|  |
| --- |
| **Sasken Technologies Ltd.**  **KenTest**  **Front End Web Application**  **UI Mockups V 0.1** |
| PID: 01-01602 |
| Version: 0.1 |
| Date: 11-May-2020  Approved By:<> |
| **Solutions- Engineering R&D Practice** |
| **Sasken Technologies Ltd.** |

**Document Control**

|  |  |
| --- | --- |
| The authorized version of this document is an electronic master stored in the Document Repository (https://kenpoint.sasken.com/OE/SMS/default.aspx).  Be aware that if you are reading a hard copy of this document, it is to be considered an uncontrolled. It is advised that the version of the document in the repository be matched with the hard copy before using it. | The information contained in this document is proprietary to Sasken Technologies Ltd.  ©All rights to this information are reserved.  Disclosure without authorizing is prohibited. |

Template used: <>

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision History** | | | |
| **Version** | **Approval Date** | **Description of change(s)** | **Author(s)** |
| 0.1 |  | Initial Draft | Devaraj G R  Kajal  Bhavani  Karthikeyan Thenraj  Dinesh Thirunarayanan |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1 Purpose 4

2 Scope 4

2.1 Terms and Definitions 4

2.2 References 4

3 Background – KenTest Core Framework 5

3.1 Background 5

3.2 KenTest Folder Structure 5

3.3 Keyword based Test Script – sample 9

3.4 Session File – sample 10

3.5 Executing Tests 10

4 KenTest Web App UI Mockups 10

4.1 KenTest UI Screens classification based on features 10

4.2 Dashboard (Home Page Screen) 12

4.3 Test Script Management 13

4.4 Test Execution 18

5 Others 19

5.1 Limitations 19

5.2 Deviations 19

6 Open Issues <Optional> 19

1. Purpose

This document describes the UI Mockups for the front-end web application of KenTest, as per the User Stories defined in KenTest\_UI\_Userstories.xlxs.

1. Scope

KenTest is Sasken’s test automation framework that provides a scalable and sustainable automation solution. KenTest is build leveraging the capabilities of opensource tools for test automation​. Enables creating modular test suites and remote execution of test scripts against device connected on a Sasken custom hosted remote server (AKA Device Test Farm)​. Test can be executed from a centralized location while test servers can be at customer labs across geographical locations​.

Some of the key features are: -

* Support automated testing on Android mobile, Android IVI Device - System and Applications
* Supports automated testing of iOS Application
* Remote Test Automation using Sasken Customized Automation server
* Remote Manual Testing (Remote Access to Real Devices via the web dashboard)
* Multi device testing with Parallel execution

Currently scripts are executed from command line and we intent to develop a web-based application interface for

* Remote Device access/Control /Management
* Remote Device Manual Test and Debug
* Dashboard with Test Status, info( live and history data) etc.
* Test Scripts and other files Management
* Test Execution and Scheduling
* Viewing Test burn down charts, logs and reports
* Other Setting and config

This <work -in progress>document covers the UI Mockups for the front-end web application of KenTest as per the User Stories defined in KenTest\_UI\_Userstories.xlxs.

Intended audiences for the document are:

Team Working on this Project; Web Application Developers

Team, which will plan and execute the Integration Testing

* 1. Terms and Definitions

|  |  |
| --- | --- |
| Abbreviation/Term | Expansion/Definition |
| UI | User Interface |
| IVI | In-Vehicle Infotainment |
|  |  |

* 1. References

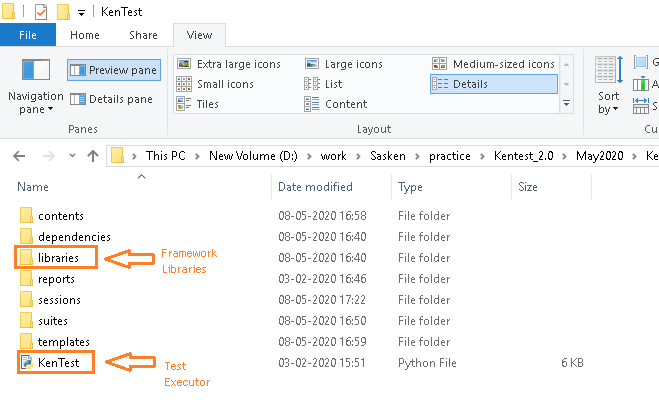
|  |  |
| --- | --- |
| **Document Name** | **Link/URL** |
| KenTest\_UI\_Userstories.xlxs |  |
|  |  |
|  |  |

