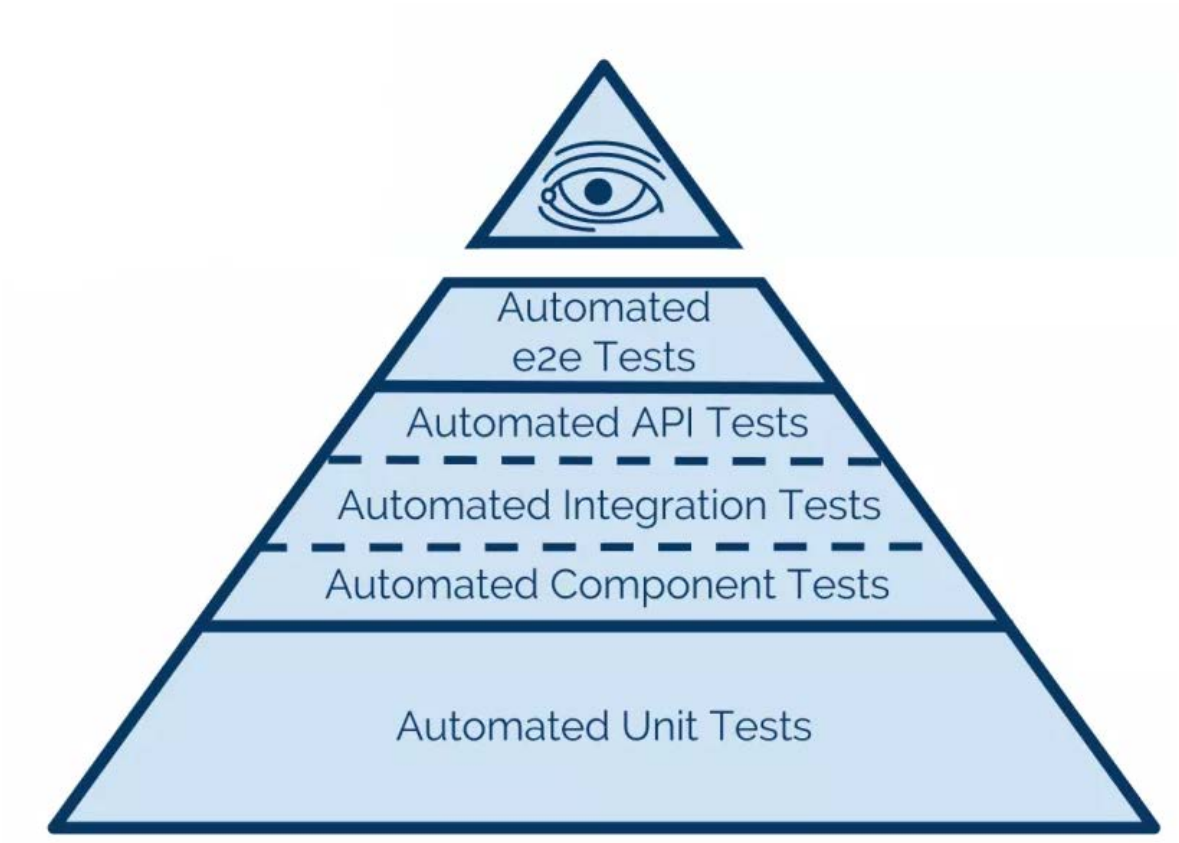
API test strategy

Test Strategy:

It is imperative to have a robust test approach to type of project. The test approach should be optimized to reduce defects and execution cycles i.e. implementing the following test pyramid as much as possible can be a good way to start.



The objective of our document is to design an optimal test strategy for API testing for Adam's Fried Chicken, we are concerned of the middle layer. Microservices should be able to operate independently hence we should have an environment where we can test the services independently (It can also be the developers machine). The following tests can be tested in the environment

i. Schema validation tests
ii. Different response status codes applicable to the service

The build that passes the above tests can be deployed to an environment where the downstream systems are stubbed wherever possible. This environment can be used to verify the transformation rules, validate the requests generated for downstream systems and whether the responses are processed as per the requirements. All the transformation requirements including any array mappings should be validated in this environment. The same environment is a good candidate

to be used for performance testing of APIs since the downstream is stubbed we will be able to measure the performance of the individual APIs.

The next environment is the environment which has all the live downstream systems and APIs. End to end scenarios should be run in this environment. The tests should precisely test end to end scenarios since extensive testing of the functionalities of individual APIs were tested thoroughly in the previous environments.

The automation framework used to test the login API can be scaled to have many APIs and tests. The advantage of having all the APIs in a single project is they can share common functions and data. The junit framework is designed using cucumber, hence it is possible to run tests of a particular API using the tag of that particular API or a tag for end to end test can be created and the end to end tests can be executed using the end to end tag. The maintenance of the project will be lot easier because of a single project and shared functions.