First Task. Tool chosen – Apache Jmeter

Two sample responses were executed.

Fig 1. Sample successful response.

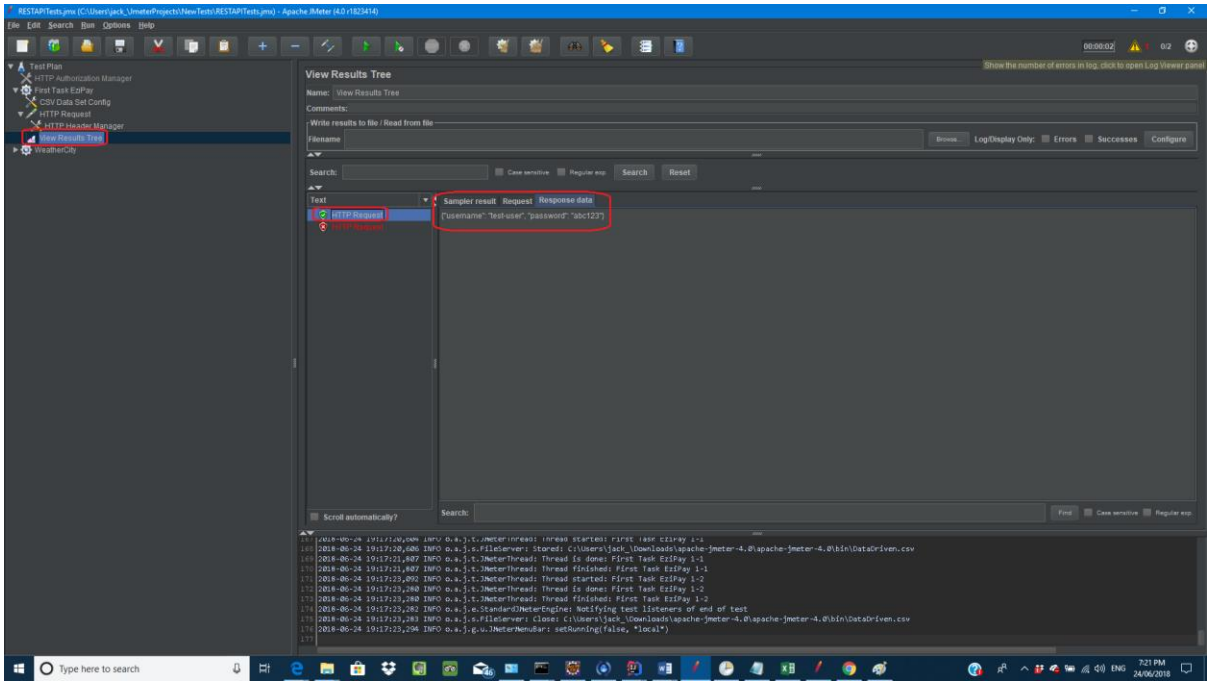
