**ProgressSoft Component Test Plan**

*CQ#:*

|  |  |  |  |
| --- | --- | --- | --- |
| Document Name | UTP.doc | Version.Rev | 1.0 |
| Prepared By | Sadhika,Sadhika | Date: | 10-18-2017 |
| Reviewed By | ProgressSoft Development Team | Date: | 10-18-2017 |
| Test Plan Results Reviewed By | ProgressSoft Development Team | Date: | 10-18-2017 |

# Description of the Project:

|  |  |
| --- | --- |
| **Project** | **Clustered Data Warehouse** |
| **Detail description** | Suppose you are part of a scrum team developing data warehouse for Bloomberg to analyze FX deals. One of customer stories is to import deals details from files into DB. The requested performance is to be able to import the file containing 100,000 records in less than 5 seconds.  File format is CSV contains the following fields (Deal Unique Id, From Currency ISO Code "Ordering Currency", To Currency ISO Code, Deal timestamp, Deal Amount in ordering currency).  Validate row structure.  Valid rows should be stored in table/document, with reference to source file name.  Invalid rows should be stored into another table/document, with reference to source file name.  The DB contains another table to maintain accumulative count of deals per Ordering Currency "Columns : Currency ISO Code, CountOfDeals ", so upon completion of importing process the system should increase count of deals per currency.  System should not import same file twice.  No rollback allowed, what every rows imported should be saved in DB. |
| **Technology Used** | SpringBoot 4, Hibernate 4, MySQL 5, and java 8 |

# Test Report:

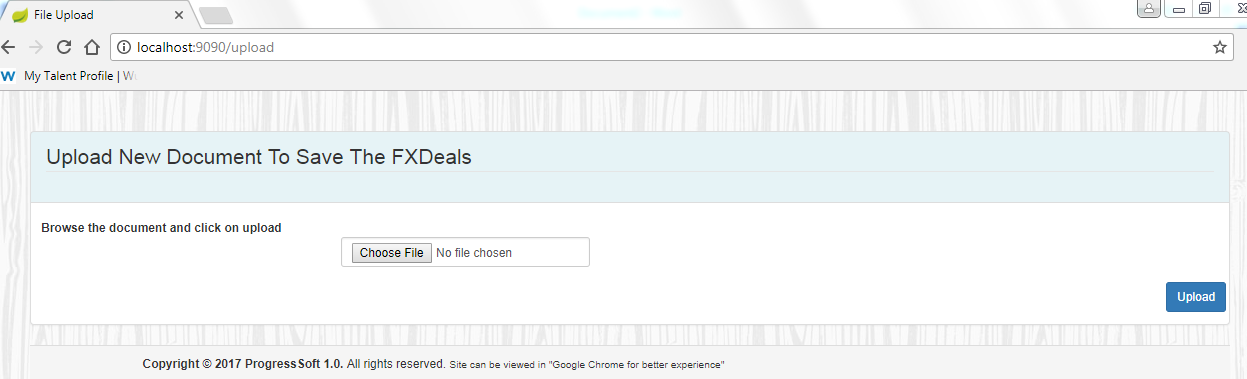
|  |  |
| --- | --- |
| **Author** | Sadhika,Sadhika |
| **Tested By** | Sadhika,Sadhika |
| **Test execution date** | **10/18/2017** |

