|  |
| --- |
| Automation Tester Guide |



Table of contents

[1. Developer Desktop 2](#_Toc470007920)

[2. Framework Overview 2](#_Toc470007921)

[3. Technology Stack 3](#_Toc470007922)

[4. IDE Setup 3](#_Toc470007923)

[4.1 Configuring Maven 3](#_Toc470007924)

[4.2 Configuring Git Repository 4](#_Toc470007925)

[5. Working with Git 4](#_Toc470007926)

[5.1 Checkout an existing project from Gitlab repository 4](#_Toc470007927)

[5.2 Using Git from the command line 5](#_Toc470007928)

[6. Database configuration 5](#_Toc470007929)

[7. Customer and Digi-Cert Creation 5](#_Toc470007930)

[7.1 Create Customer’s and ROS Digi-certs 6](#_Toc470007931)

[7.2 Linking Agent and Customer 6](#_Toc470007932)

[7.3 CRS and ITP Customer Creation Tool (Excel) 6](#_Toc470007933)

[8. Automation Framework Workflow 6](#_Toc470007934)

[9. Important links and folders 7](#_Toc470007935)

# Developer Desktop

Firstly, a developer desktop is required for setting up the test automation framework. To request it a ticket can be raised via ICT self-help for the same: <http://revnet2.revenuedomain.ie/web/ictl/ict-help>. Log a request and fill out the mandatory details. Just ensure that you get the approval from the AP/Manager for the developer desktop so that the approval mail can be attached with the request to speed up the process.

Once the desktop has been set up, you will receive a confirmation email from the Revenue Service desk and the Revenue Desktop Refresh project team.

# Framework Overview

For test automation, we use:

* Cucumber
* Selenium
* Spring

[**Cucumber**](https://cucumber.io/) allows for tests to be written and documented in clear and simple English. The tests follow an easy to follow approach of **Given** some conditions, **When** the certain actions are performed, **Then** a certain result should be achieved.

[**Selenium**](http://www.seleniumhq.org/) is used for connecting to real web browsers programmatically to execute the necessary steps to carry out the test.

[**Spring**](https://spring.io/)allows good programming practice to be followed for automated test code in the same way as for normal application code.

Cucumber allows a BDD approach to test automation through the use of its feature files which enables better readability and understanding from business point of view.

The programming language used is Java and the IDE used is Eclipse or Spring Tool Suite (STS). It is recommended to use STS to have better compatibility with java libraries. It should help ensure that the plugins we use will be compatible with the software versions we use in Revenue

# Technology Stack

After getting the developer desktop, just make a quick check to ensure all the tools and softwares are installed as listed in the technology stack when you first log in.

* Programming Language- Java (To check if you have Java installed, open command prompt and type “***java -version***”. If installed correctly, it should display your java version)
* IDE- [Eclipse](https://eclipse.org/), [STS](https://spring.io/tools)(Spring Tool Suite) – Development environment in which you will be creating all the automation projects
* Testing tools- [Selenium](http://www.seleniumhq.org/), [Cucumber](https://cucumber.io/), [Junit](http://junit.org/junit4/), [Test-NG](http://testng.org/doc/index.html)

(Selenium is used as we are doing test automation for web applications. Junit and Test-NG are the testing frameworks which we use to execute the different test scenarios)

* Build Manager- [Maven](https://maven.apache.org/) (To build projects and handling dependencies)
* Database tools- [Squirrel](http://confzone.revenuedomain.ie:8080/display/DEV/SQuirreL+SQL) (Simple GUI to access the database)
* Repository- [Git](https://revcode.revenuedomain.ie/groups/automated-testing) (Code Repository where all the test automation projects are stored and maintained)

**\*\*To know more, just click on any of the mentioned tools for detailed information\*\***

**Click** [**here**](http://confzone.revenuedomain.ie:8080/display/TEST/Automation+Test+Tools+in+use) **to view in detail about the tools used in Test Automation**

# Integrated DEVELOPMENT ENVIRONMENT (IDE) Setup

In Revenue, we use Maven to build projects and handling dependencies for your Java project

## Configuring Maven

Ensure that every test automation project in your IDE has the following dependency in the pom.xml

***<dependency>  
 <groupId>ie.revenue.common</groupId>  
 <artifactId>autotest</artifactId>  
 <version>17.4.0-SNAPSHOT</version>  
</dependency>***

It includes common architecture classes and other reusable methods for writing automated tests.

Also, ensure that there is a **settings.xml** file configured in the following path< **C:\Users\<username>\.m2**

> like the one embedded in this document:



Edit the following line in the settings.xml to point at your local repository with your username:

<localRepository>C:\Users\<username>\.m2\repository</localRepository>

Also, make sure that the IDE is pointing at the file as well:

**Eclipse->Preferences->Maven->User Settings**

## Configuring Git Repository

All the test automation projects are stored in the Revenue Gitlab repository. You can use the revenue credentials to login to Git to view the existing projects. Click on the [link](http://confzone.revenuedomain.ie:8080/display/ATS/GitLab) for more details.