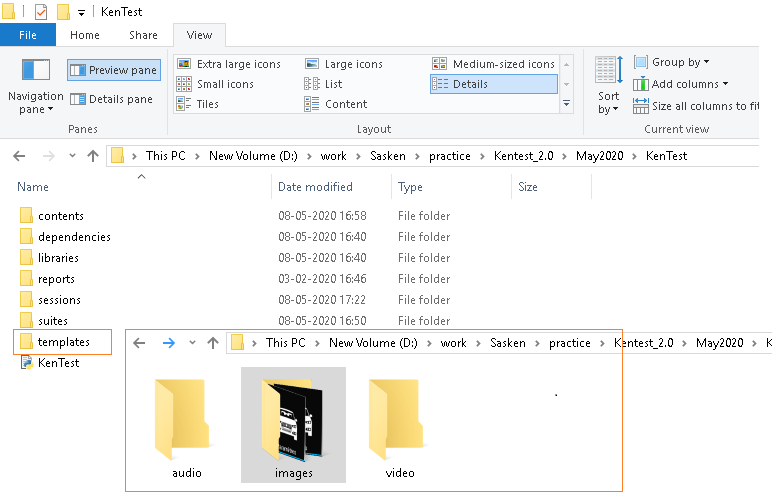
1. Background – KenTest Core Framework
   1. Background

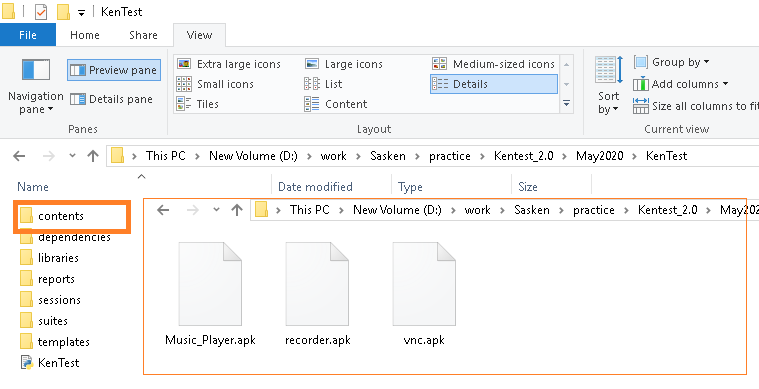
KenTest Engine consist of

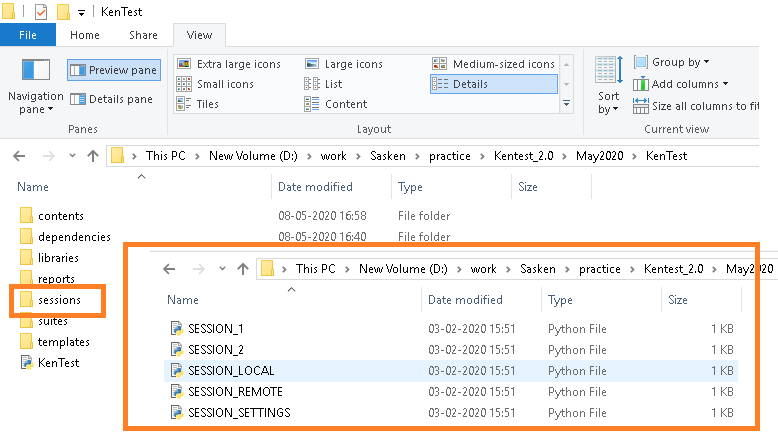
|  |  |
| --- | --- |
| Item | Description |
| Test Executor | Python file |
| Test Scripts | Robot Files |
| Contents | Support APKs |
| Templates | Images and videos |
| Session File | Python or JSON bases config files |
| Framework Libraries | Python files |

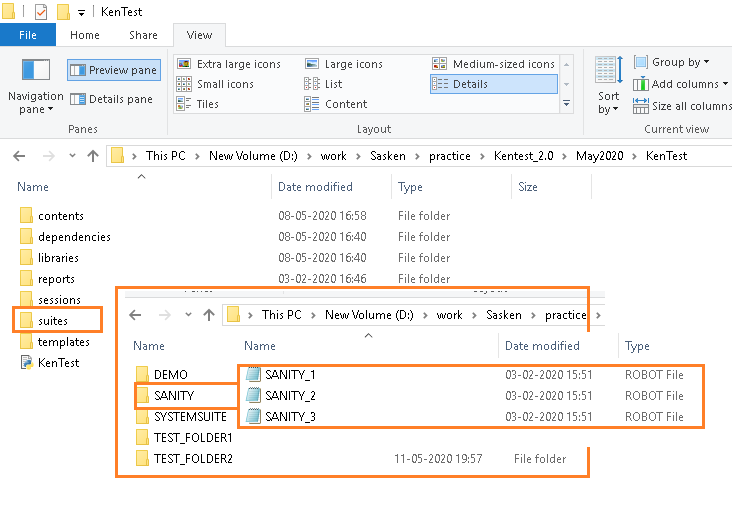
* Robot based Test Scripts are written and used for Testing​
* Set of Test Cases are functionally grouped in Robot file. This file is called a Test suite​
* A set of Files grouped into a Folder is  called a collection of Test Suites​
* A session file is a config  containing the details of the device against which the tests are executed
  1. KenTest Folder Structure











* 1. Keyword based Test Script – sample

**Test\_Contacts\_App.robot**

\*\*\* Settings \*\*\*  
Suite Setup Suite Setup  
Suite Teardown Suite Teardown  
Test Teardown Test Teardown  
Library AndroidGeneric.py  
  
