# Disclaimer

This document is contains the thought process while strategizing for the test cases based on the Requirements and Acceptance Criteria given.

## Process

The features were first laid out with corresponding user stories and acceptance criteria provided to have a clearer overview, followed by analysing and listing assumptions of certain unclear or unspecified factors. Based on the features, modules categorisation was decided, before listing the straight-through test cases. Test cases were then elaborated with boundary values and test data variations, before prioritising high-risk test cases to be scripted for the submission.

Tools were researched on and considered before the file structure was determined to best cater to the Page Object Model, before creating the pages required. Web elements required were identified and the custom keywords required to interact with the elements were also added in the script. With each custom keyword added, unit testing was done to ensure that it is working as intended, so that debugging for scripts will be easier later on.

Test cases with similar test steps were then identified for template building later. Because most of the templates are for invalid test scenarios, straight-through testcases were first scripted. With that, test data csv files were built and populated with values, before moving on to scripting the templated test cases. All the tests go through multiple run throughs and debugging before compiling a batch file for easier execution for submission.

# Requirements

|  |  |
| --- | --- |
| Features | User Stories and Acceptance Criteria |
| F1: Enable **Clerks** to populate a list of working class heroes to the system | **(1) As the Clerk, I should be able to insert a single record of working class hero into database via an API**  AC1: Single record of a working class hero should consist of Natural Id (natid), Name, Gender, Birthday, Salary and Tax paid  **(2) As the Clerk, I should be able to insert more than one working class hero into database via an API**  AC1: Enhancement of (1) , with the ability to insert a list  **(3) As the Clerk, I should be able to upload a csv file to a portal so that I can populate the database from a UI**  AC1: First row of the csv file must be natid, name, gender, salary, birthday, tax  AC2: Subsequent rows of csv are the relevant details of each working class hero  AC3: A simple button that allows me to upload a file on my pc to the portal  *Question:*  *1. Notice that when request submitted, response is 202 Alright, not 200 OK. Is that expected? What is the requirements to achieve 200 OK response?*  *2. Single and multiple record insertion order is birthday 🡪 salary 🡪 tax. For csv it is salary 🡪 birthday 🡪 tax? Does the order matter?* |
| F2: Enable **Bookkeepers** to retrieve the payable taxation relief for each working class | **(4) As the Bookkeeper, I should be able to query the amount of tax relief for each person in the database so that I can report the figures to my Bookkeeping Manager**  AC1: a GET endpoint which returns a list consist of natid, tax relief amount and name  AC2: natid field must be masked from the 5th character onwards with dollar sign ‘$’  AC3: computation of the tax relief is using the formula as described:    AC4: After calculating the tax relief amount, it should be subjected to normal rounding rule to remove any decimal places  AC5: If the calculated tax relief amount after subjecting to normal rounding rule is more than 0.00 but less than 50.00, the final tax  relief amount should be 50.00  AC6: If the calculated tax relief amount before applying the normal rounding rule gives a value with more than 2 decimal places, it should  be truncated at the second decimal place and then subjected to normal rounding rule |
| F3: Enable **Governor** to dispense the money to each working class hero at her discretion | **(5) As the Governor, I should be able to see a button on the screen so that I can dispense tax relief for my working class heroes**  AC1: The button on the screen must be red-colored  AC2: The text on the button must be exactly “Dispense Now”  AC3: After clicking on the button, it should direct me to a page with a text that says “Cash dispensed”  *Question:*  *1. Cash is able to dispense even though there’s no records?*  *2. After clicking “Dispense Now”, list will not clear – so able to dispense to the same working heros again – is that intended?* |

# Assumptions

Testcases will be built with the below assumptions:

* While insertion with “” value passed in for gender, name and natId will not return Response 500 error, it will cause error on the main page. Therefore, testcases will be built with the assumption that all field are required to have a valid value.
* It was not state the data format and type requirements for each field (other than date format) to achieve Reponse 200 OK, therefore testcases will be built with the assumption that Response 202 Alright signifies successful execution.
* The order of the information given in Single Record doesn’t matter – request will still return Response 202 Alright.
* Because there is no way to verify if cash is truly dispensed, success message “Cash dispensed” will be assumed that it signifies that it is dispensed.
* To prioritize testing for high-risk scenarios, some testcases (listed in blue) are grouped under ‘Extra’ and not scripted for submission due to one or more reasons:
  + Not listed as requirements/acceptance criteria
  + No feasible way to verify failure (eg, invalid .csv submitted gives no ui response)
  + Time consuming cases that can be verifed in other testcases (due to time constraint)

# Modules

* Population and Retrieval of records
* Tax Calculations
* Dispense
* Miscellaenous