For integrating Git into Eclipse, see the following [link](http://confzone.revenuedomain.ie:8080/display/ARCH/How-to+setup+Eclipse+for+GIT) for step-by-step instructions.

To have a general overview or to know about the Git basics, see the following [link](http://confzone.revenuedomain.ie:8080/display/ARCH/How-to+GIT+basics) to get a basic understanding

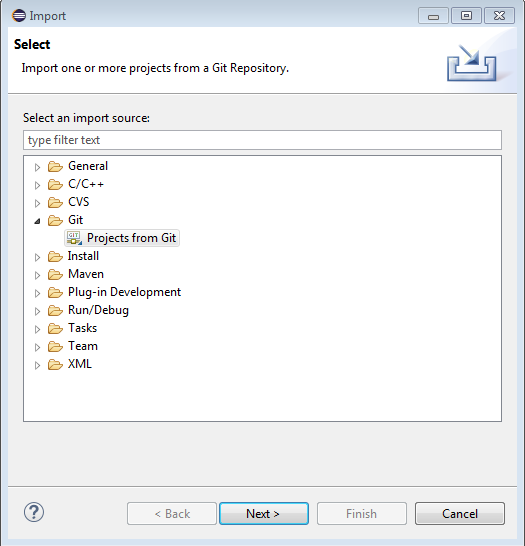
The existing automated projects can be found [here](https://revcode.revenuedomain.ie/groups/automated-testing) which can be used as a reference while starting a new project based on the selenium/cucumber framework

# Working with Git

## Checkout an existing project from Gitlab repository

Steps:

1. To do a checkout or pull an existing project from the Gitlab, log into Gitlab and open the project which you want to check out.
2. Copy the project link
3. Open Eclipse->File->Import->Projects from Git



1. Select Clone URI as Repository Source->Paste the project link
2. You can select the branches which you want to import then click on Finish

## Using Git from the command line

***Steps to checkout an existing project***

1. *Open command prompt*
2. *Navigate to the project folder*
3. *git init*
4. *git config --global user.name “username“*
5. *git config --global user.email <revenue Email>*
6. *git clone <project link>*
7. *git fetch\*\**
8. *git remote –v\*\**

**Steps to create a branch, make changes and push it to the repository with the added changes**

1. git branch
2. git branch <branch\_name>
3. git checkout <branch\_name>
4. git status
5. Add something and execute git status
6. git add filename\*\*\* , git status
7. git commit –m “commit reason”
8. git log
9. git show HEAD
10. git merge origin/master \*\*\*

A full list of commonly used basic Git commands, click on the following [link](http://confzone.revenuedomain.ie:8080/display/Excise/Basic+Commands).

[Git Cheat sheet](http://confzone.revenuedomain.ie:8080/display/Excise/GIT+Cheatsheet) is also available for quick reference

# Database configuration

Squirrel SQL should be installed in the developer desktop. If not, you can download it from [here](http://squirrel-sql.sourceforge.net/#installation).

To configure Squirrel, click on the following wiki [link](http://confzone.revenuedomain.ie:8080/display/DEV/SQuirreL+SQL) for more details and follow the steps provided.

For configuring databases as per different environments which has said by your manager/AO, see the following [link](http://confzone.revenuedomain.ie:8080/display/RM/1.6+Environment+Configuration) to see the credentials required to add different databases like Itp, Crs,Ros etc.,

# Customer and Digi-Cert Creation

While automating applications via ROS, you require a ROS digi-cert to login which you might have to create in order to automate/execute the test cases. Below listed are instructions on how to:

## 7.1 Create Customer’s and ROS Digi-certs

As a tester (and in many other roles at Revenue), you may have to make your own test data from time to time. Creating test data involves creating a new customer, adding different tax heads to this customer profile and (depending on the scenario) applying for ROS digi-certs. The process can be tedious but the link provided [here](http://confzone.revenuedomain.ie:8080/display/MAE/Creating+a+ROS+Digicert) gives a detailed description of how to create new customers, new tax heads to this customer and how to create ROS digi-certs.