  
\*\*\* Variables \*\*\*  
${NAME} Alvin  
${NUMBER} 9999999999  
${NEW\_NAME} Simon  
${NEW\_NUMBER} 8888888888  
${LINK\_NAME} Theodore  
  
\*\*\* Test Cases \*\*\*  
TC\_CREATE\_CONTACT  
 Add Contact ${NAME} ${NUMBER}  
   
TC\_EDIT\_CONTACT\_NAME  
 Edit Contact Name ${NAME} ${NEW\_NAME}  
  
TC\_EDIT\_CONTACT\_NUMBER  
 Edit Contact Number ${NEW\_NAME} ${NUMBER} ${NEW\_NUMBER}  
   
TC\_SET\_AS\_FAVORITE  
 Set Contact As Favorite ${NEW\_NAME}

TC\_LINK\_CONTACT  
 Create And Link Contact ${NEW\_NAME} ${LINK\_NAME}  
   
TC\_CALL\_CONTACT  
 Make Call To Contact ${NEW\_NAME}

\*\*\* Keywords \*\*\*  
Suite Setup  
   Print Message    SUITE SETUP HAS STARTED  
   Log To Console    -----------------------------------------------------------  
   Print Message    Creating driver for device 1.  
   &{CAPABILITIES\_1}    Create Dictionary    udid=${DEVICE\_1\_CAP}  
   &{DRIVER\_1}    Create Android Driver    ${APPIUM\_HOST\_1}    &{CAPABILITIES\_1}  
   Set Suite Variable    &{DRIVER\_1}  
   @{ARGS}=    Create List    DemoContacts.py    Test    ${DRIVER\_1}  
   Import Suite Library    ${ARGS}  
   Print Message    \n  
  
Suite Teardown  
   Run Keyword And Ignore Error    Quit Driver    &{DRIVER\_1}  
  
Test Teardown  
   Run Keyword If Test Failed      Capture Screenshot And Logs    &{DRIVER\_1}  
   Move To Home    &{DRIVER\_1}  
   Log To Console    -----------------------------------------------------------

* 1. Session File – sample

Session1.py

REMOTE\_HOST\_1 = 'http://127.0.0.1:4723/wd/hub’

DEVICE\_1\_CAP = {'serial': 'BH9000CF81', 'model': '', 'platform': '', 'version': ‘’}

STF\_ADDRESS = 'https://10.1.101.69’

TOKEN = '83bb539592444805865a37d4d1e1ea0dcd3a90752ab54880a47db79eaf537eea'

* 1. Executing Tests

Following table list a few commands that can be used to execute testing using the framework from command line

|  |  |
| --- | --- |
| SL.No. | Command |
| 1 | Python KenTest.py –SU SUITE\_NAME –SE SESSION\_FILE |
| 2 | Python KenTest.py -SU DEMO\DEMO\_SETTINGS -SE SESSION\_SETTINGS |
| 3 | Python KenTest.py -SU TEST\_FOLDER1\TEST\_1 TEST\_FOLDER2\TEST\_1 -SE SESSION\_1 |
| 4 | Python KenTest.py –FO SANITY -SE SESSION\_3 |
| 5 | Python KenTest.py –SU SUITE\_NAME –TG TAG\_1 TAG\_2 –SE SESSION\_FILE |

1. KenTest Web App UI Mockups
   1. KenTest UI Screens classification based on features

KenTest web app must have the following classified items

1. **Dashboard (Home Page Screen)**
2. Dashboard to display daily test execution graph and statistics
3. and circular/progress view of the existing total scripts in the test bed(scripts overview)
4. Dashboard to display the count of Success, Failure and Total scripts execution.
5. Common header (Page info and company logo) and footer (Copyright info and contact details etc.,).
6. More to be defined
7. **Test Script Management**
8. Suite Management
   1. Create/update (upload) /Edit/Delete/search a Suite File.
   2. Create/search and Delete a suite folder
9. Session File Management
   1. Create/update (upload) /Edit/Delete/search a Session File.
10. Template Management (Images/video)
    1. Create/update (upload) /Edit/Delete/search a Template File.
    2. Create/search and Delete a suite folder
11. Contents Management (APKs/PDFs/any other)
    1. Create/update (upload) /Edit/Delete/search a Content File.
    2. Create/search and Delete a suite folder
12. User roles for test scripts management.
13. Merge from Git.(Advanced)
14. More to be defined
15. **Test Execution**
16. Load one or multiple Suite/s
17. Load one or multiple session file/s
18. Select one and/or multiple devices.
19. Execute the test and see the logs streaming
20. Save/export the logs to files or DB
21. Stream the logs and cache it in viewer window
22. **View Logs and Reports**
23. View the logs and reports list (Table view providing searching, sorting and filtering options)
24. Provide view option to view detailed logs and report.
25. Delete the logs and reports.
26. Export logs as PDF or text files. etc.,
27. Email the logs and reports.
28. Download the logs and reports.
29. Segregate the logs into success and failure logs.
30. **Settings**
31. Settings is extra feature to enable the user profile related activities.
32. Change password or Forgot password.
33. Profile update activities.
    1. Dashboard (Home Page Screen)

