Software Engineering Project

Group 3

**Technical Documentation**



Team Members:

**120101002** ABHINAV MITTAL

**120101003**  ABHINAV RAVI

**120101004**  ABHISHEK GOYAL

**120101032**  KAZIPETA SHARATH

**120101036**  KONAKANCHI REVANTH GOPI

**120101041**  MOHIT

**120101045**  NISHANT GUPTA

**120101057**  RANU VIKRAM

**120101058**  RAVINDRA SINGH GURJAR

**120101061**  ROHITASHVA KUMAR MEENA

**120101062**  ROSHAN NATHANI

**120101066**  SANDEEP SAMATAM

**120101083** ABHIJEET SINGH

­­**120101087** ABHISHEK SEN

**Preface**

The document herein was produced by the **Saanji Drive Team**, a group of highly enthusiast computer science undergraduates of the Indian Institute of Technology Guwahati. This document is intended to provide a technical over-view of this project, Saanji Drive, a web-based file sharing portal with the provision of auto-synchronization of files as the user is on the move.

This project was developed mainly in the timespan of one month, during March 20, 2014-April 18, 2014. Developed under the guidance of **Prof. (Dr.) Pradeep Kumar Das**, this group could visualize the importance of working as synchronized chunks of software development units and could complete this project, while enjoying the challenges that crept up. There are no restrictions on the reproduction, distribution, translation or use of this document. However, incorporation of this document, in part or in whole, into any other document does not convey or represent an endorsement of any kind by the Saanji Drive Team.

The Saanji Drive Team

2nd Year,

Department of Computer Science and Technology,

Indian Institute of Technology Guwahati

Guwahati – 781039,

India

Date – 18th April, 2014

**Contents:**

1. **Introduction**
2. **Website**
   1. **Frameworks**
   2. **Purpose**
   3. **Scope**
   4. **Functions**
3. **Desktop Application**
   1. **Frameworks**
   2. **Purpose**
   3. **Scope**
   4. **Components**
   5. **Functions**
4. **Reference**

1. **Introduction**

This is an introduction to the technical descriptions of the Project on Web-Based-File-Hosting-Server. This Project has been developed to construct an online drive storage developed with these functionalities:

* Login

-- Sign in

-- Sign out

-- Sign up

-- Forgot password

* Manage Folders

-- Add folder

* Upload file
* Download file
* Search
* Share file
* Preview file
* Edit file

1. Website
   1. **Framework:**

We have used the Bootstrap Framework to build up the website pertaining to this project. Bootstrap is a free and open source web application framework, which contains HTML and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions.

For the basic aesthetic look of the website, we have used the Bootstrap CSS package to avoid re-defining the classes of the CSS classes for the different subsections of the web-site.

We have used the Bootbox Framework to build all alert boxes pertaining to this project. Bootbox is much like bootstrap, but for alert, prompt and other javascript pop-up boxes.

We also used PYOD converter for providing preview of document files. PyODConverter (for Python OpenDocument Converter) is a Python script that automates office document conversions from the command line using LibreOffice or OpenOffice.org.

jQuery(javascript library) is used for providing animations and other aesthetic look and UI aspects of the project.

Dropzone javascript library is used for providing drag and drop feature.

* 1. **Purpose**

SaanjhiDrive is created with the purpose to provide efficient sharing of files and backing up data on the cloud. It has an added feature of signing up, this is not present in the executable version. SaanjhiDrive allows users to create a special folder on each of their computers, which SaanjhiDrive then synchronizes so that it appears to be the same folder (with the same contents) regardless of which computer is used to view it. Files placed in this folder also are accessible through the website.

* 1. **Scope**

We can allow for the upload of files directly to the user’s account, sharing of files, and the deletion of the file(s). When this particular user connects to the server from the personal computer, these files would be synchronized to that particular computer and thus the user who might or might not have the PC available to himself/herself, would still be able to properly store the files as per the needs of the hour.

