Last 2 projects which ran almost for close to 2.6 Years, those projects were,

* GSP SABB (Saudi Arabia British Bank) - This was a complete "Personal internet banking project"
* GDOCS (Global documentation for HSBC Bank) - This was a client on-boarding platform for clients who opted for various products related to CASH. Example of one such product is "Sweep" where let say if we have bank accounts to multiple countries then all cash will from those accounts will be deposited in the let say kalyani nagar branch.
* Backend solution was divided into 3 microservices (Party, Product and Case)...and these microservices had their own 3 dedicated databases (Party DB, Product DB and Case DB).

Given : <Some Condition>  
When : <action>  
Then : <Expected output>

Example of above is,

Given : API url is http://api.github.com  
When : User sends a GET request  
Then : Response status code is 200

Helped team in creating small testing framework in Java for testing of web api / restful api. This was done using famous Apache library, Apache HttpClient.

Q. What all you have done as part of automation testing and in general testing ?  
Ans. Below are the things done by me on testing front and also as part of automation testing,

* Write a small testing framework in Java for testing of web api / restful api. This was done using Apache library HttpClient and also we used TestNG.
* Helped team creating a testing database in MongoDB. That particular database was containing only test data. Later i took dump of testing database and checked in inside GIT for later use
* Created stubs to simulate backend response (Mainframe backend) for load testing of API in DPE environment.
* Once stubs were ready and load testing was done, memory leaked was found in one of war (It was MoveMoney.war).
* Integrated application with YourKit and did local load testing with the help of standalone Java application to find the class and method responsible for memory leak. We found the responsible class and methods and fixed the issue.
* Performance Testing of "dashboard load slow" problem. End to end analysis of response time of all services involved used in loading dashboard in browser. To analyze we started from doing "HAR file analysis" on chrome browser. HAR file analysis gives all the ajax call happening from client and which in turn was triggering various backend services. After rigrous analysis it was found that it was size of background image which was much greater than other tenants causing performance issue. Used AppDynamics to find response time of each node and tier involved.
* Did massive testing of data migration. This testing was done for the validation of migrated telecom data. Main testing was to be done for,
  + - Subscribers (i.e Customers)
    - Services of Subscribers (Diffetent services for Prepaid and Postpaid Customers)
    - Equipments of Customers (MSISDN, SIM)
    - VAS (Value added services).
  + Basically there are 2 different type of services,
  + Core services,
  + Value added services