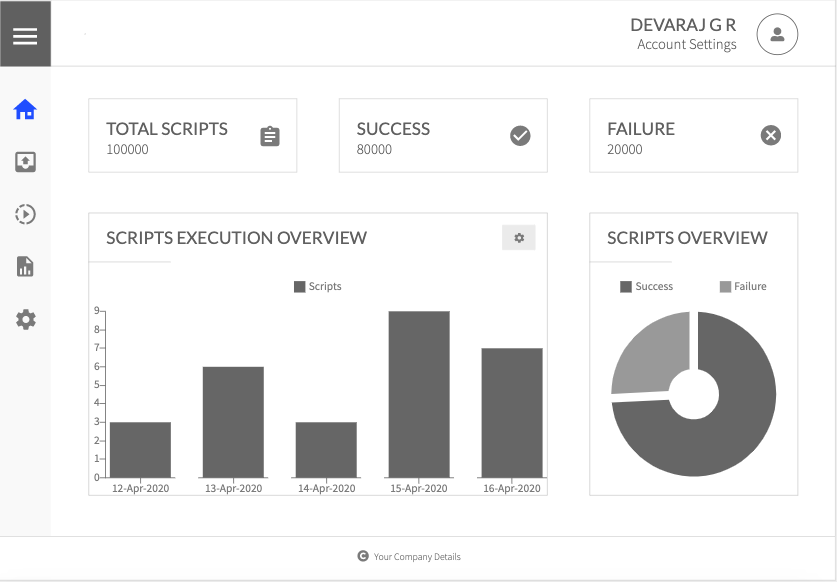


Figure 1: Dashboard to display overall test execution analytics

* 1. Test Script Management

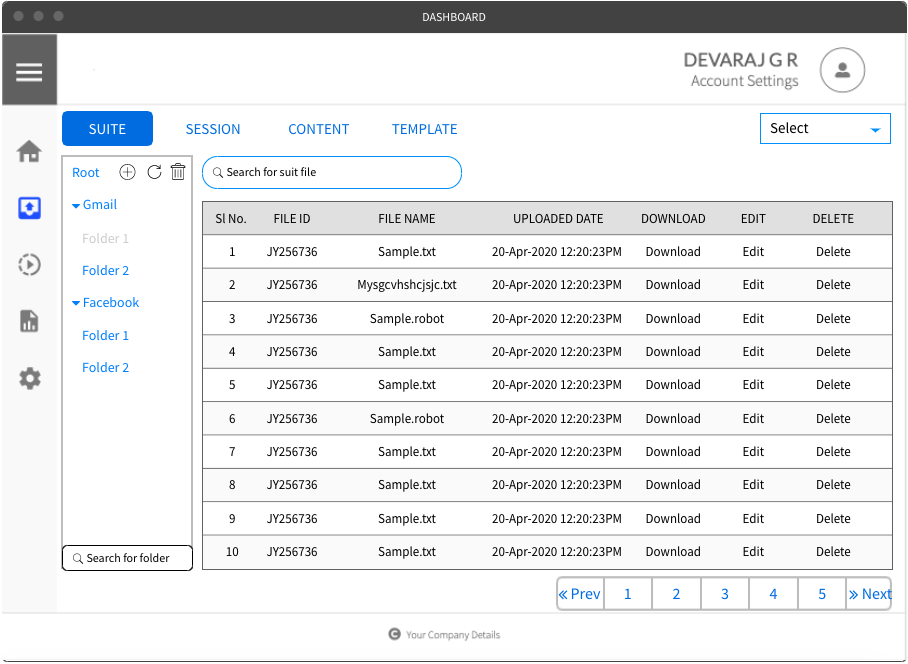


Figure 2: Suite files management

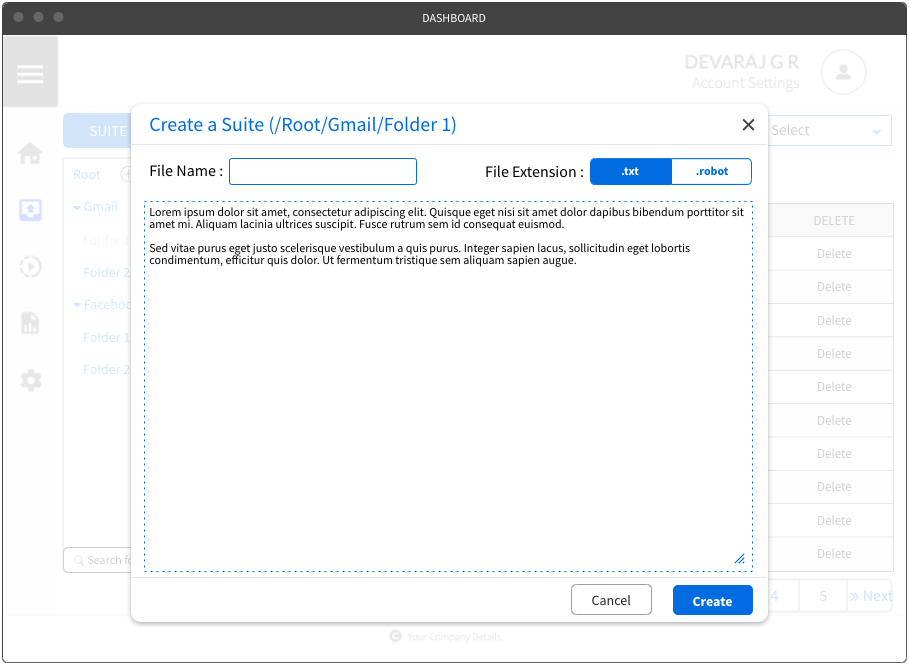


Figure 3: Create a suite file

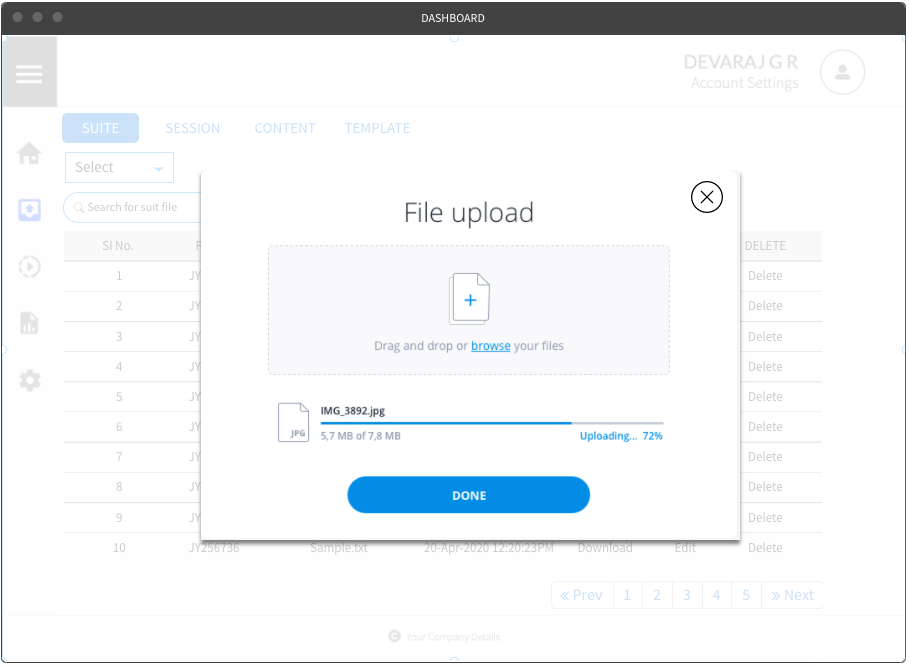


Figure 4: Upload a suite file

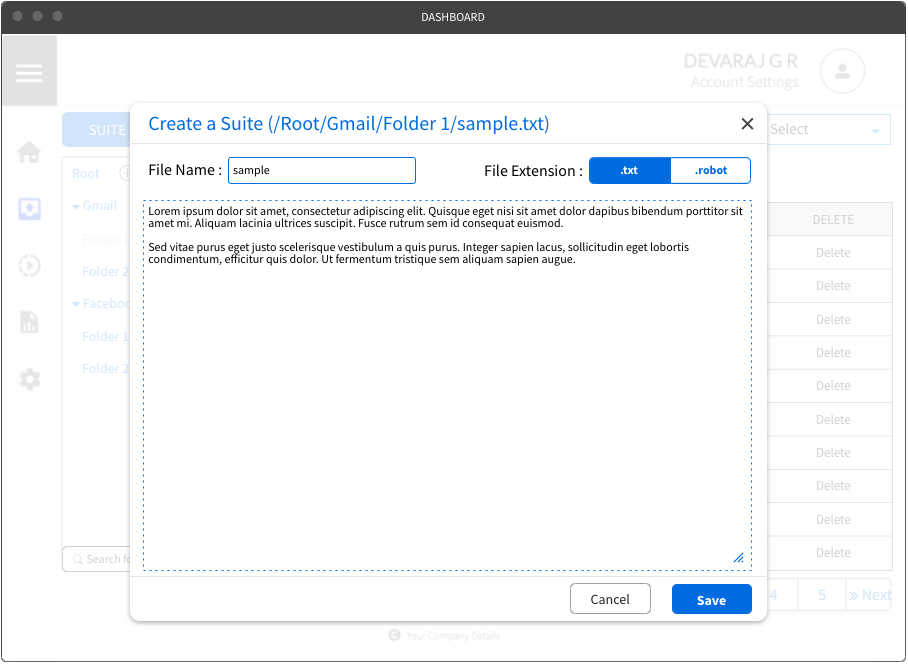


Figure 5: Edit a suite file

