

# Cloud-Based Backup Service

---

**Cloud Computing Course Project Report – Fall 2013**

**Team:**

**Adithya Nayabu  
Bhagyasri Pavuluri  
Manoj Kidambi  
Saitej Erupaka**

**University of Houston**

## Table of Contents

<b>Abstract.....</b>	<b>2</b>
<b>1. Introduction.....</b>	<b>2</b>
<b>2. Design and Implementation.....</b>	<b>3</b>
<b>2.1 Technology and Cloud Service.....</b>	<b>3</b>
<b>2.2 Requirements.....</b>	<b>4</b>
<b>2.3 Architecture.....</b>	<b>5</b>
<b>2.4 Implementation.....</b>	<b>6</b>
<b>2.5 Contribution Report.....</b>	<b>9</b>
<b>3. Results.....</b>	<b>10</b>
<b>3.1 Requirements.....</b>	<b>10</b>
<b>3.2 Data.....</b>	<b>10</b>
<b>4. Results.....</b>	<b>12</b>
<b>Conclusions.....</b>	<b>13</b>

## **Abstract:**

In the current scenario, people carry or back up files in flash drives and external drives. Popular applications like Dropbox, Google Drive are also being used for these purposes. Limitations of using these methods are they have memory constraints and the user needs to manually need to upload the files/folders to the flash drives/ Dropbox folders. Cloud-based backup service automatically backs up the folders to the cloud on a timely basis, releasing the burden on the user to manually drag/ upload them to the flash drivers/ cloud. Users can add the list of folders to be backed up in the client application, and these folders can be located in different drives in the user machine. This application backs up specified files and folders to the cloud automatically easing the effort of the user to manually upload them. It provides the flexibility to the user, to specify the time at which all the pending files need to be synced. The user may also specify a different path for an already uploaded file. The application has two main components: Client application and the web application. Client application is the desktop application using which the users select the files to upload and also specify the time at which the files should be uploaded. Web application displays all the uploaded files and provides download option. The users may also share their files by sharing the URL.

## **1. Introduction:**

Our application consists of two parts: client application and web application. Client application is a desktop application that allows the user to register with the system and allows the user to configure the paths to the folders that to be backed up to the cloud. The application continuously monitors the specified paths and automatically backs up the modified files/folders to the respective buckets in the cloud at a certain time. The client application needs to be run continuously in the background.

We maintain the metadata of the files and folders which are backed up in the server. We use this information to determine whether the file is modified and needs to be uploaded. We also store the information of the user in separate database. The cloud we are planning to use is Amazon S3 and Amazon EC2. Each user's data is collected in a bucket assigned to him during the registration. Web application helps the user to access the files that are uploaded to the cloud. The user can access the files anywhere he wants.

Amazon S3 provides flexible interface to upload the files to bucket via our client application. It provides storage through web service interfaces. It aims to provide a scalability, high availability and low latency. Amazon EC2 is a part of the Amazon's cloud computing platform. It allows the users to rent virtual computers on which they run their own computer applications. EC2 allows scalable deployment of applications by providing a Web service through which a user can boot an Amazon Machine Image to create a virtual machine, which Amazon calls an "instance", containing any software desired. The services used and further information regarding the implementation in the project, project architecture and

requirements are discussed in Section 2.0. The client application is compatible with Windows OS and comes with an easy installable file. The web URL can be accessed from anywhere and provides a user friendly interface to view and download the uploaded files. The results, screenshots and the input formats are discussed in Section 3.0.

## 2. Design and Implementation:

### 2.1 Technology and Cloud Service:

The tools and technologies used in the project are listed below:

- Amazon S3 / Amazon S3's clone
  - Amazon EC2
  - C#, .Net
  - Application Server
  - SQL Database server
- **Amazon S3:** Amazon S3 is storage for the Internet. It is designed to make web-scale computing easier for developers. Amazon S3 provides a simple web services interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the web. It gives any developer access to the same highly scalable, reliable, secure, fast, inexpensive infrastructure that Amazon uses to run its own global network of web sites. The service aims to maximize benefits of scale and to pass those benefits on to developers. It enables the users to write, read, and delete objects containing from 1 byte to 5 terabytes of data each. The number of objects you can store is unlimited. Each object is stored in a bucket and retrieved via a unique, developer-assigned key.