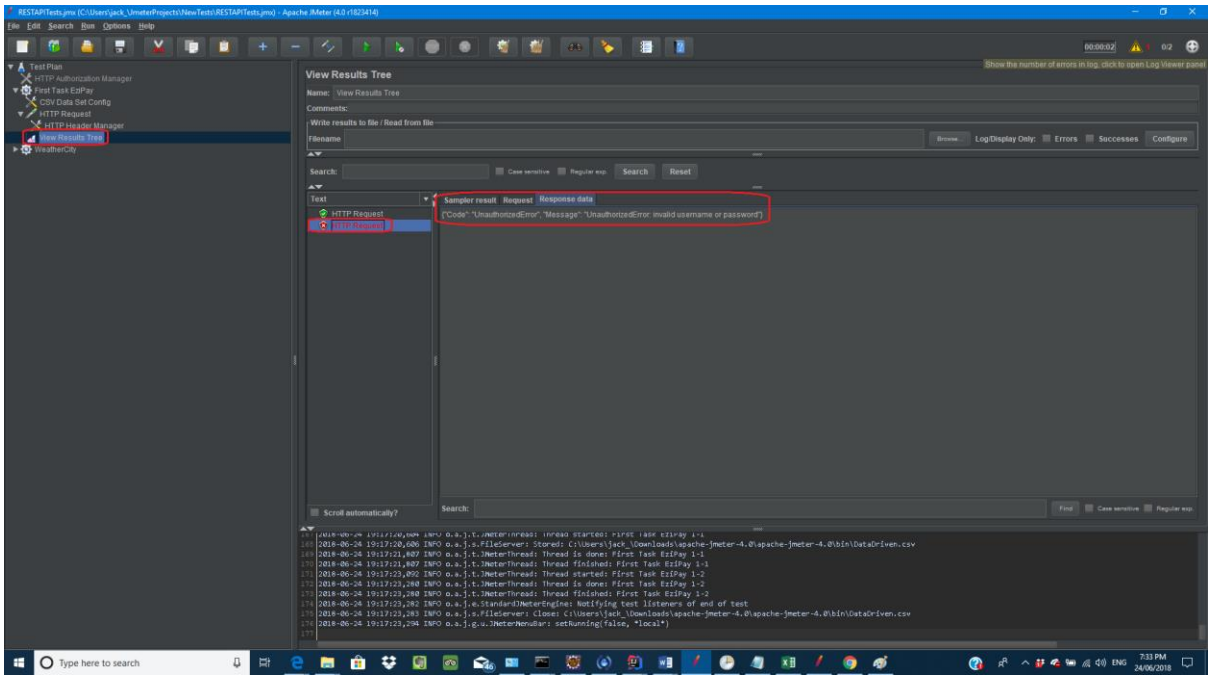
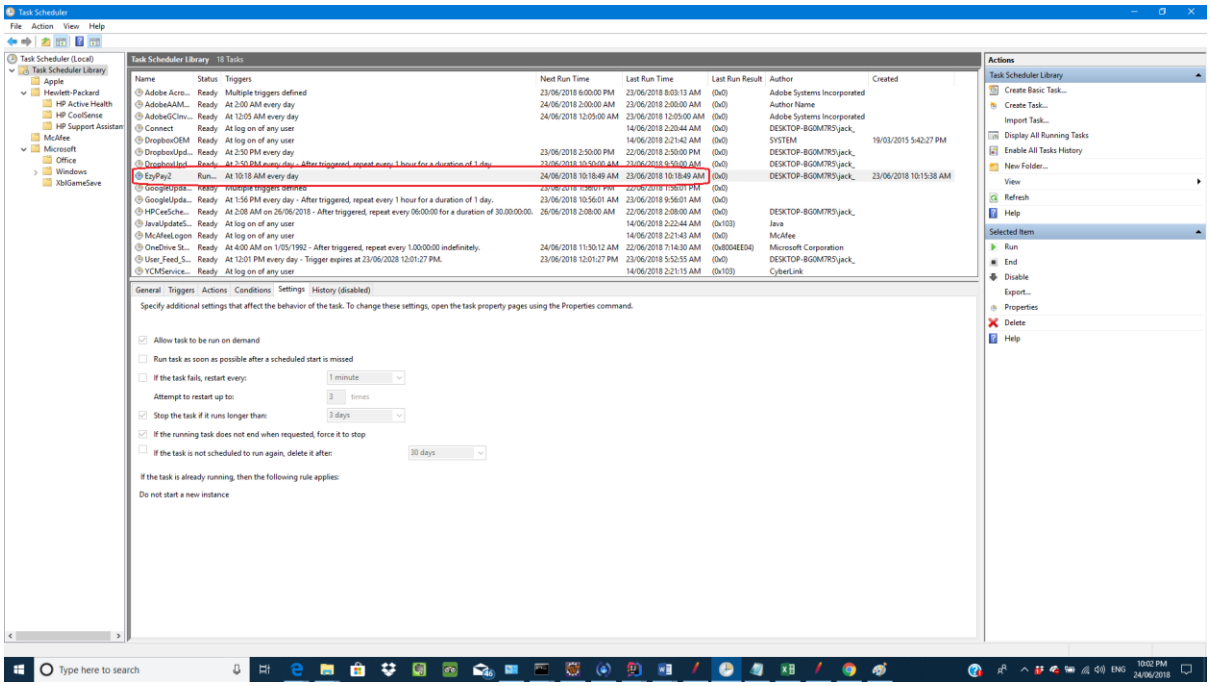


Fig 2. Sample failed response



Automation.