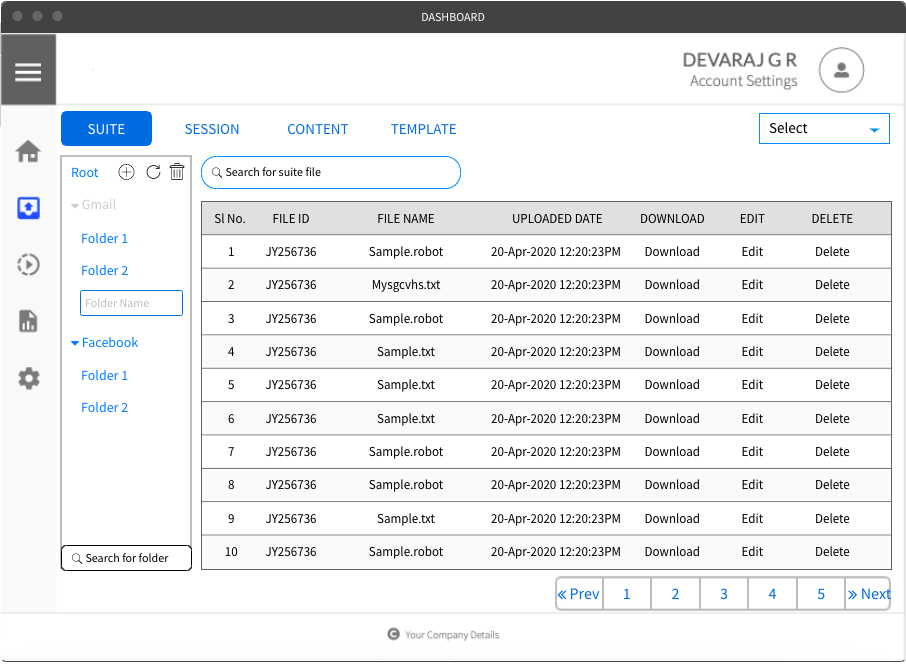


Figure 6: Create a folder in folder structure

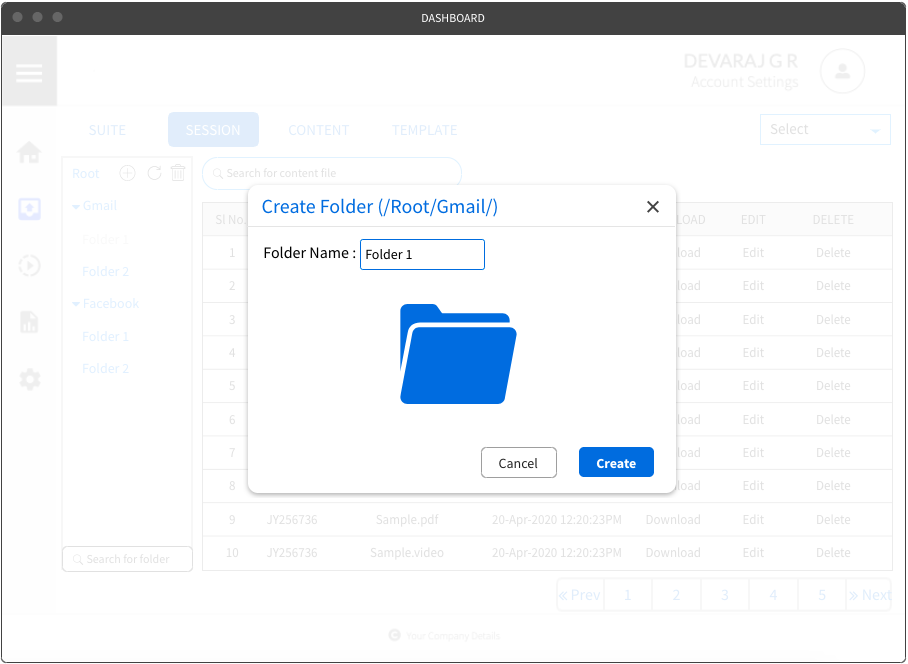


Figure 7: Create a folder in popup view

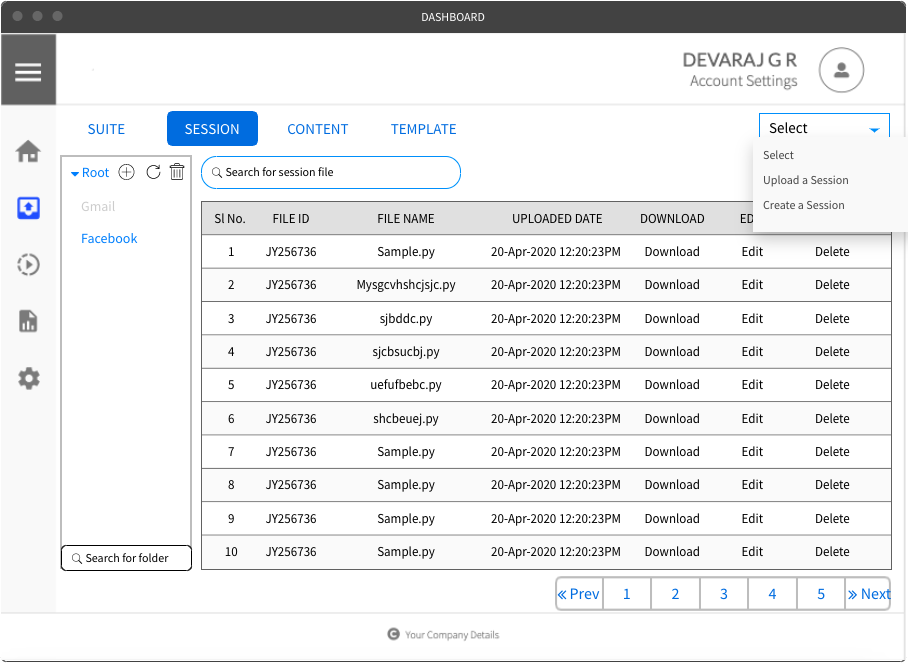


Figure 8: Session file management

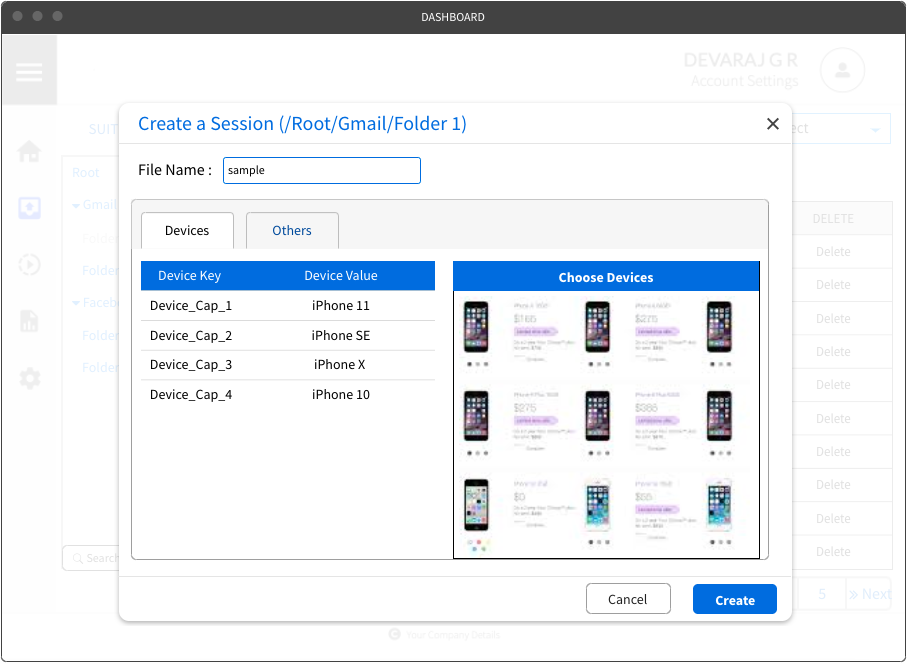


Figure 9: Create a session by selecting devices

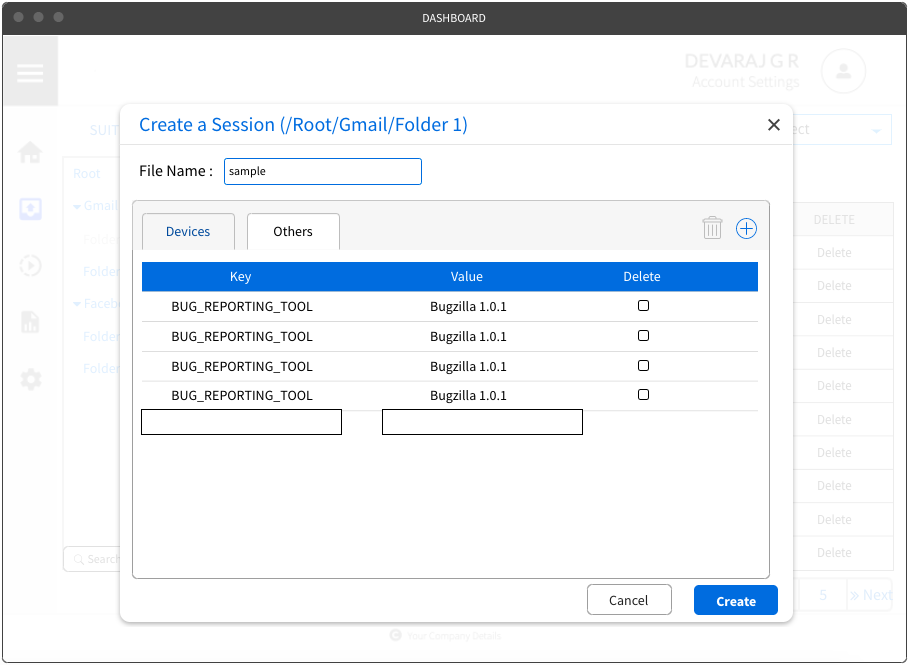


Figure 10: Create a session by adding other attributes