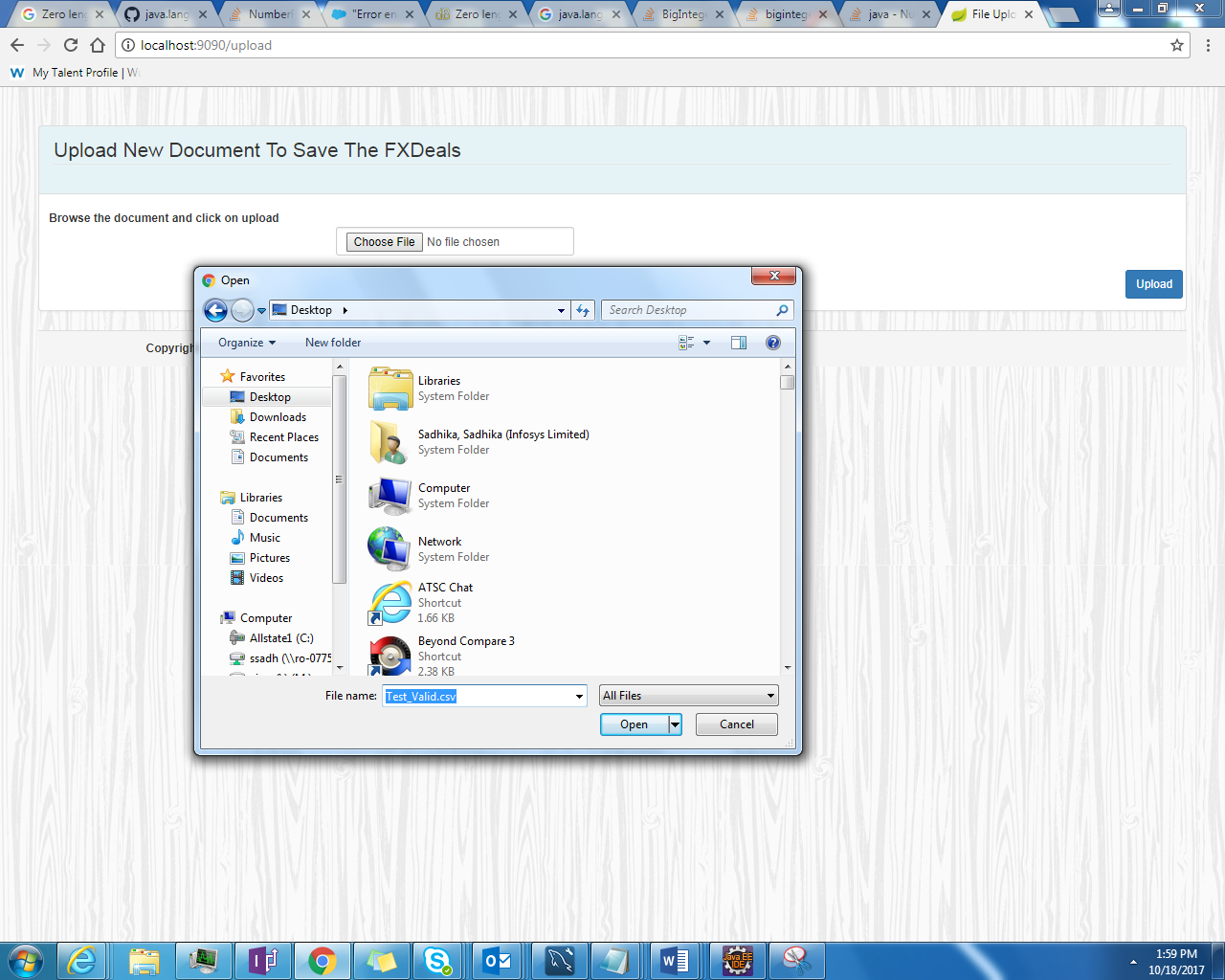
# Test Scenarios

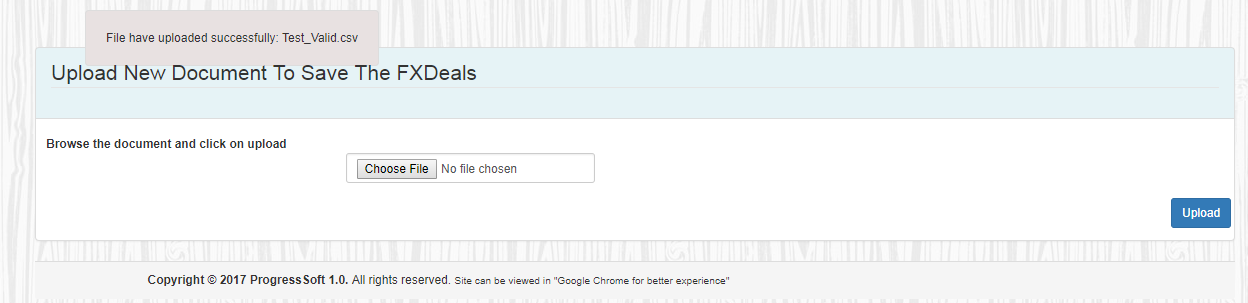
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **S No.** | **Test Step Description** | **Input Data (Reference)** | **Expected Result** | **Test Result** | | **Remarks** |
| **Pass/Fail** | **Cycle** |
| 1. | 1. Start the application by clicking “Run as – Java Application” 2. Once the application is started run the application by - <http://localhost:8099/upload> 3. Select a CSV File and click on upload button. | <http://localhost:9090/upload> | Data of the file should be successfully uploaded to database and message “File have uploaded successfully: ‘File name’ should be displayed | Pass | 1 | Executed Successfully |
| 2. | 1. Start the application by clicking “Run as – Java Application” 2. Once the application is started run the application by - <http://localhost:8099/upload> 3. Select a same File which was uploaded in case 1 and click on upload button. | <http://localhost:9090/upload> | Data of the file should not be uploaded to database and message- “"File already exists, please don’t make a duplicate insert” should be displayed | Pass | 1 | Executed Successfully |
| 3. | 1. Start the application by clicking “Run as – Java Application” 2. Once the application is started run the application by - <http://localhost:8099/upload> 3. Do not select any file and click on upload button. | <http://localhost:8099/upload> | Data of the file should not be uploaded to database and message- “No file found, Please select the file to be uploaded” should be displayed | Pass | 1 | Executed Successfully |

