Ex 5	**Implementation of a servlet for uploading and downloading files using Multipart Config and File Output Stream**

Aim

To Implement a servlet for uploading files to the server and downloading them using Multipart Config and File Output Stream.

Definitions

Servlet

A servlet is a Java-based software component that extends the capabilities of a server, most commonly a web server, to provide dynamic content. It functions by receiving and responding to client requests, typically over HTTP, and is frequently used to build web applications.

Uploading

Uploading refers to transmitting data from one computer system to another through means of a network. Common methods of uploading include: uploading via web browsers, FTP clients, and terminals. Uploading can be used in the context of clients that send files to a central server.

Downloading

Downloading is the process of transferring data from a remote computer or server to a local one, such as your own computer or mobile device. It's how you get files like music, movies, documents, and software from the internet onto your device. This is the opposite of uploading, which is sending data from your device to the internet.

Multipart Config

The @MultipartConfig annotation in Java Servlets is used to indicate that a Servlet expects requests to be made using the multipart/form-data MIME type. This is primarily used for handling file uploads within web applications.

File Output Stream

A FileOutputStream in Java is a byte-oriented output stream used for writing raw bytes to a file. It is part of the java.io package and extends the OutputStream abstract class.

Procedure

Open NetBeans IDE.

To create a Project go to File Menu → choose New Project → choose Java Web from Categories →choose Web Application from Projects →click next →specify the project name as FileHandling → Click Next → Click Next → click Finish.

Expand Web Pages Folder → Open index.html file → type the following codes:

index.html

```
<!DOCTYPE html>
<html>
<head>
   <title>File Upload</title>
</head>
<body>
   <h2>Upload File</h2>
   <form action="FileUploadDownloadServlet" method="post" enctype="multipart/form-data">
      Select File: <input type="file" name="fileUpload"/><br/><br/>
      <input type="submit" value="Upload"/>
   </form>

   <h2>Download File</h2>
   <form action="FileUploadDownloadServlet" method="get">
      File Name: <input type="text" name="filename"/><br/><br/>
      <input type="submit" value="Download"/>
   </form>
</body>
</html>
```

Right click on Source Packages Folder → choose New → select Servlet → specify the class name as FileUploadDownloadServlet → Click Next →click Finish.

Type the following codes in FileUploadDownloadServlet.java:

FileUploadDownloadServlet.java

```
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import javax.servlet.ServletException;
import javax.servlet.annotation.MultipartConfig;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.Part;

@WebServlet("/FileUploadDownloadServlet")
@MultipartConfig(
    fileSizeThreshold = 1024 * 1024 * 2, // 2MB
    maxFileSize = 1024 * 1024 * 10,      // 10MB
    maxRequestSize = 1024 * 1024 * 50    // 50MB
)
public class FileUploadDownloadServlet extends HttpServlet {

    private static final String UPLOAD_DIRECTORY = "uploads"; // Directory to save uploaded files

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
```

```
        String uploadPath = getServletContext().getRealPath("") + File.separator + UPLOAD_DIRECTORY;
        File uploadDir = new File(uploadPath);
        if (!uploadDir.exists()) {
            uploadDir.mkdir();
        }

        try {
            for (Part part : request.getParts()) {
                String fileName = extractFileName(part);
                if (fileName != null && !fileName.isEmpty()) {
                    String filePath = uploadPath + File.separator + fileName;
                    try (InputStream is = part.getInputStream();
                         FileOutputStream fos = new FileOutputStream(filePath)) {
                        byte[] buffer = new byte[1024];
                        int bytesRead;
                        while ((bytesRead = is.read(buffer)) != -1) {
                            fos.write(buffer, 0, bytesRead);
                        }
                    }
                    response.getWriter().println("File " + fileName + " uploaded successfully to " + uploadPath);
                }
            }
        } catch (Exception ex) {
            response.getWriter().println("Error uploading file: " + ex.getMessage());
        }
    }

    private String extractFileName(Part part) {
        String contentDisp = part.getHeader("content-disposition");
```

```
        String[] items = contentDisp.split(";");
        for (String s : items) {
            if (s.trim().startsWith("filename")) {
                return s.substring(s.indexOf("=") + 2, s.length() - 1);
            }
        }
        return null;
    }

    // ... (FileUploadDownloadServlet class definition continues)

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String fileName = request.getParameter("filename");
        if (fileName == null || fileName.isEmpty()) {
            response.getWriter().println("File name not provided for download.");
            return;
        }

        String uploadPath = getServletContext().getRealPath("") + File.separator +
UPLOAD_DIRECTORY;
        File downloadFile = new File(uploadPath + File.separator + fileName);

        if (!downloadFile.exists()) {
            response.getWriter().println("File not found: " + fileName);
            return;
        }

        response.setContentType(getServletContext().getMimeType(fileName));
        response.setHeader("Content-Disposition", "attachment; filename=\"" + fileName + "\"");
```

```
        response.setContentLength((int) downloadFile.length());

        try (InputStream is = new java.io.FileInputStream(downloadFile);
             java.io.OutputStream os = response.getOutputStream()) {
            byte[] buffer = new byte[1024];
            int bytesRead;
            while ((bytesRead = is.read(buffer)) != -1) {
                os.write(buffer, 0, bytesRead);
            }
        } catch (Exception ex) {
            response.getWriter().println("Error downloading file: " + ex.getMessage());
        }
    }
}
```