* 1. Test Execution

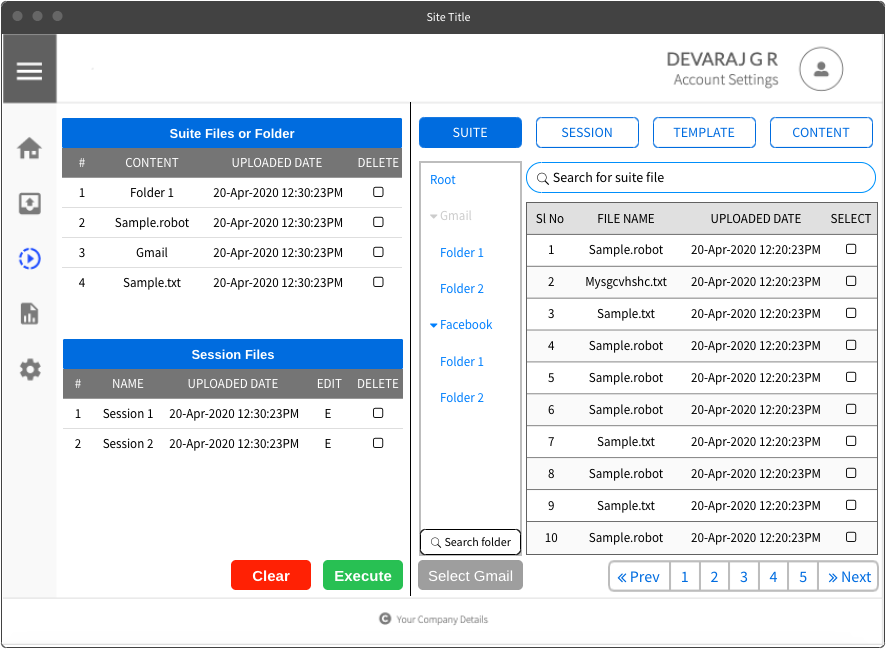


Figure 11:Test execution files selection and execute

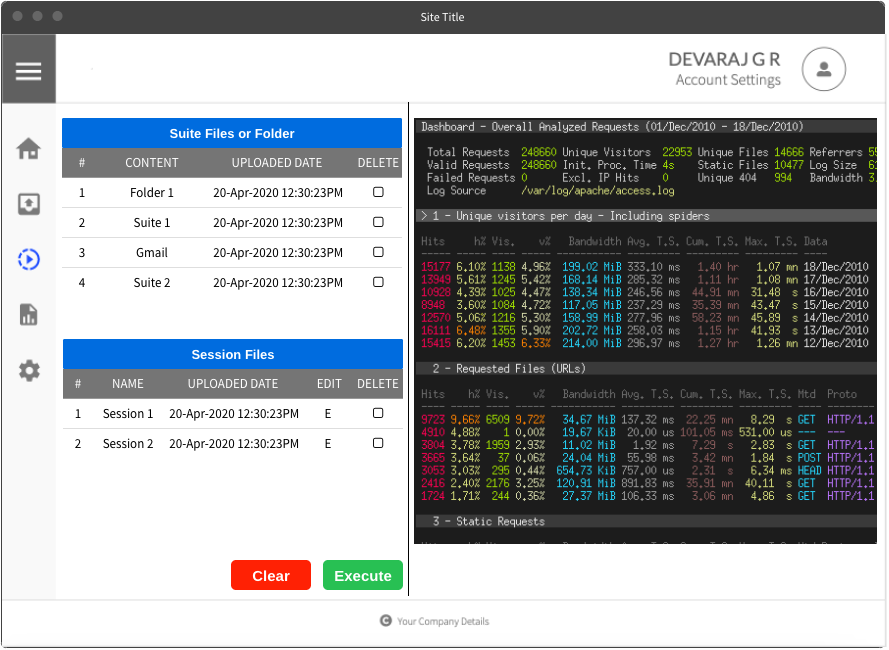


Figure 12: Test execution and viewing logs

1. **View Logs & Reports**
   1. View the logs and reports list (Table view providing searching, sorting and filtering options).
   2. Delete the logs and reports.
   3. Email the logs and reports.
   4. Export logs as PDF or text files.
   5. Download the logs and reports.
   6. Segregate the logs into success and failure logs.
2. **Settings**
   1. Settings are extra feature to enable the user profile related activities.
   2. Change password or Forgot password.
   3. Profile updates activities.

1. Others
   1. Limitations

<Describe the constraints, if any. Describe what your design can and cannot do>

* 1. Deviations

<Indicate the deviations from standards, or other requirements here.>

1. Open Issues <Optional>

<The open issues faced during the design of the system may be recorded here but **tracking and closure of the open issues should be taken up in the Action Item tracking or Issue tracking tool used by the project,** as mentioned in the Project Plan. This section should be updated once the issues are resolved. Use a table format for ease of reference.>