* 1. **Functions**

The following are the functions used in the program along with their purpose of each function used:

* ***Source::* index.php**

Functions:

* **function signin() { }**

*Functionality:* checks validation of username and user password of form “ signinform” in index.php and redirects to ‘mainpage.php’ on proper values

* **function check\_username() { }**

*Functionality:* checks validation of username in ‘signup\_form ‘

* **function check\_userpwd() {}**

*Functionality:* checks validation of user password in ‘signup\_form ‘

* **function check\_userrepwd() {}**

*Functionality:* rechecks the password entered previously is matching with the second one

Variables:

* var uname : This variable stores the value of the username entered in the form.
* var upwd : This variable stores the value of the password entered in the form.
* var reguname : This variable stores the value of the username entered in register form.
* var regupwd : This variable stores the value of the password entered in register form.
* var regreupwd : This variable stores the value of the reenterd password in register form.
* **Source :: sign\_in.php**

*Variables ::*

* $name : This variable stores the username which is received from the index.php.
* $pass: This variable stores password of the user received from index.php.
* $con : This variable stores the connection of the mysql database.
* $query : This variable stores the query of selecting username and password from users where username is from $name.
* $result : This variable stores the result after execution of the $query.
* $row : This variable fetches the array of the variable $result

*Functionality ::* this sign\_in.php file receives the username and password from index.php file by POST method . It access the database and verifies the username with password, if valid then a new session starts using the username otherwise it shows an error message and redirects the user to home page using GET method. If remember me is checked, then it defines a cookie to be sent along with the rest of the HTTP headers, thus providing the users to not provide his/her identity every he/she visits the website.

* **Source :: register.php**

*Variables ::*

* $name : This variable stores the username which is received from the index.php.
* $pass : This variable stores the password of the user which is entered in index.php
* $email : This variable stores the email of the user which is entered in index.php
* $con : This variable stores the connection of the mysql database
* $query : This variable stores the query of selecting username from user where username is from $ name
* $data : This variable stores the data after coneection to database and executing the query
* $user : This variable stores the value of $name
* $response : This variable is used to store the response
* $con1 : This variable stores the connection to backup server .

*Functionality****:*:** This register.php file receives the username ,user password , user email from index.php file by post method . It access the database and checks whether the entries are already present in the database, if not then it adds to the database. It also maintains the status of the query in log.txt in case a server crash occurs, while processing the request. Also, connection to a backup server is established to make backup of the query performed.

* ***Source ::* mainpage.php**

Functions:

* **function callUpload() { }**

*Functionality::*this function is called when upload is clicked. It performs an AJAX query with the server. It contains an XMLHttpRequest object which is used to exchange data with a server so that it is possible to perform the uploading by performing a SQL query on the server, without reloading the whole page. It sends the execute ‘upload\_file.php’.

* **function getFiles () { }**

*Functionality::*it contains XMLHttpRequest object which is used to exchange data with a server so that it is possible to update parts of a web page, without reloading the whole page. It sends a request to a server, to open ‘populate.php’ and receive file list as link data using XMLHttpRequest object.

* **Source :: Upload\_file.php**

*Variables:*

* $shared\_with : This variable stores the users to whom the file is shared .
* $file\_id: This variable stores file id of the requested file