In our project we use Amazon S3 to store the files uploaded by the user. We create individual buckets for each user and store the files uploaded into the bucket.

- **Amazon EC2:** Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud. It is designed to make web-scale computing easier for developers. Amazon EC2's simple web service interface allows you to obtain and configure capacity with minimal friction. It provides you with complete control of your computing resources and lets you run on Amazon's proven computing environment. Amazon EC2 reduces the time required to obtain and boot new server instances to minutes, allowing you to quickly scale capacity, both up and down, as your computing requirements change. Amazon EC2 changes the economics of cloud computing servers by allowing you to pay only for capacity that you actually use. Amazon EC2 provides developers the tools to build failure resilient applications and isolate themselves from common failure scenarios. To use the service, Users need to select a pre-configured, template Amazon Machine Image (AMI) to get up and running immediately. Or create an AMI containing your applications, libraries, data, and associated configuration settings. Configure security and network access on your Amazon EC2 instance. Choose which instance type(s) you want, then start, terminate, and monitor as

many instances of your AMI as needed, using the web service APIs or the variety of management tools provided.

In our project we use Amazon EC2 instance to host our web application in the cloud.

- **C#, .Net:** .NET Framework is a software framework developed by Microsoft that runs primarily on Microsoft Windows. It includes a large library and provides language interoperability (each language can use code written in other languages) across several programming languages. Programs written for .NET Framework execute in a software environment (as contrasted to hardware environment), known as the Common Language Runtime (CLR), an application virtual machine that provides services such as security, memory management, and exception handling. The class library and the CLR together constitute .NET Framework. .NET Framework's Base Class Library provides user interface, data access, database connectivity, cryptography, web application development, numeric algorithms, and network communications. Programmers produce software by combining their own source code with .NET Framework and other libraries. .NET Framework is intended to be used by most new applications created for the Windows platform. Microsoft also produces an integrated development environment largely for .NET software called Visual Studio.

We have used .Net to develop the client application as well as web application.

- **Application Server:** IIS (Internet Information Services) is an extensible web server created by Microsoft for use with Windows family. IIS supports HTTP, HTTPS, FTP, FTPS, SMTP and NNTP. With IIS, Microsoft includes a set of programs for building and administering Web sites, a search engine, and support for writing Web-based applications that access databases. Microsoft includes special capabilities for server administrators designed to appeal to Internet service providers (ISPs). It includes a single window (or "console") from which all services and users can be administered. It's designed to be easy to add components as snap-ins that you didn't initially install. The administrative windows can be customized for access by individual customers.

In our project, we use IIS to host our published application in the remote AWS instance.

- **SQL Server:** Microsoft SQL Server is a relational database management system developed by Microsoft. As a database, it is a software product whose primary function is to store and retrieve data as requested by other software applications, be it those on the same computer or those running on another computer across a network.

In our project we use SQL server to store the user information such as usernames, passwords and all the details user enters during the registration. We also maintain the details of the bucket assigned to each user in Amazon S3 and the file paths in the user's local system.

## 2.2 Requirements:

Below mentioned are the software requirements for our project:

**2.2.1 User Registration:** This provides the registration feature for a first-time user. The registration part is included in the Client application. Any new user, who is using the application for the first time can create an account with this feature.

- **Expected input:** User Details
- **Expected output:** New user account.

**2.2.2 Manage File Path:** After user login, the user is presented with two tabs: Upload and Set Manage File Path. Manage file path allows the user to specify the file path in their local systems. These files will be uploaded into the user's bucket at the time specified in Upload tab.

- **Expected input:** File paths of the user files in local machine
- **Expected output:** File paths and the last modified dates are displayed in a table.

**2.2.3 Upload:** Upload tab enables the users to specify the time at which the files need to be uploaded into the bucket. Time must be given in 24 hour-format. It also provides with an "Upload Now" option, which enables the application to upload the files immediately.