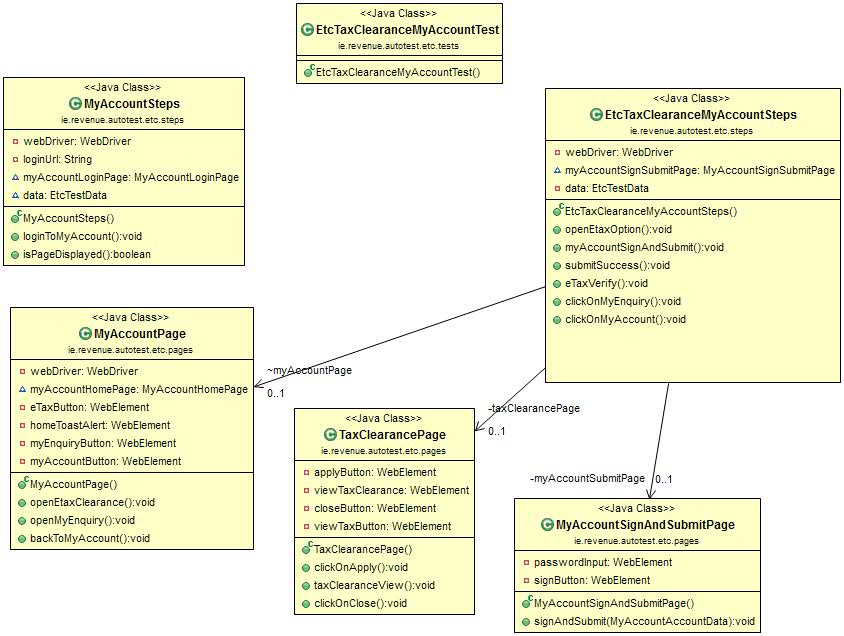
## 7.2 Linking Agent and Customer

After creating test data, you may also need to link an agent profile to a customer profile in order to carry out certain tests. The link provided [here](http://confzone.revenuedomain.ie:8080/display/TEST/RCM+-+Create+Link++Agent+to+Customer?preview=%2F148799507%2F148799510%2FCreating+a+link+Agent+to+Customer.doc) details exactly how to go about linking an agent and a customer

## 7.3 CRS and ITP Customer Creation Tool (Excel)

This tool is an Excel workbook with macros that can create simple customers in CRS & ITP. There is one tool per environment. The tool generates SQL insert scripts, within the Excel workbook, for each core table on CRS & ITP. It then saves these scripts to a location where a file runner job picks up and deploys the scripts every hour into the appropriate environment. The customers are then accessible via the applications. A link to the guide on this tool is available [here](file:///S:\Public\ITS\ITP\TESTING\Test%20Support\Business%20Processing%20Disturbance%20Testing\How%20To\CRS%20and%20ITP%20Cust%20Creation%20Tool\the%20CRS%20%20ITP%20Customer%20Creation%20Tool%20Guide.doc).

# Automation Framework Workflow

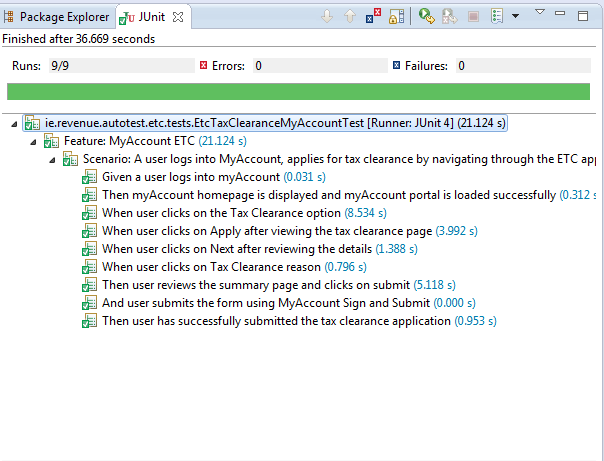


The above UML diagram is an example for the E-Tax clearance application. The project can be viewed in the Gitlab by clicking the [link](https://revcode.revenuedomain.ie/automated-testing/etc-automated-test)

**EtaxClearanceMyAccountTest** class is the test runner which reads the cucumber feature file for the test steps.

For every step mentioned in the feature file, there is a function defined in the **MyAccountSteps** and **EtcTaxClearanceMyAccountSteps** class which is defined under the *steps* package of the project.

Each function defined in the steps class is connected to the Page class in which all the functions are defined to interact in different ways to the components on the page like entering PPSN, loading ROS digi-Certs for ROS login, clicking web elements etc.



* After execution of the test case, summary of the test steps can be viewed in the IDE which shows the status of each step (Green tick indicating successful execution, will be red otherwise)

# Important links and folders

* [Revnet](http://revnet2.revenuedomain.ie/)- Revenue internal portal providing access to services like MyClockwise, ICT self-help etc.
* [Confluence Dashboard](http://confzone.revenuedomain.ie:8080/dashboard.action)- Revenue wiki for technology related information
* [Project Folders](file:///S:\Public\ITS\ITP\SYSTEM%20TEST\05.%20Projects) – Contains all the previous and upcoming releases documentation
* [ICTL Developer Guides, Tools and Software](http://confzone.revenuedomain.ie:8080/display/DEV/Home) – Contains useful information like developer guides in general for different technologies like Spring, Hibernate
* [Automated Testing](http://confzone.revenuedomain.ie:8080/display/DEV/Automated+Testing) – Information about automation testing using the new framework
* [Test Automation Framework](http://confzone.revenuedomain.ie:8080/display/TEST/Automation+Framework)- Detailed info on the framework design, guidelines on using the framework