$owner: This variable is used to stores the owner of the file

* $shared\_with: This variable stores the username of the user whom the file is shared by
* $file\_hash: This variable stores hash of the requested file.
* $already\_present: This variable stores a numerical value indicating whether the file is present or not if so file belongs to the same user or other different user. i.e if file\_hash and owner matches then variable stores ‘2’ or if the file\_hash matches and owner doesn’t match then variable stores ‘1’.
* $file\_name: This variable stores file name of the file.
* $file\_id\_ext: This variable used to store extension of the file.
* $temp: This variable stores tokens separated by ‘.’ from $\_FILES["file"]["name"]
* $ext: This variable stores the extension of the file from $temp.
* $result\_new: This variable stores after connection and execution of the query, which is selecting everything from filesystem where owner is $owner and directory\_path is directory\_to\_upload and order by file\_name.
* $row1: This variable stores in array form after execution of $result\_new
* $query: This variable stores the query of selecting the file\_hash from filesytem .
* $result: This variable stores after connection to database and selects file\_id, file\_hash, owner from filesystem and order by file\_id
* $con: This variable stores the connection to the database.
* $retval: This variable stores the query after execution of the query $query andconneting to database.
* $directory\_to\_upload: This variable stores

*Functionality ::* this php file checks database connections and on proper connection fetches ‘file\_id’ and ‘file\_hash’ from file system . this retrived information is rechecked with the upload file information .If both the values match then it implies that file is already present otherwise the file will be uploaded .

* **Source :: Logout.php**

*Functionality::* This php file destroys the session and , enables the cookies and redirects to the index.php

* **Source :: Remove.php**

*Variables:*

* $name: This variable stores the name of filename which is received from previous php file.
* $currDir: This variable indicates the current directory
* $user: This variable stores the username of the user which is received from the precious php file.
* $con: this variable is used to store the connection to the database.
* $id: This variable stores the $row['file\_id'].
* $query: this variable stores the query of selecting everything from filesystem wher file\_name is $name and directory\_path is $currDir.
* $temp: This variable stores the tokens of $name by exploding using ‘.’.
* $ext: This variable stores extension of the file concerned.
* $count: This variable is used to keep track of the row no from the result obtained by performing SQL query.

*Functionality ::* This php file receives the name of the file and current directory details from the previous form by post method and connects to the database . It runs a query in the database searching for the file, if the count is equal to 1 then it deletes the file, for all other cases it displays a fail.

* **Source :: Rename.php**

*Variables:*

* $newName: This variable stores the new name of the file to be renamed which is received from the previous file.
* $oldname: This variable stores the name of the original file, which is received from the previous file.
* $currDir: This variable stores the current directory of the file name to be changed, which is received from the previous file.
* $user: This variable stores the name of the user.
* $query: This variable stores a query of updating filesytem by setting file\_name as $newName where owner as $user and directory\_path as $currDir and file\_name as $oldName.
* $result: This variable executes the query stored in $query by connecting to database

*Functionality ::* This php file receives the current name , old name and current directory details from the previous form by post method. Then it connects to the database and updates the old name of file with the new name of the file.

* **Source ::**  **search.php**

*Variables:*

* $user: This variable stores the name of the user.
* $key : this variable stores the value of the key to be searched , which is received from the previous file.
* **$**query: This variable stores a query of selecting everything from file system where owner as $user and order by descending order if it is Folder.
* $con: This variable stores the connection of the database.
* $result: This variable executes the query stored in $query by connecting to database
* $cond1: This variable is used as a flag.
* $cond2: This variable is used as a flag.
* $temp: This variable stores the tokens of $name by ‘.’.
* $ext: This variable stores extension of the variable $temp.
* $pos: This variable is used as a flag.
* $x: This variable is used as a flag.

*Functionality ::* This php file gets the keyword to search in the text file . It connects to the database, and sorts the files and folders. It firstly searches each folder name whether it contains the keyword, if it contains then it displays the folder list by including the ‘FileList.php’.

It also searches insides the contents of text file whether the key is present, if present it returns the file list by including the ‘FileList.php’ .

*Functions:*

***Function search ($Filename, $keyword) { }***

*Functionality:: this function gets the contents of the ‘.txt’ file and searches for the keyword , if found then it returns TRUE or returns FALSE.*

* **Source ::FileList.php**

Variables:

* $name: This variable stores the file\_name.
* $temp: This variable stores the tokens of $name by ‘.’.
* $ext: This variable stores extension of the variable $temp.
* $count: This variable is used to count.
* $folder\_name: This variable stores the foldername.

