| **1.Git Conflict** | **Resolved** |
| --- | --- |
| When we did our first push,pull and merge, there was lots of conflict even though everyone had same format of java classes and files, the issue is because of different “JAVA COMPILER” | We changed all our java compiler to same version |

| 2. **Aws DynamoDB** | **Resolved** |
| --- | --- |
| Diet table is present in NoSqlWorkBench, but it not displayed while using aws list of tables command and not able to access the data present in the table (using postman) | Inside the DynamoDbLocal\_lib folder, there will be numerous “.db” files, remove all or move all those “.db” files to some other folder |

| 3. **Running Java Developed API** | **Resolved** |
| --- | --- |
| API is not displaying tables, when we use the get recipes and user method in both postman and swagger. The response body is empty | We removed the “@Getter” and “@Setter” methods in Java API code and changed to POJO class. Because Lombok was not working as expected. |

| 4.**Java API** | **Trouble** |
| --- | --- |
| Endpoint in the Java API was not working as expected. | This issue can be resolved by the developer's side. |

| 5. **Running Python Developer Code** | **Resolved** |
| --- | --- |
| Virtual environment was not created | Virtual Environment is supported only for Python versions 3 and above. So we upgraded to python version 3. |

| **6. Basic Test cases are getting failed** | **Resolved** |
| --- | --- |
| Unit tests are not followed properly during development. As a result, testers encountered difficulties performing basic testing . | We did a lot of manual testing to identify the issues in both the java and python . |

| **7.Implementing Endpoint parameter** | **Resolved** |
| --- | --- |
| We are unable to identify some of the endpoint parameters.  (path variable, path param, query param, form param.) | We formed the param string and concatenate it with the baseurl. We noticed that the endpoint behaved similarly to a formparam. However, when the content type is JSON, form parameters are not recommended. |

| **8.Batch payload testing** | **Trouble** |
| --- | --- |
| We used an array of objects to implement batch payload | The application post & put methods are not supporting an array of objects in the request body. It is designed in such a way |

| **9. Code** | **Resolved** |
| --- | --- |
| In API Invalid data is also giving **200 OK** Status , So it was very difficult to design an optimized framework. | We managed to optimize through minimal usage of IF - ELSE condition |

| **10. Json Response Body** | **Resolved** |
| --- | --- |
| In API , message fields for error codes keep on changing from Lowercase to UpperCase , sometimes in short form (msg). Validating Json's body was tricky. | We solved it by using the hamcrest core matcher method. |