Problem: Implement a Restful API in spring-boot application to upload a file and save it local disk:

Project Files Description:

-> **pom.xml:** This project is developed by using maven build in tool. with maven we don't have to worry about dependency injections, project structure and build process. I have loaded all the required jars for spring boot application from spring repo "https://repo.spring.io/libs-release".
-> **Application.java:** This is a spring boot app. This let us run the spring application standalone. This is achieved through the annotation "@SpringBootApplication"
-> **UploadingFile.java:** This is a service layer for the application. When a rest call is made from the UI, Our service layer will be called and will make appropriate rest call based on the URL or method type. In this example I have used only POST method.
-> **Index.html:** This is simple UI used to upload a file.

Code Base:

pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://
maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/
maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>org.test</groupId>
  <artifactId>SpringBoot</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>1.5.6.RELEASE</version>
    </parent>

    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
```

```
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
        <dependency>
            <groupId>com.jayway.jsonpath</groupId>
            <artifactId>json-path</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>

    <properties>
        <java.version>1.8</java.version>
    </properties>


    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
        </plugins>
    </build>

    <repositories>
        <repository>
            <id>spring-releases</id>
            <url>https://repo.spring.io/libs-release</url>
        </repository>
    </repositories>
    <pluginRepositories>
        <pluginRepository>
            <id>spring-releases</id>
            <url>https://repo.spring.io/libs-release</url>
        </pluginRepository>
    </pluginRepositories>

</project>
```

Application.java

```
package test.restful;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class Application {
	public static void main(String[] args) {
		SpringApplication.run(Application.class, args);
	}
}
```

UploadingFile.java

```
package test.restful;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
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.RestController;
import org.springframework.web.multipart.MultipartFile;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;

@RestController
public class UploadingFile {
	// Save the uploaded file to this folder
```

```
	private static String UPLOADED_FOLDER = "/Users/macpro/test/";
	// @RequestMapping(method=RequestMethod.POST, value="/upload")
	@PostMapping("/upload")
	public String singleFileUpload(@RequestParam("file") MultipartFile file) {
		if (file.isEmpty()) {
			return "File is Empty. Please select a file to upload";
		}
		try {
			// Get the file and save it somewhere
			byte[] bytes = file.getBytes();
			Path path = Paths.get(UPLOADED_FOLDER + file.getOriginal-
Filename());
			Files.write(path, bytes);
		} catch (IOException e) {
			e.printStackTrace();
		}
		return "File "+file.getOriginalFilename()+" is Uploaded";
	}
}
```

index.html

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
	background-color:#AAAAAA;
	background-size: 100%;
	width: 100%;
}
</style>
</head>
<body>
<h1>Spring Boot file upload example</h1>
<form method="post" action="/upload" enctype="multipart/form-data">
    <input type="file" name="file"/><br/>
    <input type="submit" value="Upload" />
</form>
</body>
```

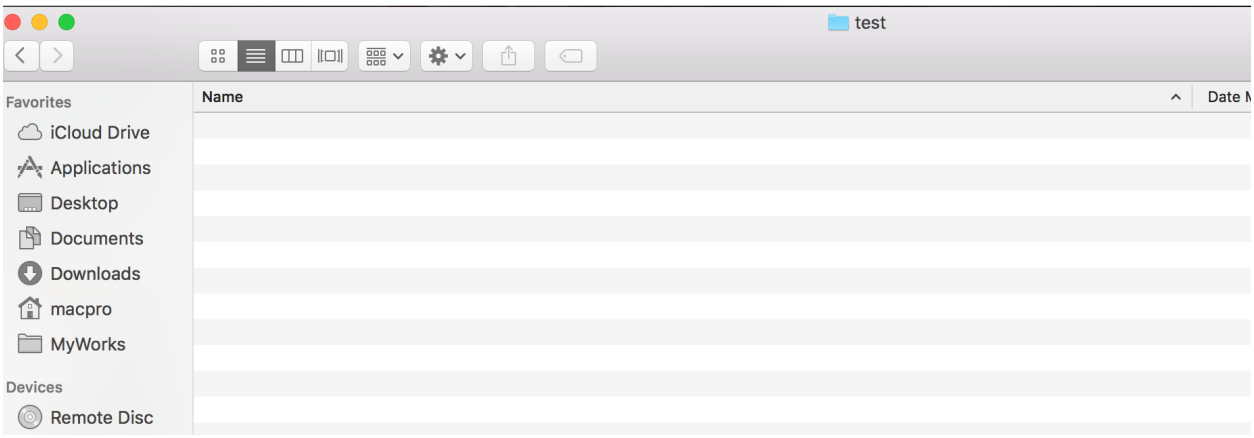
```
</html>
```

Outputs:

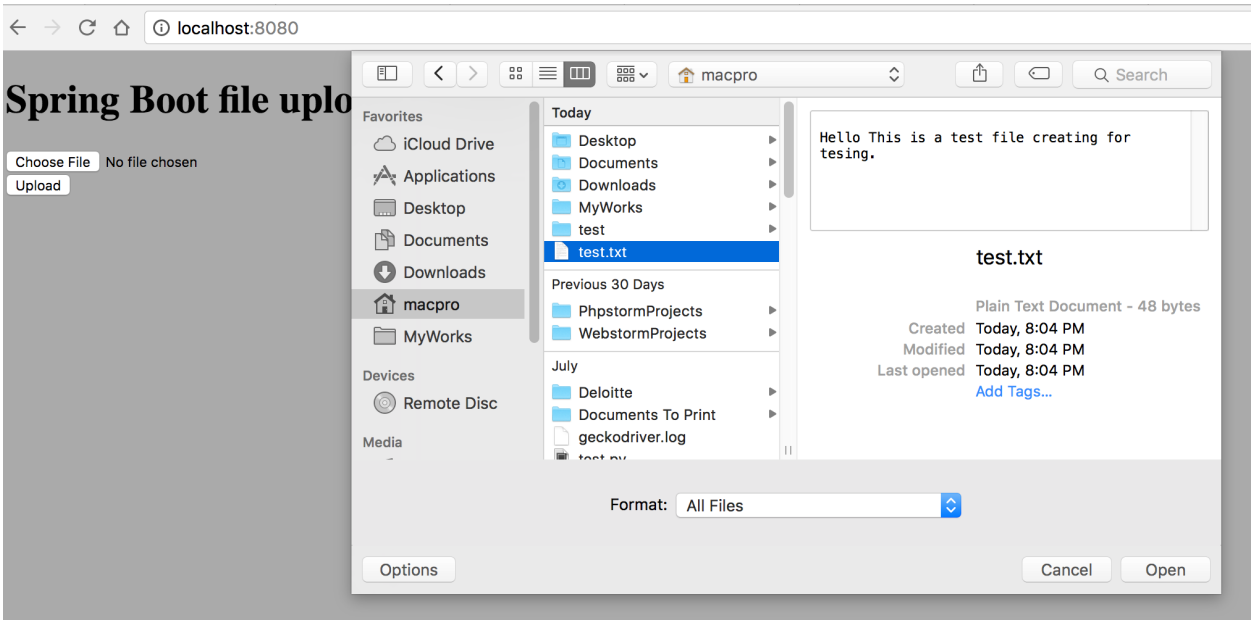
step1: run Application.java as normal java program, it should start all the required things for a spring web application. It will load all the jars required for the spring application and also it will start tomcat web server to provide response to web requests.

step2: Open browser and type "localhost:8080", it should load index.html which is present in the "static" folder of the project. Spring Boot looks for index.html files from static folder of a project. The screen should look like this,

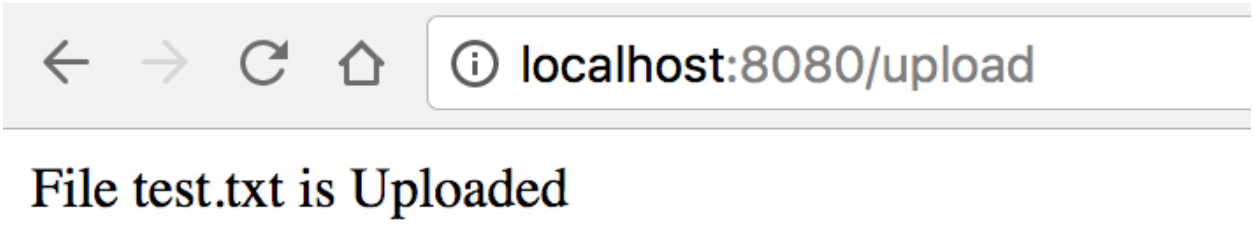
Step3: Choose File and click on Upload. Following screens will explain the process. Before we proceed let's check the "/Users/macpro/test/" folder, this is a empty folder. We will keep uploaded file into this empty folder.



Open the browser and click on browse to upload a file



Click on upload:



Let's check the "/Users/macpro/test/" now, you should see the uploaded file must be present in our test folder