# Test Approach

Q4. Can you detail your Automation framework to automate an end –end scenarios from a web application to mobile application and vice versa.

Assume the scenario is that you have booked a cargo to deliver from Goa to Delhi using a front office web application and you have received a shipment number.

Now the delivery team has a mobile application using which they keep updating the status of the shipment as – Started from Source station, On Route, Delivered in Destination.

As the delivery team updates the status on the mobile app, this status should reflect in the web application when you track the shipment number as a user.

**Automation Framework**

1. **Tools**

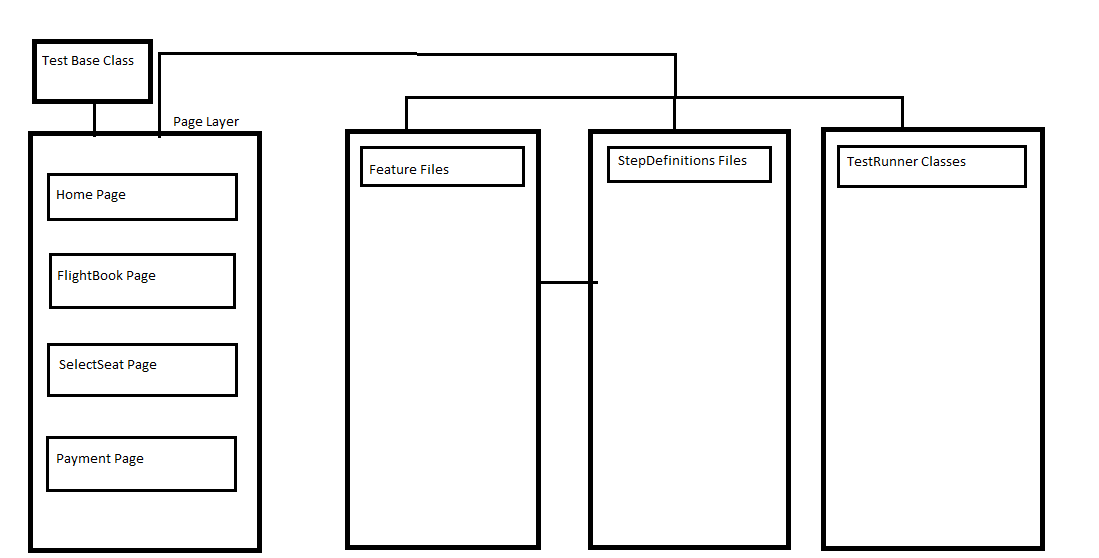
For the above Particular Scenario, An integration of Appium, WebDriver and API is required

1. WebDriver will be used to automate the scenarios on the web application.
2. Appium works on the top layer of the WebDriver, It will be used to automate the Testing on Mobile app test scenarios.
3. API classes will be used to communicate between the web app and mobile app.
4. GitHub will be used for version control and maintaining the repository.
5. Jenkins for Continuous integration.

**B. Flow of the Application**

* Web browser will be launched using WebDriver, when the web page will be loaded. Then the product will be searched on the search bar and the earch results will be displayed.
* The user will select the product and add it to the bag.
* The user then checkout from the bag, selects the shipping address and navigated to the Payment gateway to do the payments.
* After making the payment a shipment Id/Order Id will be generated and stored in the database with the product details.
* Then the mobile application will be launched via appium, which delivery team will be handling to update the status of the shipment.
* When the status will be changed via mobile app then the webservice at the backend will be triggered to update the new changed status in the web app and the database.
* Notification will be generated via webservices.

**The framework will be having:**



**Test Base Class**

Base class is the main class which will take care of Browser setup, loading [**configuration file**](http://learn-automation.com/object-repository-in-selenium-webdriver/) **, loading properties file** and other reusable methods like [**screenshot**](http://learn-automation.com/how-to-capture-screenshot-for-failed-test-cases-in-selenium-webdriver/) and many more.

The Page layer will be extending the base class for the reusability of code like launching browser and etc.

**Properties Files**

They will contain the urls for mobile app and web app, Browser details, userid/password for mocking and etc.

**Page Layer**

It contains the page classes of the application containing the web elements and methods related to that page only. All the functionalities/test will be written in this class.

**Feature Files**

All the test scenarios will be defined in the feature files. Each page/functionality will have its corresponding feature file.

**Step Definitions Class**

It will contain all the mapping of the steps in the feature file and the corresponding method calls on the corresponding page layer.

**Test Runner**

Test Runner classes will be used for executing the test scenarios and providing the cucumber options like for reporting and etc.