*Functionality ::* This Php file receives the file name from the previous php document ,firstly this trims the file name . Finds the extension of the file and depending on the extension of file it correspondingly displays the class of the file and also displays the by whom file is shared .

* **Source ::convert.php**

VARIABLES:

* $source: This variable stores the source of the file.
* $name: this variable stores the tokens of the $source
* $dest : This variable stores the file name.
* $ext: This variable stores the extension of the file.
* $query: This variable stores a query of '"C:\Program Files (x86)\LibreOffice 4\program\\python" cgi-bin\DocumentConverter.py '.$source.' '.$dest.'.pdf';
* $result: This variable stores after execution of the query $query.

*Functionality ::*This php file receives the source of the file and finds the extension of the file . If the extension is not pdf or pptx then the file uses python on the server to convert the file to pdf.

* **Source ::updatePath.php**

**Variables:**

* $path: This variable stores the path of the file, which is received from the previous file.
* $user: This variable stores the username of the user.
* $filename: This variable stores the filename, which is received from the previous file.
* $initPath: This variable stores the initial path, which is received from the previous file.
* **$**query: This variable stores a query of updating filesystem by setting directory\_path = $path where owner as $user and file\_name as $filename and directory\_path as $initPath.
* $con: This variable stores the connection of the database.
* $result: This variable executes the query stored in $query by connecting to database

*Functionality ::* This Php file receives the final path and initial path from previous file. It connects to the database and updates the file directory path.

* **Source ::shared\_with\_me.php**

**Variables:**

* $user: This variable stores the username of the user.
* **$**query: This variable stores a query of selecting everything from filesystem where owner as '$user' and shared by! = ''” and isFolder = '0'.
* $con: This variable stores the connection of the database.
* $result: This variable executes the query stored in $query by connecting to database.
* $name: This variable stores the filename.
* $temp: This variable stores the tokens of $name by ‘.’ .
* $ext: This variable stores extension of the variable $temp.

*Functionality ::* This Php file runs a query which finds all the files which are already shared by the user .

Depending on the extension of the file it displays an image. It provides a preview if the file is an image. In each row of list it displays the user who shared it.

* **Source ::populate.php**

Variables::

* $user: This variable stores the username which is received from its previous php file.
* $folder: This variable stores the name of the folder which is received from its previous php file.
* $con: This variable stores the connection with the MySQL database.
* $query: This variable stores the query updates the filesystem with entries owner as $user and directory\_path as $folder ordered by isFolder in descending fashion.
* $result : This variable stores the result after execution of $query.
* $row: This variable stores the fetched array from the variable $result.

*Functionality ::* This Php file receives the folder name from previous file. It connects to the database, if connected it includes the ‘FileList.php’.

* **Source ::Create\_folder.php**

*Variables::*

* $owner : This variable stores the username which is received from its previous php file.
* $file\_name : This variables stores the updated version of the entry folder\_name after replacing “ “ with “\_”.
* $directory\_path: This variable stores the directory path of the file received from its previous php file.
* $con : This variable stores the connection with the MySQL database.
* $result : This variable stores the query which after setting up the connection with the database , updates the entries file\_id and file\_name in the filesystem ordered by file\_id.
* $row: This variable stores the fetched array from the variable $result.
* $file\_id : This variable stores the file id of the file after the filesystem has been updated.
* $insert\_query: This variable stores the query to insert the entries file\_id, file\_name, owner, directory\_path, isFolder into the filesystem as '$n\_file\_id', '$file\_name', '$owner', '$directory\_path', '1' respectively.
* $insert\_result : This variable stores the result after execution of the query $insert\_query.

