DWE - Authoring Application

1. **Project Details:**
   1. This project is an authoring application (just like any other CMS), thus its admin part contains two sections:
      a. Server
      b. Client (in this we refer the admin as client section).
      The folder structure of the project also follows the same naming conventions.

   2. The server is setup using **Node.JS framework**. It is a server side scripting framework based on Javascript. Since, we are using NodeJs for the server setup, we have used **Node-Package-Manager (NPM)** for most handling most of the modules/packages that we use in the project.

   3. The front-end of the website is developed using **AngularJS**. Since we have used many packages of angular-stack in the project, we have also used **Bower Package Manager (bower)** for installing front-end dependencies.

   Note: Any package can be installed using npm/bower in the following manner:

   ```
   npm install -g package-Name
   bower install -g package-Name
   ```

   The -g *flag installs the package globally. If certain packages are only required globally, execute the above command without the* -g *flag.*

   While installing globally, the node packages get installed at the following location:
   User/AppData/Roaming/npm

   When we install a package locally, npm creates a node_modules folder in the current directory and installs the package inside it.

   (Please read about package managers carefully to develop a full understanding of npm and bower).

   4. The database for this project is setup using **MongoDB Database**. It is a purely relational database which uses JSON like structure to store the data. Read in detail about mongodb at (https://docs.mongodb.com/v3.2/tutorial/). This link provides all the information from installation to working with mongodb databases.

   Note: MongoDB requires a specific folder structure in the system. You need to manually create the following directory in the C-Drive of your system:

   C:/data/db

This is because once you run the mongo-instance, it searches for this folder and stores all the database files in this directory only.

Note: To be able to read the databases created in this directory, you require Robomongo a utility that enables to access and read databases present in the mongo-server.

5. To automate certain tasks in the development phase, we have used **Grunt Task automation tool** based on JavaScript. (*This module can be installed using npm.*)

6. Pre - requisites on the system:
    1. Node.JS
       This can be installed from its official site. You can download the zip folder and provide its path in the PATH environment variable.
    2. NPM
       Node package manager comes along in the node.js package.
    3. Bower
       It can be installed using the node-package-manager.
    4. MongoDB and Robomongo
    5. Grunt

2. Steps to run the application:

1. Download the repository from https://github.com/DhruvS1213/dwe_admin_multiple
2. Unzip and open terminal in the project's directory.
3. Perform following operations:

    1. `npm install`

    This will install all the packages mentioned in the package.json file and store it in a directory named node_modules.
    (You will require to set npm-proxy before executing the above command while working from a corporate (TCS) network.)

    2. `bower install`

    This will install all the packages mentioned in the bower.json file and store it in a directory named client/bower_components
    (You will require to set http-proxy and https-proxy before executing the above command while working from a corporate (TCS) network.)

    3. `grunt serve`

    This will run the grunt-task mentioned in GruntFile.js and run the application on the server `(http://localhost:9000)`.