Project Report Review-1

on

DOC Upload

Submitted in partial fulfillment of the

requirement for the award of the degree of

Bachelor of Technology in Computer Science and Engineering

2nd Year (4th Semester)

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA
INDIA
MARCH 2023

Name of Supervisor : Dr. Anupam Kumar Sharma
Designation : Professor

Submitted By :

Aman Shukla (21SCSE1011658)
Kanika Yadav (21SCSE1011084)
Sakshi Srivastava (21SCSE1011610)
Arnav Kumar Gupta (21SCSE1011650)

CERTIFICATE

The Final ProjectViva-Voce examination of ~~Name: Admission No~~ has been held on __________________ and his/her work is recommended for the award of ~~Name of Degree.~~

Signature of Examiner(s) **Signature of Supervisor(s)**

Signature of Program Chair **Signature of Dean**

Date: 29 March, 2023

Place: Greater Noida

GALGOTIAS UNIVERSITY, GREATER NOIDA

CANDIDATE'S DECLARATION

We hereby certify that the work which is being presented in the project, entitled **"DOC Upload"** in partial fulfillment of the requirements for the award of the ~~Name of Degree~~ submitted in the School of Computing Science and Engineering of Galgotias University, Greater Noida, is an original work carried out during the period of month, Year to Month and Year, under the supervision of Dr. Anupam Kumar Sharma, Professor, Department of Computer Science and Engineering , of School of Computing Science and Engineering , Galgotias University, Greater Noida

The matter presented in the project has not been submitted by us for the award of any other degree of this or any other places.

~~Name, Admission No~~

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Dr. Anupam Kumar Sharma

Professor

Abstract

1. Existing Problem

- Sometimes the user wants to upload a file larger than the available limit which occurs an error while uploading the file.
- You cannot upload files with names containing Unicode characters (multibyte) because of character set mismatch at the Operator System level or Application level.
- Files uploaded with same names are overridden by one another. This occurs because the *ServiceExport*directory is common for all the users. If two users upload files with same name, then one of the user's files is overridden by the other file.

2. Proposed Solution

- You may want to adjust the file size limit of your upload forms to allow more than the default value. However, the situation gets complicated if you've integrated your forms with Dropbox or Google Drive.
- User has to change the file name uploading the file because as it is an default error and didn't allow you to upload the file .
- User can give the file_name a different background colour, or add a [Hidden] tag to the file.

3. Tools and Technology Used

- Tool used - Intellij idea
- Technology used - Java 11, servlet , HTML, CSS

4. Results and output

- File upload components value attribute associated with the UploadFile instance.
- Using of fileUpload requires including the fileUpload component within a form, its enctype is **multipart/form-data**.
- Dummy action provided has used to print out the name and size of the uploaded file.

Where, the result of demo will be: Simple input button has been rendered into your browser.

5. Conclusion and Future Scope

The file upload is an essential component to make a form that store some image kind of data. It helps in applications using image upload or in the file sharing. This file-upload component uses *file.io* API for uploading file and in return it provides a shareable link. Furthermore, we can send get request to shareable link to get the file but for now, our only focus is on upload section so we only use the post method.

In future we are planning to upgrade it by implementing Dropbox, Gdrive using Google sdk.
And we can also store our files on AWS S3 using AWS sdk.
As this will allow user to save their data permanently without any hassle.

INTRODUCTION

File uploading is a feature for accepting and managing user's files: images, videos, PDFs, or other documents. Most websites need this feature for user-generated content, for instance, product photos on eBay, user avatars on Facebook, mood boards on Pinterest, CVs or portfolios on Indeed, homework assignments on Coursera, videos on Youtube, and so on.

In its simplest form, users upload files from their device storage. To make file uploading convenient for every user, you need to integrate a wide range of upload sources. Most common are:

- **The local webcam** – a user can take an instant photo (for an avatar, for instance) via a computer or mobile device camera and upload it right away.
- **Social media** – uploading files from Facebook, Instagram, etc.
- **Any remote URL** – uploading by copying and pasting a publicly available link into the upload dialog.
- **Cloud storage** – uploading from Google Drive, Dropbox, OneDrive, Box, etc.
- **Other services** like Evernote, Flickr, Huddle, etc.

Not every website needs all of them, but giving a user several options always makes your website more user-friendly.

How to Upload Files in a Browser

1. In order to build an Angular file upload component, we need to first understand how to upload files in plain HTML and Javascript *only,* and take it from there.

2. The key ingredient for uploading files in a browser is a plain HTML input of type file.This input will allow the user to open a browser file selection dialog and select one or more files (by default, only one file).
3. With this file input box, you can select a file from your file system, and then with a bit of Javascript, you can already send it to a backend.

Angular File Upload

- The best way to handle file upload in Angular is to build one or more custom components, depending on the supported upload scenarios.

- A file upload component needs to contain internally an HTML input of type file, that allows the user to select one or more files from the file system.