*Functionality:*: This php file creates a new folder in the directory specified by the user. It first establishes connection with database, checks if the user has same folder in that directory if not then creates the folder as defined by the user

* **Source::download\_file.php**

Variables ::

* $owner : This variable stores the username which is received from its previous php file.
* $directory\_path\_initial : This variable stores the directory path of the file received from its previous php file.
* $file\_name : This variable stores the name of the file which is received from its previous php file.
* $con : This variable stores the connection with the MySQL database.
* $get\_file\_query : This variable stores the query which after setting up the connection with the database , updates the entries file\_id and file\_name in the filesystem by setting file\_name as $file\_name and owner as #owner nad directory path as $directory\_path\_initial and isFolder=’0’.
* $row : This variable stores the fetched array which is received from the variable $ get\_file\_query.
* $n\_file\_id : This variable stores the file id of the file after the filesystem has been updated.
* $temp : This variable stores the tokens of ‘file\_name by ‘.’ .
* $ext : This variable stores the extension of $temp.

Functionality:: This php file downloads the file specified by the user from a given folder/directory. It first establishes connection with the database, checks if the file to be downloaded exists in the folder/directory or not, if yes then it downloads the file.

* **Source::delete.php**

*Variables ::*

* $file : This variable stores the amended file after replacing the character ‘/’ with ‘\\’ from the same file.
* $query : This variable stores the query of deleting the file.
* $result : This variable stores the result after execution of the query.
* $source : This variable stores the source which is received from its previous php file.
* $ext : This variable stores the extension which is received from its previous php file.

*Functionality::* This php file deletes the file specified by the user from the specified folder/directory.

* **Source::share.php**

Variables::

* $owner : This variable stores the username which is received from its previous php file.
* $user\_to\_share\_with : This variable stores the user information that has to be shared.
* $directory\_path\_initial : This variable stores the directory path of the file received from its previous php file.
* $file\_name : This variable stores the name of the file which is received from its previous php file.
* $con : This variable stores the connection with the MySQL database.
* $query : This variable stores the query of selecting the username from users.
* $result : This variable stores the result of the query after establishing connection with the database.
* $row : This variable stores the fetched array which is received from the variable $result.
* $get\_file\_query : This variable stores the query which after setting up the connection with the database , updates the filesystem by setting file\_name as $file\_name and owner as #owner nad directory path as $directory\_path\_initial.
* $n\_file\_id : This variable stores the file id of the file after the filesystem has been updated.
* $n\_file\_hash : This variable stores the file hash of the file after the filesystem has been updated.
* $share\_query : This variable stores the query to insert the entries file\_id, file\_name, owner, file\_hash, directory\_path, isFolder, shared\_by into the filesystem as '$n\_file\_id', '$file\_name', '$user\_to\_share\_with', '$n\_file\_hash', '!', '0', '$owner' respectively.
* $perform\_share : This variable stores the result after execution of the query $share\_query.

*Functionality:* This php shares the file specified by the user from the database. It first checks if the user is valid i.e. if his/her username is valid to access the user tables, then shared the file as requested by the user.

1. Desktop Application
   1. **Framework:**

Visual C# has been used to prepare the GUI for the application. ASP.NET is used to communicate with the server.

* 1. **Purpose**

This application serves the following essential functions:

* Login
* Upload
* Download
* Synchronizing
  1. **Scope**

User can login via this application to the username/password given at the website. He can then add the paths of the file/folders that are to be added to his/her account on the file sharing server. Common errors by the users that have been accounted for include:

* + - If wrong username and/or password are used then upon pressing the login button this message is displayed "Incorrect query".
  1. **Components Used:**

**Form1:**

1. **Buttons**:
2. Select folder: browses to select the folder for storing the files and folders of the drive.
3. Save : to save the username, password and path for uploading the data in drive.
4. Sync: to sync the files and folders with the online storage
5. **Labels:**
6. Username, password,website,saanjhi drive to display the label headers
7. Settings : To change the visibility of panels, and itself changes to Back.

