To create a web page using Bootstrap 5 where users can choose a CSV file and submit it, you need an HTML form with a file input and a submit button. Here's a simple example:

### Step 1: Include Bootstrap 5

Include Bootstrap 5 CSS and JS in your HTML file:

html

Copy code

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>File Upload Form</title>

<link href="https://stackpath.bootstrapcdn.com/bootstrap/5.1.3/css/bootstrap.min.css" rel="stylesheet">

</head>

<body>

<div class="container mt-5">

<h2>Upload CSV File</h2>

<form id="uploadForm" enctype="multipart/form-data">

<div class="mb-3">

<label for="formFile" class="form-label">Choose CSV file</label>

<input class="form-control" type="file" id="formFile" name="file" accept=".csv">

</div>

<button type="submit" class="btn btn-primary">Submit</button>

</form>

<div id="result" class="mt-5"></div>

</div>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"></script>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

<script>

$(document).ready(function () {

$('#uploadForm').on('submit', function (e) {

e.preventDefault();

var formData = new FormData(this);

$.ajax({

url: '/upload', // Replace with your server URL

type: 'POST',

data: formData,

processData: false,

contentType: false,

success: function (response) {

// Display response (e.g., HTML table)

$('#result').html(response);

},

error: function (jqXHR, textStatus, errorThrown) {

console.error(textStatus, errorThrown);

alert('File upload failed!');

}

});

});

});

</script>

</body>

</html>

### Step 2: Backend to Handle File Upload

You need a backend to handle the file upload, read the CSV file, and generate the HTML table. Here's an example using Java with Spring Boot:

#### Spring Boot Backend

1. **Add Dependencies**: Ensure your pom.xml includes the necessary dependencies:

xml

Copy code

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.apache.commons</groupId>

<artifactId>commons-csv</artifactId>

<version>1.9.0</version>

</dependency>

1. **Controller to Handle File Upload**:

java

Copy code

import org.apache.commons.csv.CSVFormat;

import org.apache.commons.csv.CSVRecord;

import org.springframework.web.bind.annotation.\*;

import org.springframework.web.multipart.MultipartFile;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.Reader;

import java.util.List;

@RestController

public class FileUploadController {

@PostMapping("/upload")

public String handleFileUpload(@RequestParam("file") MultipartFile file) throws IOException {

if (file.isEmpty()) {

return "Please upload a valid CSV file.";

}

Reader reader = new InputStreamReader(file.getInputStream());

List<CSVRecord> records = CSVFormat.DEFAULT.withFirstRecordAsHeader().parse(reader).getRecords();

return generateHtmlTable(records);

}

private String generateHtmlTable(List<CSVRecord> records) {

StringBuilder html = new StringBuilder();

html.append("<table class=\"table\">")

.append("<thead><tr>");

if (!records.isEmpty()) {

CSVRecord header = records.get(0);

for (String column : header.toMap().keySet()) {

html.append("<th>").append(column).append("</th>");

}

html.append("</tr></thead><tbody>");

}

for (CSVRecord record : records) {

html.append("<tr>");

for (String value : record) {

html.append("<td>").append(value).append("</td>");

}

html.append("</tr>");

}

html.append("</tbody></table>");

return html.toString();

}

}

1. **Spring Boot Application**:

java

Copy code

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class FileUploadApplication {

public static void main(String[] args) {

SpringApplication.run(FileUploadApplication.class, args);

}

}

### Running the Application

1. **Frontend**: Open index.html in your browser. This will present a file input and submit button.
2. **Backend**: Run the Spring Boot application. This will start a server that can handle file uploads at the /upload endpoint.

### Summary

* The HTML form allows users to choose a CSV file and submit it.
* jQuery handles the form submission and makes an AJAX request to the server.
* The Spring Boot backend reads the CSV file, converts it to an HTML table, and returns the HTML to be displayed on the page.

This solution integrates the frontend and backend to handle file uploads and display CSV content as an HTML table using Bootstrap 5 for styling.

4o

Top of Form

Bottom of Form