Right click on FileHandling Project → choose Clean and Build → right click again → Choose Deploy → right click again → choose Run.

You can see the following result in the output window.

Output

```
ant -f C:\\Users\\PGLAB\\Documents\\NetBeansProjects\\FileHandling -Dnb.internal.action.name=run -Ddirectory.deployment.supported=true -DforceRedeploy=false -Dnb.wait.for.caches=true -Dbrowser.context=C:\\Users\\PGLAB\\Documents\\NetBeansProjects\\FileHandling run

init:

deps-module-jar:

deps-ear-jar:

deps-jar:

library-inclusion-in-archive:

library-inclusion-in-manifest:

compile:

compile-jsps:

Incrementally deploying FileHandling
```

Completed incremental distribution of FileHandling

run-deploy:

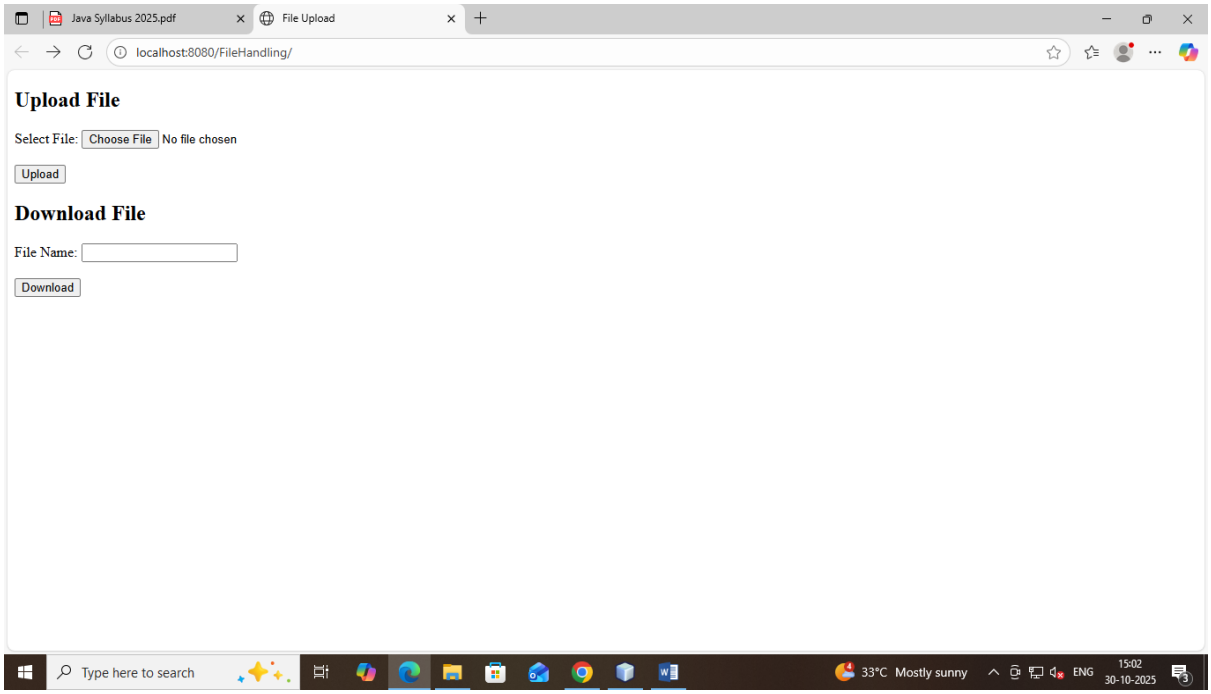
Browsing: http://localhost:8080/FileHandling