**(c)** **Textboxes:**

1. Txtusername to write the username.

2. Txtpassword to write the password.

3. Textbox1 to write the path of storage site.

**(d) Panels** to group the different buttons and labels.

* 1. **Functions**

Following are the functions used in the page **Form1.cs**

Of this application:

1. private void Form1\_Load(object sender, EventArgs e)

This function is called when Form1 is loaded. This function mainly sets the visibility of panels and buttons when the form is loaded first. It also checks whether the file **userCredentials.txt** exists. If it exists it will delete the data in it otherwise it will create this file.

1. private void button2\_Click(object sender,EventArgs e)

This function is called when button (select folder) is clicked. It opens the browsing dialogue for selecting the folder location for the directory our drive. It also writes the username, password and site path to file **userCredentials.txt.**

1. private void lblSettings\_Click(object sender, EventArgs e)

This is function is called when the label settings is clicked. This is mainly used to switch between panels . If settings is clicked the panel for entering the username , password opens and it’s label changes to Back . if we click Back the panel containing the sync button will be opened.

1. private void button1\_Click(object sender,EventArgs e)

This function is called when button (save) is clicked The main function is to save the username , password and directory path to the file **userCredentials.txt.** It writes the username in first line , password in second and directory path in the last line.

1. private void DownloadStatusFile\_Click(object sender, EventArgs e)

This function is called when the button (sync) is clicked. First of all It will call the function Sync\_Click. Then it will check the directory by calling the function TraverseDirs. Traversing the directory generates the present structure of the files and the folders and saves them to the file “presentStructure.txt”. Then it reads the status.txt and if the status is ready to download it will call the function DownloadFromServer to download the files and folders from the server. After that, the files that are not present on the server but are present in the user’s desktop (found by comparing the files – “presentStructure.txt” and “currentStatus.txt”) are uploaded to the server.

1. private void Sync\_Click(object sender, EventArgs e)

This function is called in the above function. This function takes the username, password and site address and write them to the file status.txt. It also sends the data of status.txt to server for verification by using upload.aspx.

1. private void TraverseDirs(DirectoryInfo dir)

This function is called in DownloadStatusFile\_Click.The main work of this function is to traverse each and every folder present in the home directory and server. It also checks which of the files and folders are not synced with the online storage.

1. private int DownloadFromServer(string whereToDownloadFrom,string filePathOnClient,string actualFileName)

This function is called in DownloadStatusFile\_Click.The work of this function is to download the files and folders from the server to the home directory which are not present in home folder.

1. private void generateFilesToUploadToServerandUploadThem()

This function is called in DownloadStatusFile\_Click.It reads the files currentStatus.txt and if the files are not present on the server , then it writes the path of these files and folders to the file xtrafiles2beadded2server.txt ,then these files are uploaded to the server by calling the function uploadFileToServer().

1. Private int uploadFileToServer(string fullUploadFilePath)

This function is called in the above function. The main work of this function is to upload the files to server which are present in home folder but not on server. It does this task by using the page upload.aspx.

ASP.NET Pages:

These include three files:

1. uploadStatus.aspx - It is used to upload the “status.txt” to the server. It uses it to check the password and username of the user. Also, it generates the file “currentStatus.txt” (which is downloaded later by the user) containing the present files of the user saved on server.
2. Upload.aspx – All the upload queries from the server are served by this page. It uploads the file to the server.
3. XtraFilesFromClient2Server.aspx – This script updates the database with the new files that have been added to the server from the client’s desktop application. It checks if the file is already present on the server (by MD5 file hash). If yes, it deletes the newly uploaded file and retains the already uploaded file.
4. Reference
5. <http://getbootstrap.com/css/>
6. <http://jquery.com/>
7. <http://www.artofsolving.com/opensource/pyodconverter>
8. <http://searchwindowsserver.techtarget.com/definition/IIS>
9. <http://www.dropzonejs.com/>