# References

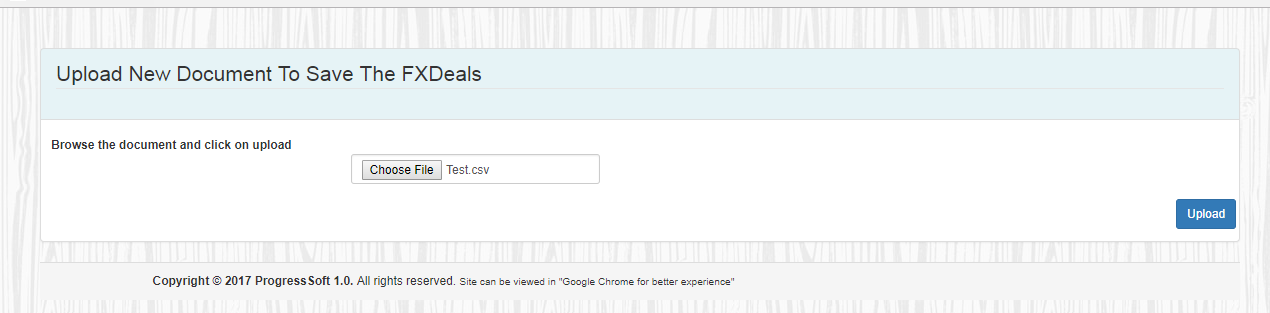
# Case 1: Insert Valid Data

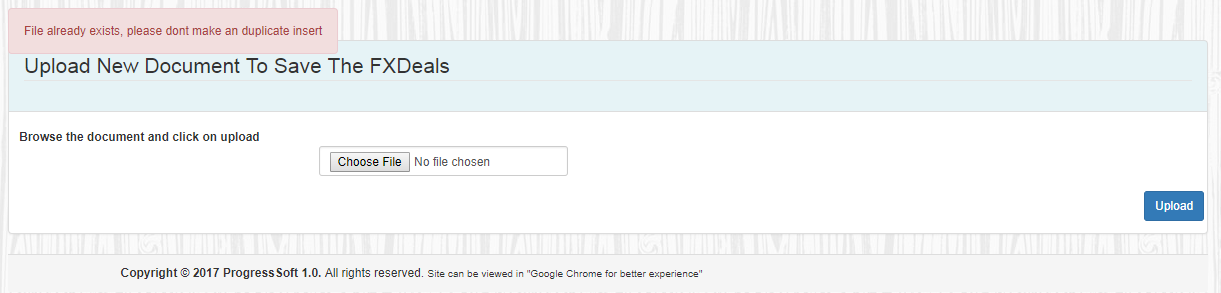




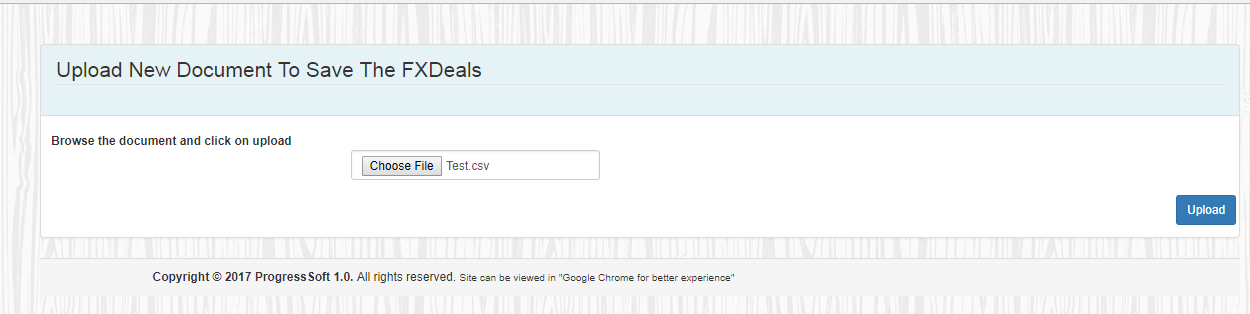


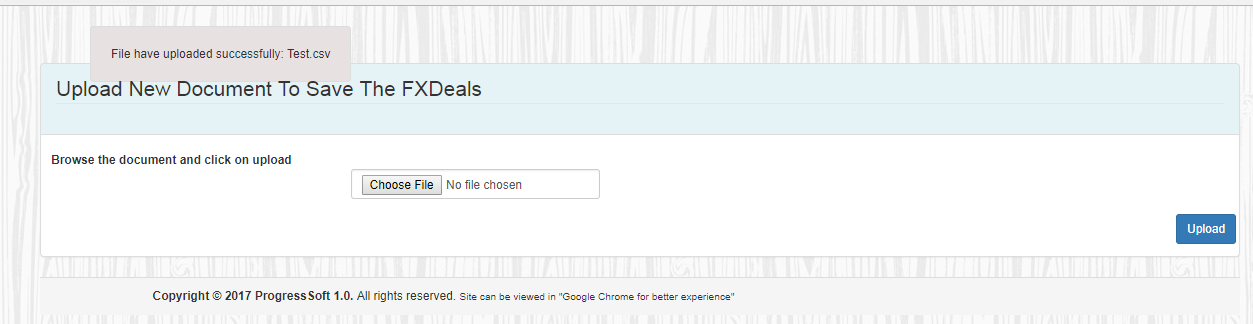