- **Expected input:** Time in 24 hours format
- **Expected output:** Files are uploaded to the Amazon S3 bucket.

**2.2.4 Web Application Login:** It enables the user to login to the web application, to view the uploaded files and download them if needed.

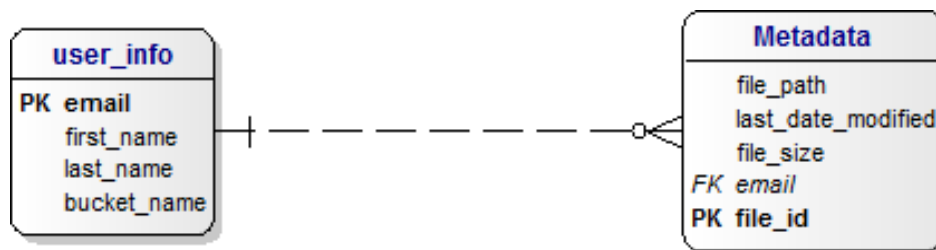
- **Expected input:** Username and password.
- **Expected output:** All the uploaded files are displayed and a download option is provided.

**2.2.5 File Download:** The files come with a download option. When the user clicks the download option, the file gets saved into the local machine.

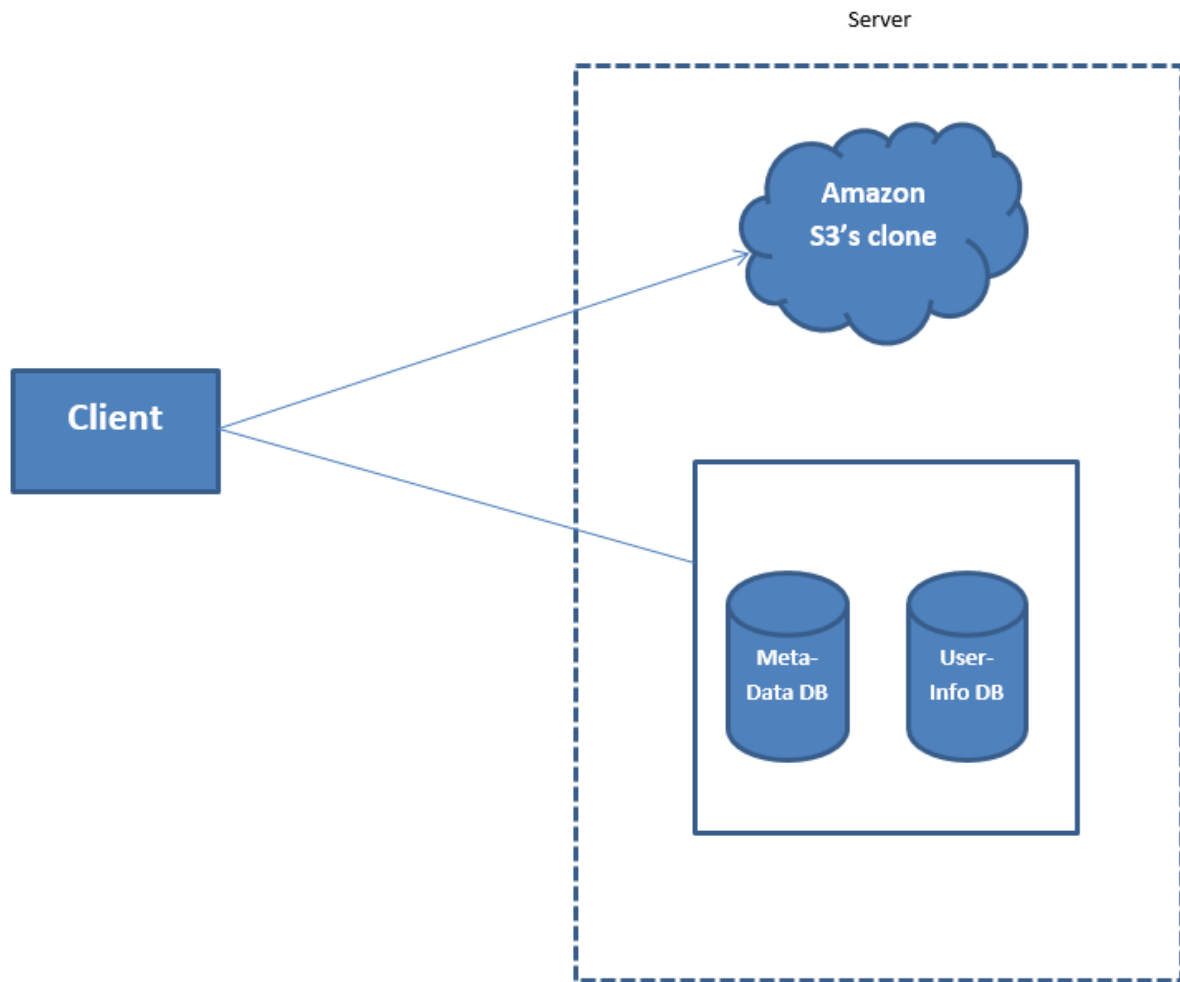
- **Expected input:** User prompt
- **Expected output:** The corresponding file is downloaded to user's local machine.

