**Assisted Practice: 2.4 File Handling**

This section will guide you to:

* Set up Eclipse to work with Spring Boot using the STS plugin
* Create a data file that will be used for downloading
* Create an HTML file that will show a form of uploading a file
* Create a Controller that will handle file uploads and downloads

**Development Environment**

* Eclipse IDE for Enterprise Java Developers v2019-03 (4.11.0)
* Apache Tomcat Server v9.0
* JRE: OpenJDK Runtime Environment 11.0.2
* Spring Boot STS 4
* All other software components are configured automatically by Spring Boot

This lab has nine subsections, namely:

* + 1. Installing the STS Plugin in Eclipse
    2. Creating a Spring Boot Starter Project which is web enabled
    3. Creating dump.txt as a file for downloading
    4. Creating an HTML file that will show a form of uploading a file
    5. Creating MainController for handling file upload and download
    6. Building the project
    7. Publishing and starting the project
    8. Running the project
    9. Pushing the code to your GitHub repositories

**Step 2.4.1:** Installing the Spring Tool Suite Plugin in Eclipse

* Spring Tool Suite is already installed as an Eclipse plugin in your practice lab. (Refer FSD: Lab Guide - Phase 3 to verify the installation.)

**Step 2.4.2:** Creating a Spring Boot Starter Project which is web enabled

* Open Eclipse
* Go the **File** menu. Choose **New->Other**
* In the **Wizard** list, select **Spring Boot->Spring Starter Project**
* In **Name,** enter SpringFiles, **Type** as Maven, **Packaging** as Jar, **Group** as com.ecommerce, and **Package** as com.ecommerce
* Click on **Next**
* In the list of **Available** dependencies, scroll down to select **Web->Spring Web Starter**
* Click on **Next**
* Click on **Finish**
* This will create the project files in the Project Explorer

**Step 2.4.3:** Creating dump.txt as a file for downloading

* In the Project Explorer, expand **SpringFiles->src->main->resources->static**
* Right click on **static** and select **New->File**
* Enter **File Name** as dump.txt and click on **Finish**
* Add the following data:

This is a file which has been downloaded via Spring Boot

**Step 2.4.4:** Creating an HTML file that will show a form of uploading a file

* In the Project Explorer, expand **SpringFiles->src->main->resources->static**
* Right click on **static** and select **New->File**
* Enter **File Name** as **index.html** and click on **Finish**
* Add the following script:

<**html**>

<**head**><**title**>File Upload</**title**></**head**>

<**body**>

<**form** method="post" enctype="multipart/form-data" action="/upload">

Upload file&nbsp;

<**input** type="file" name="fileToUpload" id="fileToUpload"><**br**><**br**>

<**input** type="submit" value="Upload " name="submit">

</**form**>

</**body**>

</**html**>

**Step 2.4.5:** Creating MainController for handling file upload and download

* In the Project Explorer, expand **SpringFiles->src->main>java>com->ecommerce->controllers**
* Right click on **controllers** and click on **New->Other**
* In the **Wizard** list, choose **Class** and click on **Next**
* In **Name,** enter **MainController** and click on **Finish**
* Add the following code:

**package** com.ecommerce.controllers;

**import** java.io.File;

**import** java.io.FileInputStream;

**import** java.io.FileOutputStream;

**import** java.io.IOException;

**import** org.springframework.core.io.ClassPathResource;

**import** org.springframework.core.io.InputStreamResource;

**import** org.springframework.core.io.Resource;

**import** org.springframework.http.HttpHeaders;

**import** org.springframework.http.HttpStatus;

**import** org.springframework.http.MediaType;

**import** org.springframework.http.ResponseEntity;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.util.ResourceUtils;

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

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

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

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

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

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

**import** org.springframework.web.multipart.MultipartFile;

**@Controller**

**public** **class** MainController {

**@RequestMapping(value = "/")**

**public** **String** index() {

**return** "index.html";

}

**@RequestMapping(value = "/upload", method = RequestMethod.POST, consumes = MediaType.MULTIPART\_FORM\_DATA\_VALUE)**

**public** **String** fileUpload(**@RequestParam("file")** MultipartFile file) {

**String** result = "File was uploaded successfully";

**try** {

**File** convertFile = **new** **File**("/var/tmp/"+file.getOriginalFilename());

convertFile.createNewFile();

**FileOutputStream** fout = **new** **FileOutputStream**(convertFile);

fout.write(file.getBytes());

fout.close();

} **catch** (**IOException** iex) {

result = "Error " + iex.getMessage();

} **finally** {

**return** result;

}

}

**@RequestMapping(value = "/download", method = RequestMethod.GET)**

**public** ResponseEntity<**Object**> downloadFile() **throws** **IOException** {

**String** fileName = "static/dump.txt";

**ClassLoader** classLoader = **new** MainController().getClass().getClassLoader();

**File** file = **new** **File**(classLoader.getResource(fileName).getFile());

InputStreamResource resource = **new** InputStreamResource(**new** **FileInputStream**(file));

HttpHeaders headers = **new** HttpHeaders();

headers.add("Content-Disposition", **String**.format("attachment; filename=\"%s\"", file.getName()));

headers.add("Cache-Control", "no-cache, no-store, must-revalidate");

headers.add("Pragma", "no-cache");

headers.add("Expires", "0");

ResponseEntity<**Object**>

responseEntity = ResponseEntity.ok().headers(headers).contentLength(file.length()).contentType(

MediaType.parseMediaType("application/txt")).body(resource);

**return** responseEntity;

}

}

**Step 2.4.6:** Building the project

* From the **Project** menu at the top, click on **Build**
* If any compile errors are shown, fix them as required

**Step 2.4.7:** Publishing and starting the project

* In the Project Explorer, right click **SpringFiles->Run As->Spring Boot App**
* Check in the Eclipse Console for the message **Started SpringBootStarterApplication**

**Step 2.4.8:** Running the project

* To run the project, open a web browser and type [**http://localhost:8080**](http://localhost:8080/)to test file uploading
* Go to[**http://localhost:8080/download**](http://localhost:8080/download)to test file downloading

**Step 2.4.9:** Pushing the code to your GitHub repositories

* Open your command prompt and navigate to the folder where you have created your files.

**cd <folder path>**

* Initialize your repository using the following command:

**git init**

* Add all the files to your git repository using the following command:

**git add .**

* Commit the changes using the following command:

**git commit . -m “Changes have been committed.”**

* Push the files to the folder you initially created using the following command:

**git push -u origin master**