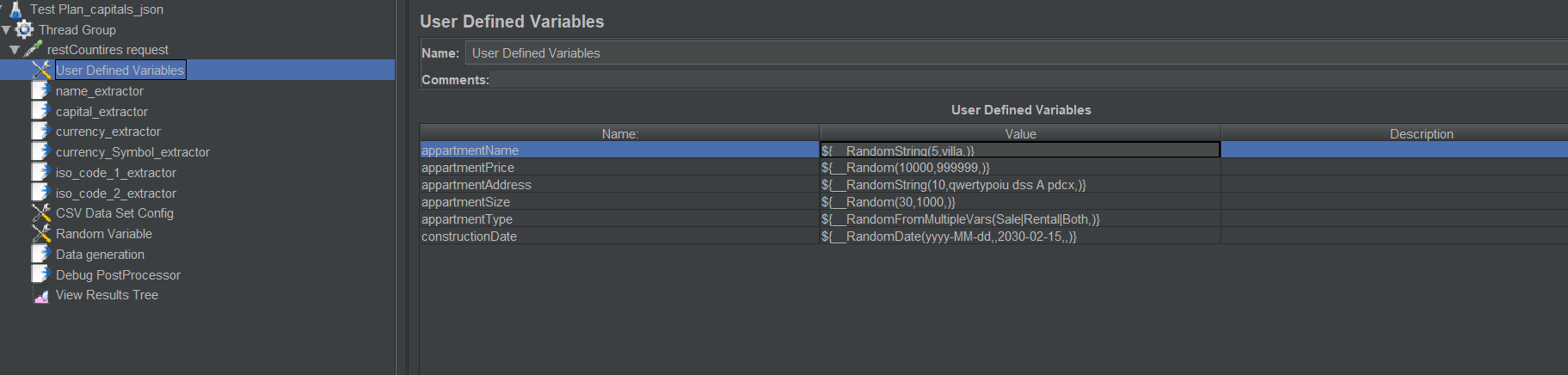
User Manual:

Although it has been clear to me that it’s up to me the implementation, I have chosen using Jmeter as a tool, as I think it’s an adaption of your company in terms of technology. Although, I would still need more time to get borader Jmeter knowledge to implement my approaches and ideas, that couldn’t figure out during the time working on this task and learning to use it at the same time (5 days).

So, this is my explanations to the work done:

1. The projects are dependent on each other, the approach is to have all of them on the same TestPlan, but I had the issue of not being able to introduce in Jmeter a delay time (tried with several delays, think times..), in order to make sure that the reading of a CSV file is made certainly after being generated and not before finishing it completely by the previous thread. In a real scenario with my current technologies I take special care of the preconditions and post conditions of every testcase, but would need to figure this point with a little more time with Jmeter
2. Random Variables with regex expressions. My approach was to use the Random Variable or even the Random functions at the User Defined variables, actually, the last one is used in the project:



Issues found: The values are kept static once generated as they are iterated within a beanshell script, not able to call to the Random function from inside, so, tried a “work around” so far, for updating their generated values over iterations… with more time, I would figure out how to make them take updated Random values with Jmeter tools.

1. Functions and modularizations for reusability:

I investigated how to perform my own functions in order to make them reusable as much as problems, but I still need more time, so much code is “harcoded” inside beanshell scripts that could be implemented inside functions.

1. Problems with Beanshell and JSR2233 porcessors…

I tried to use Java, but much of the functionalities in the shells seem need more compatibility with Java.

1. <https://fixer.io/> is very limited to the free plan, so much of the requests failed after short time, due to the limitation in the free plan. My solution is to login and get and new access key using different user account in each time.

Test Plan Explanation

Retrieve and store a list with all countries, their capital, currency and ISO code, using (but not limited) for example: <https://restcountries.eu/>

|  |  |
| --- | --- |
| **Jmeter project name** | **Generated data file** |
| allCountries\_generation.jmx | allCountries.csv |

Create a data generator, which builds on the data generated before. For each capital it needs to generate a dataset of randomised apartments with the following properties: “appartmentName”, “country” (according to the capital), “apartmentPrice” (up to 6 digits), “currency” (according to the country), “currencySymbol” (according to the country), “appartmentAdress” (strings and letters, no special characters), apartmentSize, appartmentType ( Sale, Rental, Both), constructionDate ( dateFormat). The data should be build using the common regex patterns

|  |  |
| --- | --- |
| **Jmeter project name** | **Generated data file** |
| appartmentPerCapitalGeneration.jmx | appartments\_country.csv |

Create a script (using jMeterBean Shell, Groovy, etc), which will obtain the following information:

Using API like <http://openweathermap.org/api>:

* for each apartment generate a list with the current weather according to the city.

|  |  |
| --- | --- |
| **Jmeter project name** | **Generated data file** |
| weatherPerCityGeneration.jmx | appart\_city\_weather.csv |

* for each apartment, which is in non-raining area and has “Rental” attribute, add new field “Recommended”

|  |  |
| --- | --- |
| **Jmeter project name** | **Generated data file** |
| appart\_city\_weather\_recommended.csv | recommended\_apartments.jmx |

* for each “apartment, which has “Sale” or “Both” attributes, add new field “Good Investment”.

|  |  |
| --- | --- |
| **Jmeter project name** | **Generated data file** |
| goodInvestment\_apartments.jmx | appart\_city\_goodInvestment.csv |

* **Using API( for example like https://fixer.io/), for each apartment convert the appartmentPrice value to corresponding amount in EUR and update the currency field as well. Add additional property “convertedFrom” and save there the original currency.**
* **If currency is not found by the RestApi, add “unableToConvert” in the “convertedFrom” property.**

|  |  |
| --- | --- |
| **Jmeter project name** | **Generated data file** |
| apartment\_price.jmx | appartment\_price\_conversion.csv |