### 2.3 Architecture:

- **Database Design:**



- **Data Flow Diagram:**



### 2.4 Implementation:

**Client Application:** The client application is a desktop application developed in .Net 4.5 framework.

- **User Registration:** We have created a bucket for every user at the time of registration. We have used the AWS SDK to be able to access the buckets in Amazon S3. The code snippets for creating a bucket in Amazon S3 are as mentioned below:

```
PutBucketRequest request = new PutBucketRequest();
    request.BucketName = bucketname;
    request.UseClientRegion = true;
    client.PutBucket(request);
    return true;
```

- **Manage File Path:** We bind all the file paths to a ListView, whenever the user adds a new file path, we bind that to the List. The code for binding the file paths is mentioned below:

```
string filepath = @tbNewFilePath.Text.Trim();
string filename = System.IO.Path.GetFileName(filepath);
lblFileFeedBack.Visibility = Visibility.Visible;
if (!CheckFileAlreadyExists(filepath))
{
    lblFileFeedBack.Content = "File already exists";
    tbNewFilePath.Focus();
}
else
{
    AddUpdate(filepath, filename);
}
```

- **Upload:** We upload all the files to the user's bucket in Amazon S3. For this we use the methods available in AWS SDK. Code snippet is mentioned below:

```
bool uploaded = false;
List<FileMetaData> files = dataservice.GetUploadFileList(_user.UserEmail);
foreach (FileMetaData file in files)
{
    if (file.LastModified.CompareTo(DateTime.Now) != 0)
    {
        if (amzservice.UploadFile(_user.BucketName, file.FileName,
file.FilePath))
        {
            uploaded =
Convert.ToBoolean(dataservice.UpdateLastModified(file.FileID));
        }
    }
}

public bool UploadFile(string bucketName, string fileName, string filePath)
{
    if (CheckBucketAndFilePath(bucketName, fileName, filePath))
    {
        return WritingAnObject(bucketName, fileName, filePath);
    }
    return false;
}
```

### Web Application:

- **Web Application Login:** This feature enables the user to login to the web application. The user enters the username and password.

```
string email = tbUName.Text;
string password = tbPwd.Text;
User user = DataServices.LoginUser(email, password);
if (user != null)
{
    Session["User"] = user;
```



```
        Response.Redirect("FilesDisplay.aspx");
    }
    else
    {
        lblFeedBack.Visible = true;
        lblFeedBack.Text = "Sorry! Please enter existing emailid/password.";
    }
}
```

