**Analytic Dashboard Server Documentation**

* **Project Structure:**
  + Common
    - Constant.js
    - Logger.js
  + Database
    - Database.js
    - Default.js
  + Helpers
    - Middleware.js
  + Modules
    - Announce.js
    - File-reader.js
    - Dashboard.js
    - Login-register.js
    - Report-reader.js
    - Report-writer.js
    - Server.js
    - User-management.js
  + Uploads
  + Main.js
  + Package.json

1. **Common Directory**:
   1. Constant.js: The file contains all constant variables with constant data that will not change during application runtime such as server port, database name, folder paths, folder names, script commands etc.
   2. Logger.js: Instantiate logger to write server logs during error situations to server.log file in root directory of server.
2. **Database Directory**:
   1. Database.js: The file creates a connection between database server and application server. It contains all the functions (Read, Write, Edit, Delete) of database to read and manipulate data in database.
   2. Contains default database structure file to create required collections on new server start or fresh install.
3. **Helpers Directory**:
   1. Contains Middleware.js that handles user session, checks of validity on user login. The middleware is used in passport authentication in server.js.
4. **Modules Directory**:
   1. File-reader contains functionality to read uploaded reports such as csv, xls, sav files.
   2. Dashboard contains API to return dashboard statistics such as total reports and reports generated.
   3. Report-reader and report-writer to read reports and write to database.
   4. Server.js file contains express server with session management that opens all APIs to client and session management
   5. User-management contains APIs to edit user data for super admin and fetch user profile data
5. **Uploads Directory**: The directory is used to dump all the files uploaded by the client to server in their respective sub directories as per extension or file types.
6. **Main.js**: The process file that initiates all activities according to flow.
7. **Package.json**: Contains server configurations and package configurations.

**API Documentation:**

1.  **/ (GET Request):** Return server status. i.e. (Online if running)

2. **/o/login (POST Request):** Login API when user hits login button with username and password.

3. **/o/logout** **(POST Request)**: Logout API called when user clicks logout in dashboard. The same is called when session is expired.

4. **/o/register (POST Request)**: Registration API to register the user. Takes 4 inputs, username, password, repeat password and email Id.

5. **/o/user (POST Request)**: The API is used to check the active session of that user. Hits logout if session is expired.

6. **/o/getdashboardstats (POST Request)**: The API returns dashboards statistic like Total Files Uploaded, Generated Reports, Last Report, User Data. Called on login successful.

7. **/o/uploadfiles (POST Request)**: The API is used to upload the Raw data file to server with report name. Returns the data from uploaded file.

8**. /o/profile (POST Request)**: The API return user profile data. It allows to edit profile and deactivate the account of user.

9. **/o/uploadreport (POST Request)**: The API uploads final generated report to database after selecting the data graphs on client end.

10. **/o/reporthistory (POST Request)**: Fetches all the reports generated by the user from database.

11. /**o/getusers (POST Request)**: Get all users profile data from database. (Super Admin API)

12. **/o/user-mgmt (POST Request)**: API for user management to super admin. Edit delete and perform actions to user profile using the API. (Super Admin API)

13. **/o/announce (POST Request)**: Fetch all announcements from database. API to insert new announcement. (Super Admin API)