## Population of Records

|  |  |
| --- | --- |
| F1 – (1) AC1: Single record of a working class hero should consist of Natural Id (natid), Name, Gender, Birthday, Salary and Tax paid | * Verify that single record insertion with all required valid information is successful   + success message   + new record shows up in hero list   + information appear in relevant columns (eg. NatId value to appear in NatId column) * Verify that insertion should not be successful if working class hero does not consist of any of the information below:   + Natural Id (natid)   + Name   + Gender   + Birthday   + Salary   + Tax paid * Verify that all fields requires valid values   + not “”   + data type?   + Special characters?   + Date format |
| F1 – (2) AC1: Enhancement of (1) , with the ability to insert a list | * Verify that multiple record insertion with all required information is successful   + success message   + record shows up in hero list   + information appear in relevant columns for the relevant heros * Verify that insertion should not be successful if any of the inserted working class hero does not consist of any of the information below (?):   + Natural Id (natid)   + Name   + Gender   + Birthday   + Salary   + Tax paid |
| F1 – (3) AC1: First row of the csv file must be natid, name, gender, salary, birthday, tax  F1 – (3) AC2: Subsequent rows of csv are the relevant details of each working class hero | * Verify that upload should fail if file is not in .csv format * Verify if upload is successful (success message) * Verify if that API will skip first row of csv and only capture second row onwards * Verify if data assertion is successful   + New records added   + Data displayed in table is accurate and corresponding to columns * Verify that upload should fail if file uploaded is not .csv * Verify that upload should fail if data provided is invalid   + not “”   + data type?   + Special characters?   + Date format |
| F1 - (3) AC3: A simple button that allows me to upload a file on my pc to the portal | * A simple button that allows me to upload a file on my pc to the portal   + Verify if button exist, is visible and enabled |
| F2 - AC1: a GET endpoint which returns a list consist of natid, tax relief amount and name  F2 - AC2: natid field must be masked from the 5th character onwards with dollar sign ‘$’ | * Verify if query returns list consisting of natid, tax relief amount and name * Verify if natId field values are masked from the 5th character onwards * Verify behavior if natId field has 4 characters |

## Tax Calculations

|  |  |
| --- | --- |
| F2 - AC3: computation of the tax relief is using the formula as described:    F2 - AC4: After calculating the tax relief amount, it should be subjected to normal rounding rule to remove any decimal places  F2 - AC5: If the calculated tax relief amount after subjecting to normal rounding rule is more than 0.00 but less than 50.00, the final tax  relief amount should be 50.00  F2 - AC6: If the calculated tax relief amount before applying the normal rounding rule gives a value with more than 2 decimal places, it should  be truncated at the second decimal place and then subjected to normal rounding rule | * Verify if tax relief displayed is accurate as described in the formula   + For age 17 (for at most 18)   + For age 34 (for at most 35)   + For age 49 (for at most 50)   + For age 74 (for at most 75)   + For age 77 (for at least 76) === Boundary   + For age 0   + For age 18 (for at most 18)   + For age 35 (for at most 35)   + For age 50 (for at most 50)   + For age 75 (for at most 75)   + For age 76 (for at least 76) ===   + For M, F for each of the above ages ===   + For single, multiple, csv insertion |

## Dispense

|  |  |
| --- | --- |
| F3 - AC1: The button on the screen must be red-colored  F3 - AC2: The text on the button must be exactly “Dispense Now”  F3 - AC3: After clicking on the button, it should direct me to a page with a text that says “Cash dispensed” | * Verify if button is available, visible and enabled * Verify if button is red with text “Dispense Now” * Verify is on click, user is directed to a page with text that says “Cash dispensed” |

## Miscellaenous

|  |  |
| --- | --- |
| “Total Tax Relieves  £3860.00 will be dispensed to 1 Working Class Hero/s” | * Verify if the total price and number of working class hero/s corresponds with Total Tax Relief. |
|  |  |

# Errors met:

|  |
| --- |
|  |
| * When no value(“”) is provided for name, natid and/or gender field. * When natId is lesser than 4 characters |

# Non-Functional Testing

Noticing that there was various scenarios where the system would display error, non-functional testing should be done to test the readiness of the system. Non-functional testing should increase usability, efficiency, maintainability and portability of the system. The following non-functional testing can be considered:

* Perfomance Testing
* Security Testing

## Performance Testing

Performance testing can be done to identify and eliminate bottlenecks in the application. The below items can be considered while testing:

* Response time
* Throughput or the number of transacions completed during a load test
* Process Time of activities like importing and exporting of csv files, or tax relief calculations
* Load Increment testing can be done to verify how much load the system can withstand

## Security Testing

Security Testing can be done to uncover vulnerabilities, threats and risk in the system. The below items can be considered while testing:

* Authentication – only real authentic usage of the system
* Authorization – only intended users to access the systems
* Timeout for inactivity