- **File Download:** The users can download the files uploaded in their bucket to their local system using this feature. The code snippet is as mentioned below:

```
string dest = Path.Combine(HttpRuntime.CodegenDir, fileName);
IAmazonS3 client;
using (client = Amazon.AWSClientFactory.CreateAmazonS3Client())
{
    try
    {
        GetObjectRequest request = new GetObjectRequest()
        {
            BucketName = bucketName,
            Key = fileName
        };
        using (GetObjectResponse response = client.GetObject(request))
        {
            if (!File.Exists(dest))
            {
                response.WriteResponseStreamToFile(dest);
            }
        }
    }
    catch (AmazonS3Exception amazonS3Exception)
    {
        if (amazonS3Exception.ErrorCode != null &&
            (amazonS3Exception.ErrorCode.Equals("InvalidAccessKeyId")
            ||
            amazonS3Exception.ErrorCode.Equals("InvalidSecurity")))
        {
            Console.WriteLine("Please check the provided AWS Credentials.");
            Console.WriteLine("If you haven't signed up for Amazon S3, please visit http://aws.amazon.com/s3");
        }
        else
        {
            Console.WriteLine("An error occurred with the message '{0}' when reading an object", amazonS3Exception.Message);
        }
    }
}
HttpContext.Current.Response.Clear();
HttpContext.Current.Response.AppendHeader("content-disposition",
"attachment; filename=" + fileName);
HttpContext.Current.Response.ContentType = "application/octet-stream";
HttpContext.Current.Response.TransmitFile(dest);
HttpContext.Current.Response.Flush();
HttpContext.Current.Response.End();
```

```
// Clean up temporary file.  
System.IO.File.Delete(dest);
```

- **Deployment:** We created a Windows server 2012 EC2 instance. We had deployed our database onto the Microsoft SQL Server 2008 on EC2 instance. The connection string for the database on the EC2 is as follows:

```
<connectionStrings>  
  <add name="GridProject" connectionString="Data Source = 50.19.182.199;Initial  
Catalog = GridProject;Integrated Security=false;User ID=grid; Password=grid"  
providerName="System.Data.SqlClient"/>  
</connectionStrings>
```

We hosted our Web application in IIS 6.0. The URL for accessing the application globally is [50.19.182.199/CloudApp/Web/Login.aspx](http://50.19.182.199/CloudApp/Web/Login.aspx)

### 2.5 Contribution Report:

Task	Contributor	Status
Database Design	Manoj Kidambi	Completed
User Interface Design	Saitej Erupaka (Windows) Bhagyasri Pavuluri (Web)	Completed
Windows Application	Bhagyasri Pavuluri Saitej Erupaka	Completed
Web Application	Adithya Nayabu Manoj Kidambi	Completed
Deployment	Adithya Nayabu Saitej Erupaka	Completed
Report	Adithya Nayabu Bhagyasri Pavuluri Manoj Kidambi	Completed

### 3. Results

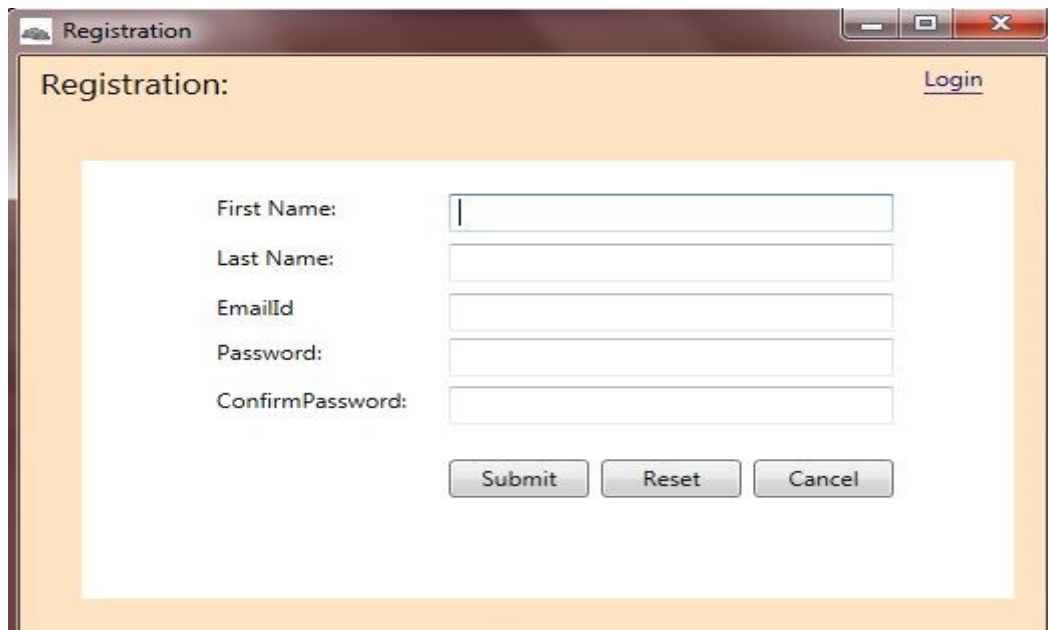
#### 3.1 Requirements:

All the requirements specified have been successfully met. The user was able to successfully register with the application. When the user is successfully registered with the application we had assigned him with a bucket. The user was able to manage his local file paths. All the file paths specified were successfully uploaded to his bucket at certain time. The user was even able to upload all the files to the bucket by pressing the upload button. The user was able to view the list of files in his bucket and download them from a remote location.

#### 3.2 Data:

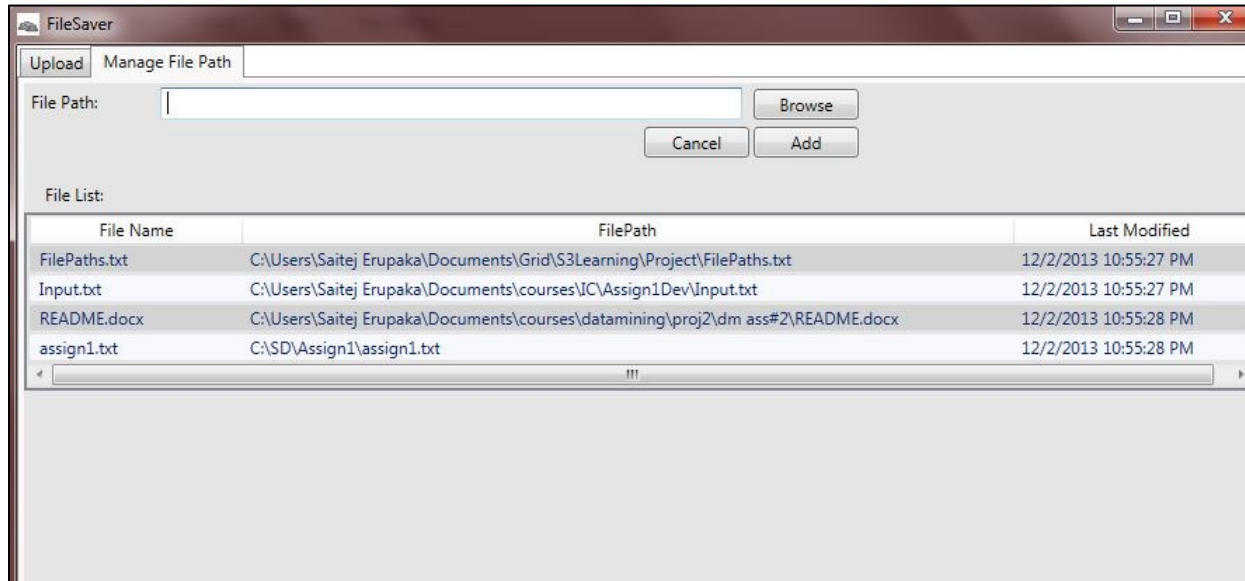
The input data and the results for each of the requirements listed in Section 2.2 are mentioned below:

- **User Registration:**
  - **Input:** User Details
  - **Screenshot:**



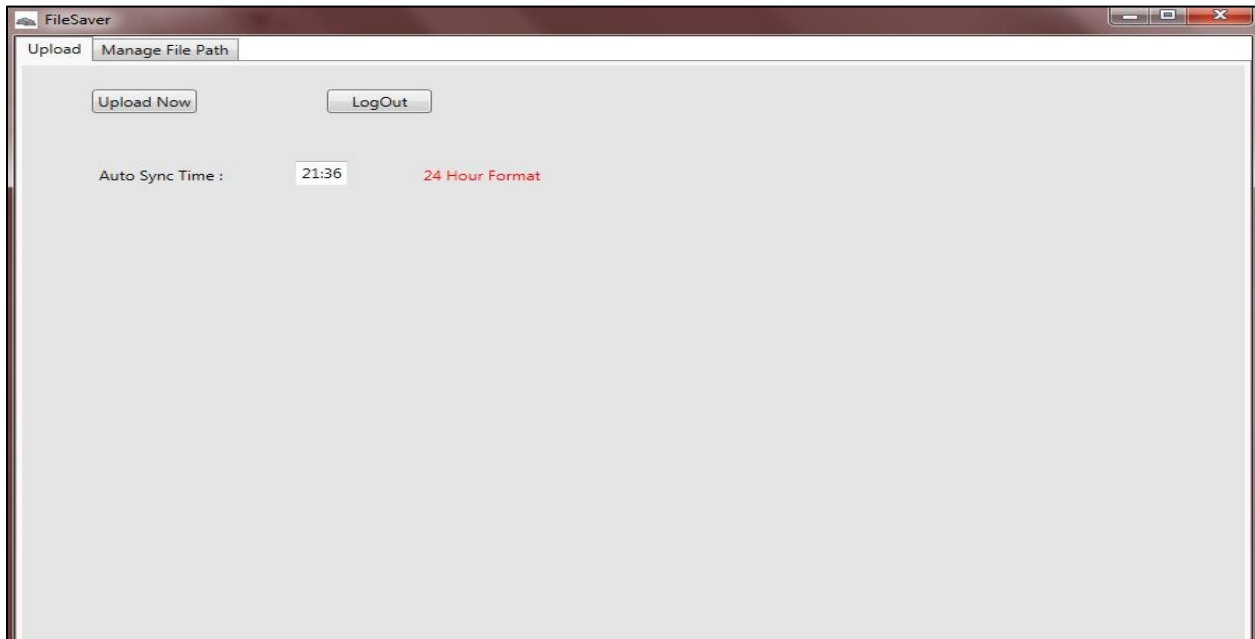
**Fig: 1 User Registration window in Client Application**

- **Output:** New user account
- **Manage File Path:**
  - **Input:** File paths of the user files in local machine
  - **Screenshot:**



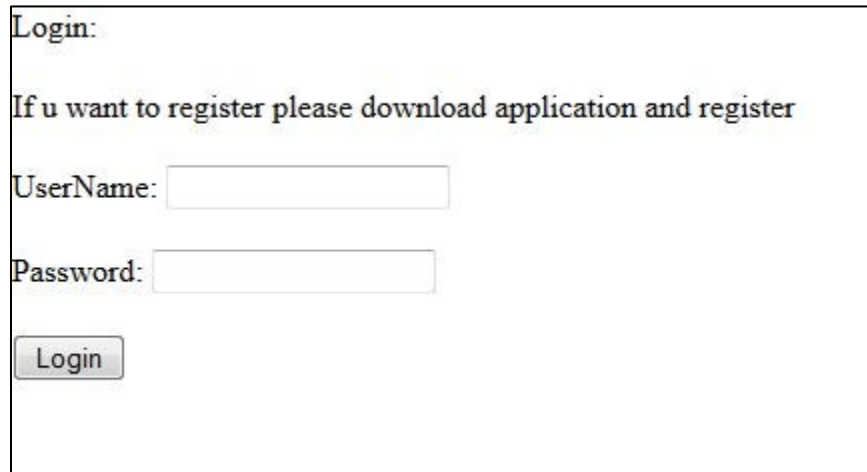
**Fig: 2 Manage File Path window in Client Application**

- **Output:** List of files already uploaded and also the files ready for upload are displayed.
- **Upload:**
  - **Input:** Time in 24 hours format
  - **Screenshot:**



