in case the defect cannot be reproduced in test or in case of environment issues.

**Important:** the project EC shouldn’t be used for managing Regression issues anymore, to avoid duplicating data and having to track two defects.

The defect must have at least the following information aside from mandatory fields:

* Defect description, containing the environment and the context in which the test is being executed.
* Steps for reproducing the defect.
* Screenshot showing the error (if applicable).
* Data used in the test
* Application name and area.
* The scenario text in case of Cucumber scenarios.

The defect priority must be defined accordingly to the following criteria:

|  |  |
| --- | --- |
| Priority | Definition |
| 1 – Critical | Defect prevents further testing in multiple or a primary functional area. |
| 2 – High | Defect prevents further testing in a functional area. |
| 3 – Medium | Defect in non-critical functional area does not prevent further testing, or a high defect with a workaround in place. |
| 4 – Low | Defect not impacting any functionality, such as minor screen formatting issues, or medium defect with a workaround in place. |

The defect priority must be changed if needed and should be kept up-to-date to reflect the current defect impact for the application/project.

The comments field in the issue at Jira must be used for defect tracking purposes. It must be avoided to follow-up issues and defects by e-mail as this history won’t be kept along with the issue. It is recommended to communicate the persons related to each functional area that an issue has been created in Jira along with its URL, but the comments section in the issue must be used for further tracking . The table at [Roles and Responsibilities](#_Roles_and_Responsibilities) can be used in order to choose the right persons to add in the communication.

Defects are going to be tested and validated accordingly to the ability of the teams to have them ready to test within the regression test window.

# Sign-Off Criteria

* All Critical and High defects found during the regression cycle must be “ready to prod” or “completed/withdrawn”, except if they have an explicit ETA postponing them to another date.
* All Critical, High or Medium defects opened during the regression cycle and that are reproducible in Test environment must be “ready to prod” or “completed/withdrawn” except in case they have an explicit ETA postponing them to another date.
* Medium regression defects without a resolution must have an ETA.
* At Least 80% of all Critical test scripts must be passed.