# Case 2: File already exists

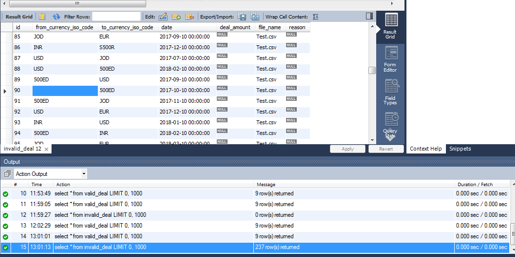




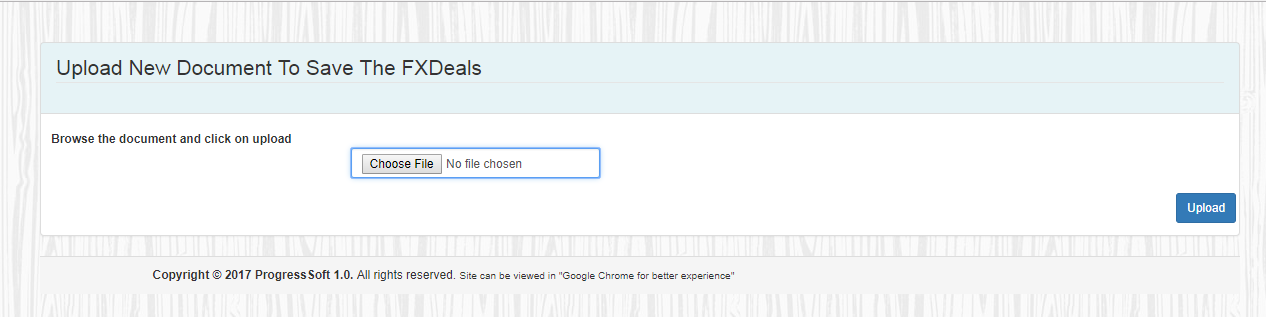
# Case 3: Insert Invalid Data

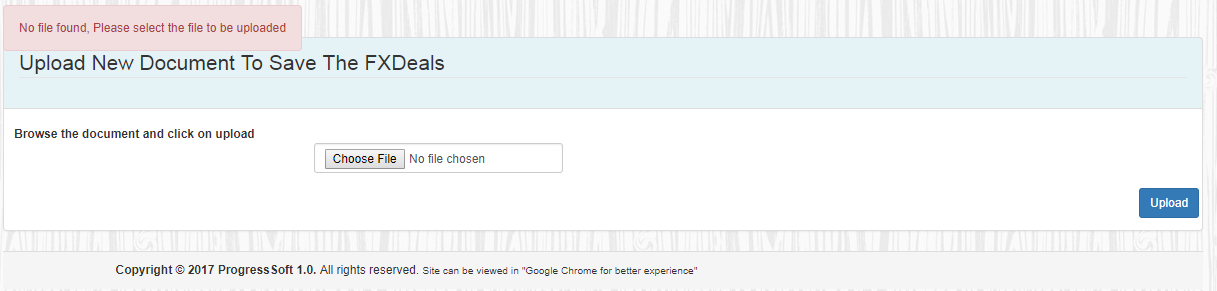






# Case 4: No file found





# Case 5: Count deals