For the purpose of this first task, the automated test was executed using Windows task scheduler, however the same test can be scheduled using either Bamboo or Jenkins.



Second Task.

1. To scale my answer in the first task to cover an entire system, I have tweaked the post script to accept data driven test scenarios in a CSV file that sits outside the scripts (making it easier to maintain). For example, Fig3 & Fig 4 demonstrates on the first row of the csv file, we have the successful response, and on the second row of the csv file, we have the unsuccessful response. We would employ the same approach for all endpoints of the micro-services.

   Column A defines the IP Server name

   Column B defines the path

   Column C defines the username

   Column D defines the password

Fig 3.

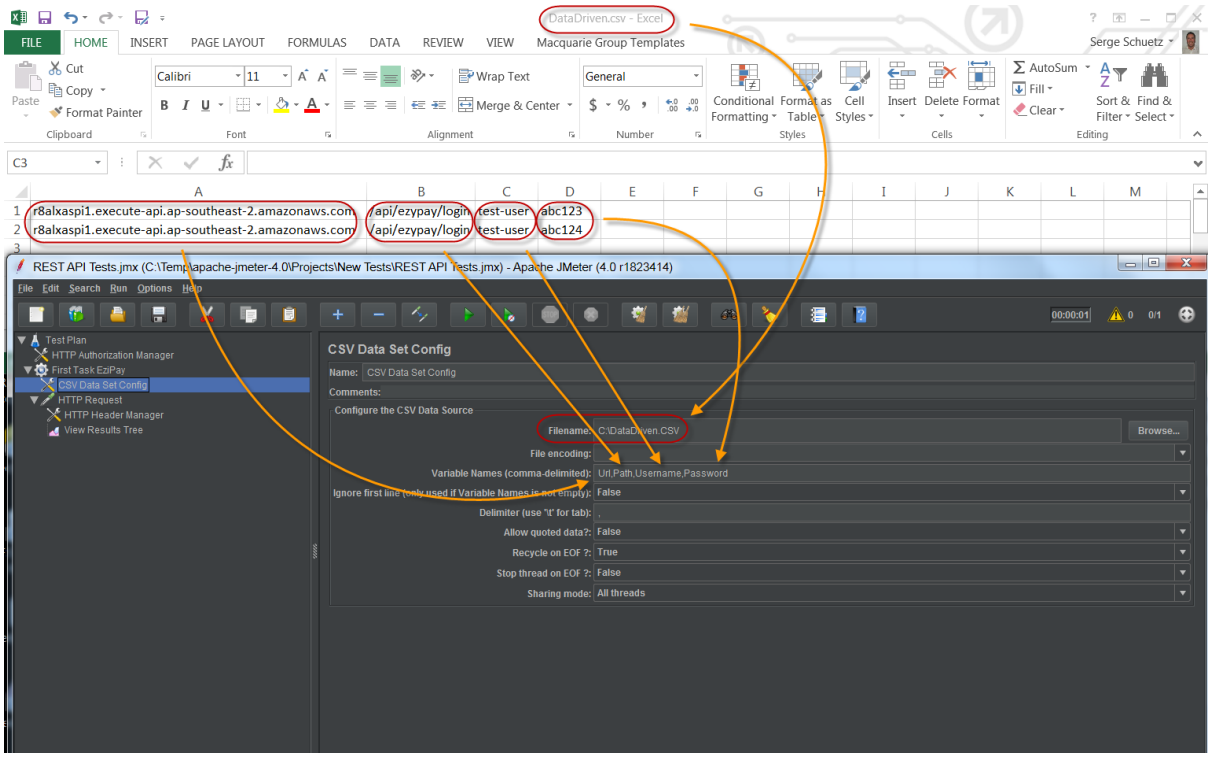
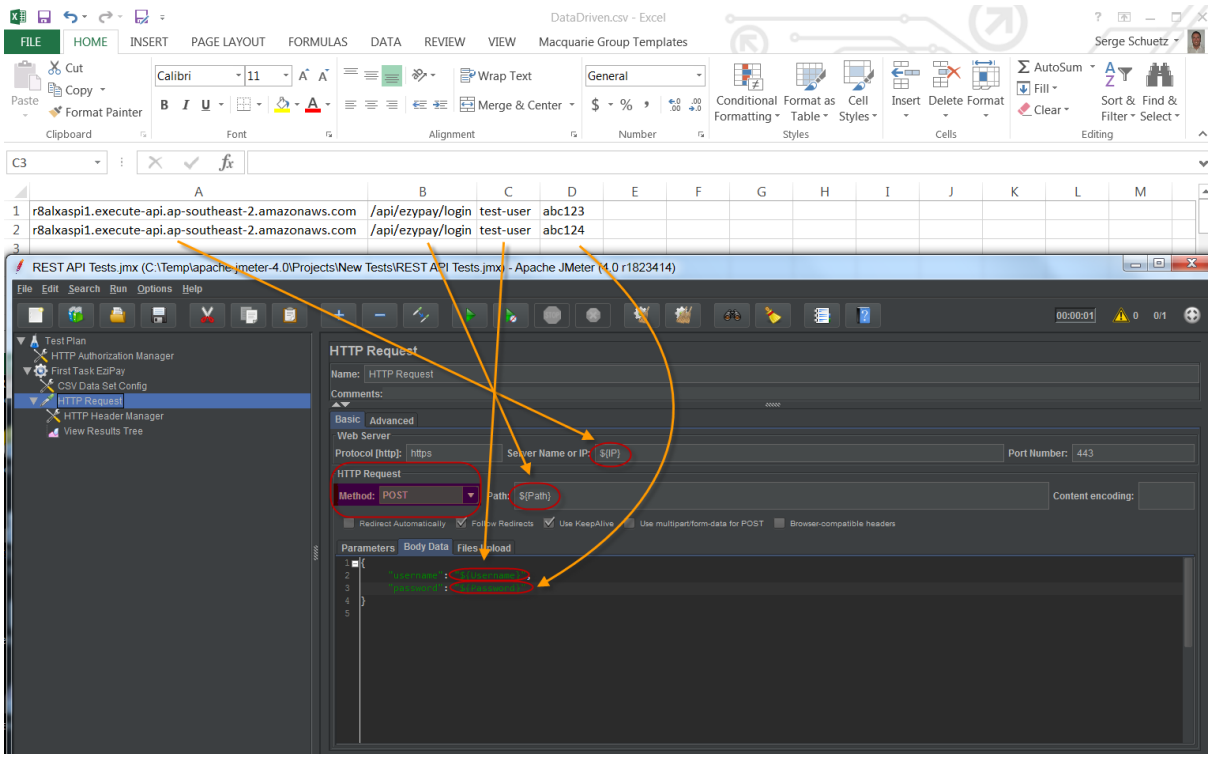


Fig 4



2. How much do you test and what scenarios?

Using the test strategy in 1, we are able to test more scenarios and consider wider coverage by enabling all you test scenarios in your csv file.

For example – the scenarios you could include in your file without needing to change the script are

- Verify the password contains at least 8 characters
- Verify the password contains at least one capital letter
- Verify the password contains at least one lower case letter
- Verify the password contains at least one number
- Verify the password contains at least one special character (you could specify specific special characters)
- Verify to use an email address as the username

Considerations

- What if the user has forgotten his/her password?
- What about password reset?
- We could be more specific, which is invalid, the username or the password?

3. The majority of the testing should be done at this lower level first, maximising data driven scenarios to test business logic APIs at the service layer. All the end points of the micro-services would be tested independently and then tested integrating with one another to get wider end-to-end coverage. This can be achieved by storing, any data required for other end points into variables.

In Fig 5, you can change the number of threads by the number of scenarios in your csv file. You can select them to run one at a time or even better, run them concurrently and execute your functional test to determine their performance.

Fig 5