- This file input should be hidden from the user as it's not styleable and replaced by a more user-friendly UI.

- Using the file input in the background, we can get a reference to the file via the change event, which we can then use to build an HTTP request and send the file to the backend.

What is Interface in Angular

Interface is a specification that identifies a related set of properties and methods to be implemented by a class. So basically using interface you can set some basic rules for your properties and methods using class.

Sometime we are creating object array with specific datatype field like id has to be integer or number, name has to be string value, birth of date have date datatype data. but sometime we might have misteck and set string value instead of integer or number then it cought problem and your app will show you error. But if you use interface then it will solve problem when you write code on your IDE. IDE will show you error quiky where is a problem.

How to define interface in Angular:

Here, i will show you very simple example how to define interface class in angular. you can see bellow code for defining interface.

```
export interface Student {

    id: number;

    name: string;

}
```

- **export:**The export keyword will help to use import this class on other component and class.
- **interface:**The interface ia a keyword so using this keyword you can set interface.
- **Student:** Student is a class name that describe interface details.

LITERATURE SURVEY

Websites that utilize file upload functionality usually display a message on their web pages: “Select or drag-and-drop files to upload.” However, this varies depending on what the website needs from the user. Sometimes, a user may be asked to upload documents, media, etc.
File upload forms allow site visitors to select files from their computers and upload them to your website.

How Does a File Upload ?

Since a file upload form is part of the website, we need a basic understanding of how websites and their pages work to know how it is displayed and programmed. A website consists of two interacting components: frontend and backend.

Frontends are created using various web technologies. One major component used in creating frontends is the HyperText Markup Language (HTML).

HTML is used to specify which elements appear on the web page. Example elements include links, images, and, you guessed it: file upload forms.

Functionality

Making the File Upload

All things working under the hood to keep your website up and running are known as the backend. Like the frontend, many web technologies can be used to program the backend. One of the responsibilities of the backend is storing files uploaded by your website visitors through your file upload form.

Setting Up the Backend

There are two ways to set up the backend once the frontend is complete. The first one is to go the traditional way: program the destination URL to accept the files and upload them to your web server. Alternatively, you may use an external service to handle the backend for you and host your files outside your website.

Building the UI of a File Upload

Because a plain input of type file is impossible to style properly, what we end up doing is hiding it from the user, and then building an alternative file upload UI that uses the file input behind the scenes.
This user interface is split up into two separate parts. On top, we have a plain file input, that is used to open the file upload dialog and handle the change event.

Below the hidden file input, we have the file-upload container div, which contains the actual UI that the user will see on the screen.
As an example, we have built this UI with Angular Material components, but of course, the alternative file upload UI could take any form that you like.

This UI could be a dialog, a drag and drop zone, or like in the case of our component, just a styled button.

Results and Discussion

File uploading means a user from client machine requests to upload file to the server. For example, users can upload images, videos, etc on Facebook, Instagram, etc.

Uploaded files represent a significant risk to applications. The first step in many attacks is to get some code to the system to be attacked. Then the attack only needs to find a way to get the code executed. Using a file upload helps the attacker accomplish the first step.

The consequences of unrestricted file upload can vary, including complete system takeover, an overloaded file system or database, forwarding attacks to back-end systems, client-side attacks, or simple defacement. It depends on what the application does with the uploaded file and especially where it is stored.

There are really two classes of problems here. The first is with the file metadata, like the path and file name. These are generally provided by the transport, such as HTTP multi-part encoding. This data may trick the application into overwriting a critical file or storing the file in a bad location. You must validate the metadata extremely carefully before using it.

The other class of problem is with the file size or content. The range of problems here depends entirely on what the file is used for. See the examples below for some ideas about how files might be misused. To protect against this type of attack, you should analyse everything your application does with files and think carefully about what processing and interpreters are involved.

Conclusion and Future Scope

Allowing an end user to upload files to your website is like opening another door for a malicious user to compromise your server. However, uploading files is a necessity for any web application with advanced functionality. Whether it is a social networking site like Facebook and Twitter, or an intranet document sharing portal, web forums and other tools to upload images, videos and numerous other file types.

Unfortunately, uploaded files represent a significant risk to applications. Any attacker wants to find a way to get a code onto a victim system, and then looks for a way to execute that code.

The most important thing is to keep uploaded files in a location that can't access though the Internet. This can be done either by storing uploaded files outside of the web root or configuring the web server to deny access to the uploads directory.

A developer must be careful not to expose applications to attack while implementing file upload functionalities because poorly designed tools for file upload implementation can lead to vulnerabilities such as remote code execution.

REFERENCES

https://www.geeksforgeeks.org/angular-file-upload/

https://blog.angular-university.io/angular-file-upload/

https://resources.infosecinstitute.com/topic/file-upload-vulnerabilities/

https://blog.powr.io/how-to-create-a-file-upload-form-in-html?hsLang=en