**Fig: 3 Upload Time window in Client Application**

- **Output:** Files are uploaded to the Amazon S3 bucket.
- **Web Application Login:**
  - **Input:** Username and password
  - **Screenshot:**



Login:

If u want to register please download application and register

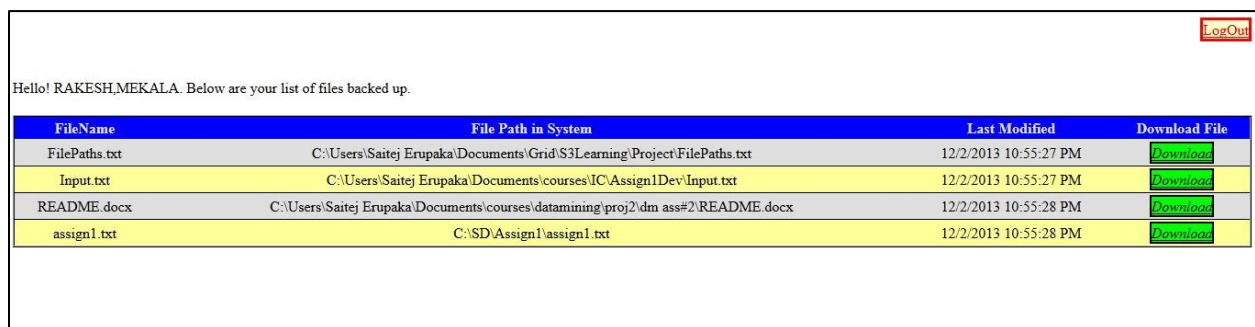
UserName:

Password:

Login

**Fig: 4 User Login window in Web Application**

- **Output:** All the uploaded files are displayed and a download option is provided.
- **File Download:**
  - **Input:** User Prompt
  - **Screenshot:**



Hello! RAKESH,MEKALA. Below are your list of files backed up.

FileName	File Path in System	Last Modified	Download File
FilePaths.txt	C:\Users\Saitej Erupaka\Documents\Grid\S3Learning\Project\FilePaths.txt	12/2/2013 10:55:27 PM	<a href="#">Download</a>
Input.txt	C:\Users\Saitej Erupaka\Documents\courses\IC\Assign1Dev\Input.txt	12/2/2013 10:55:27 PM	<a href="#">Download</a>
README.docx	C:\Users\Saitej Erupaka\Documents\courses\datamining\proj2\dm ass#2\README.docx	12/2/2013 10:55:28 PM	<a href="#">Download</a>
assign1.txt	C:\SD\Assign1\assign1.txt	12/2/2013 10:55:28 PM	<a href="#">Download</a>

LogOut

**Fig: 5 User Registration window in Client Application**

- **Output:** The corresponding file is downloaded to user's local machine.

### 4.0 Results:

Our project was successful in managing the files to be uploaded from different locations.

- In applications like Dropbox or Google Drive needs to have all the files to be uploaded in a single folder, but our application uploads all the files present in the file paths list to be uploaded to the cloud automatically.
- Dropbox doesn't provide an option to specify the time for upload. But in our application, the user can either backup the files at that moment or can schedule the backup for a later time.
- We have provided a user-friendly web interface to sign-in and download the files.

### Conclusions:

We had successfully completed the project. We were able to get familiarize ourselves with S3 API. We were able to learn about the functionalities of the Amazon S3 and Amazon Ec2. We learned about how to create instances and manage security groups in Amazon EC2. We deployed our application using IIS.

### 5.0 References:

1. IIS [http://en.wikipedia.org/wiki/Internet\\_Information\\_Services](http://en.wikipedia.org/wiki/Internet_Information_Services)
2. Amazon EC2 <http://aws.amazon.com/ec2/>
3. Amazon S3 <http://aws.amazon.com/s3/>
4. .Net Framework [http://en.wikipedia.org/wiki/.NET\\_Framework](http://en.wikipedia.org/wiki/.NET_Framework)
5. Amazon S3 API <http://docs.aws.amazon.com/AmazonS3/latest/API/Welcome.html>