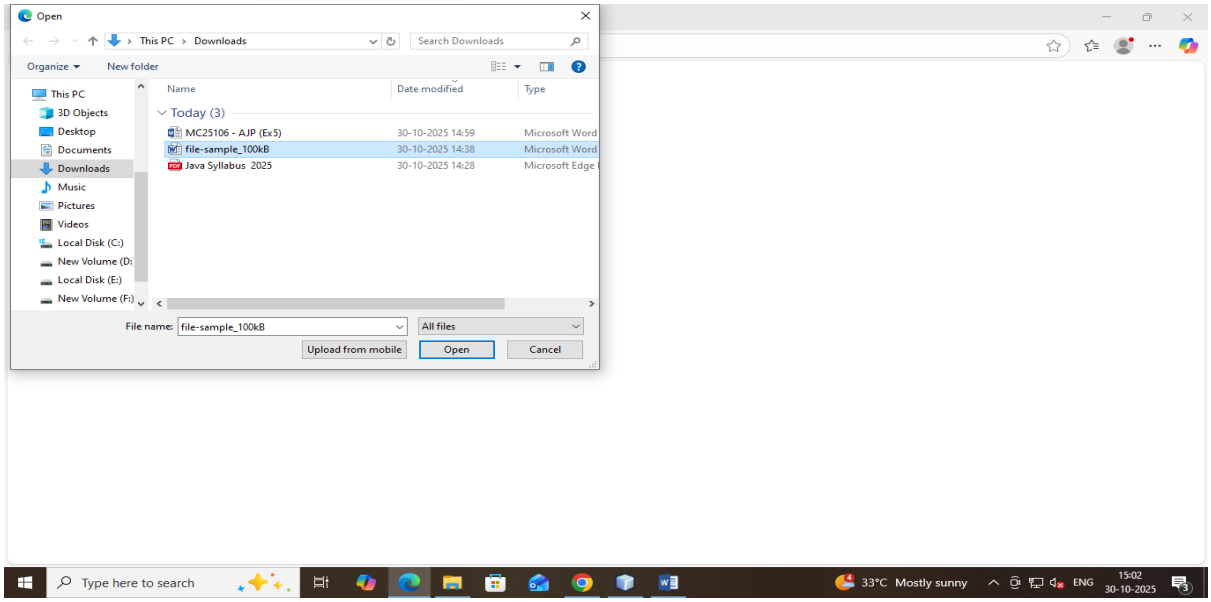
run-display-browser:

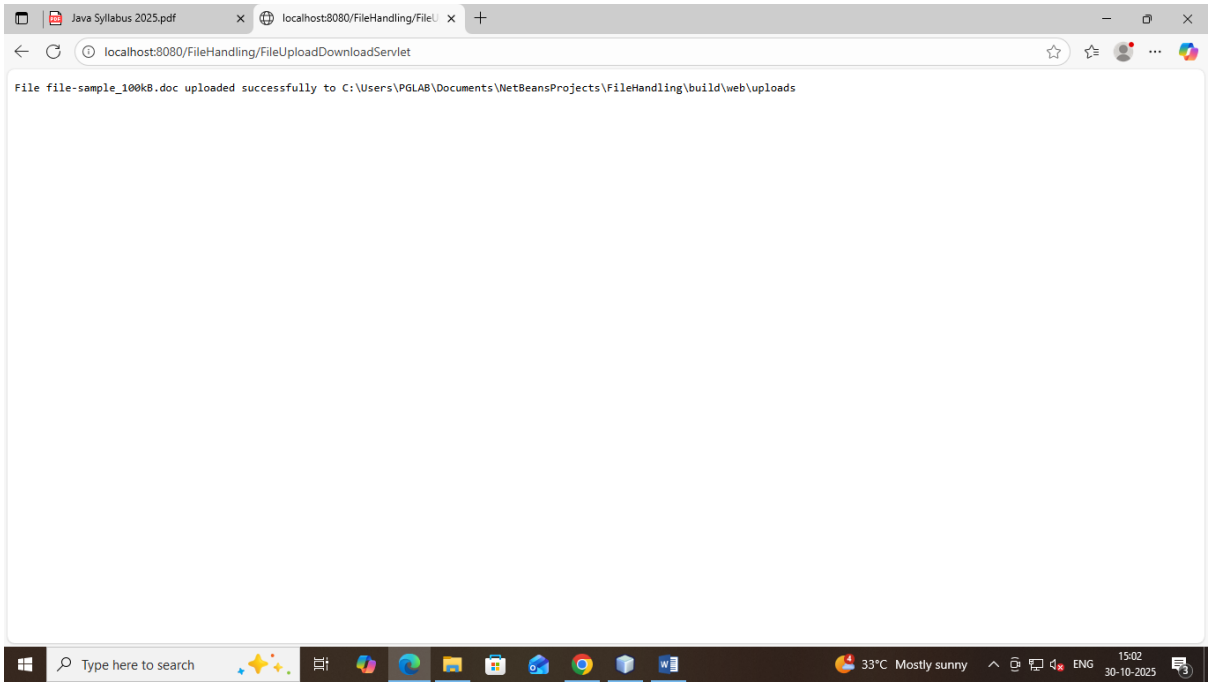
run:

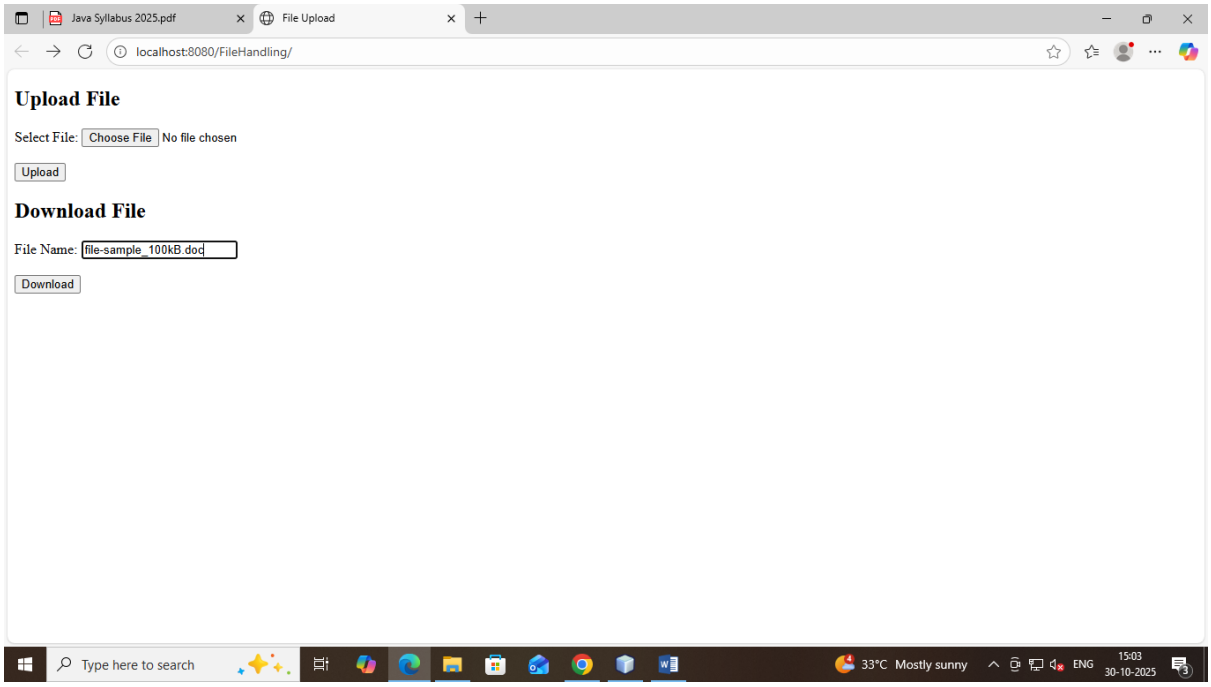
BUILD SUCCESSFUL (total time: 0 seconds)

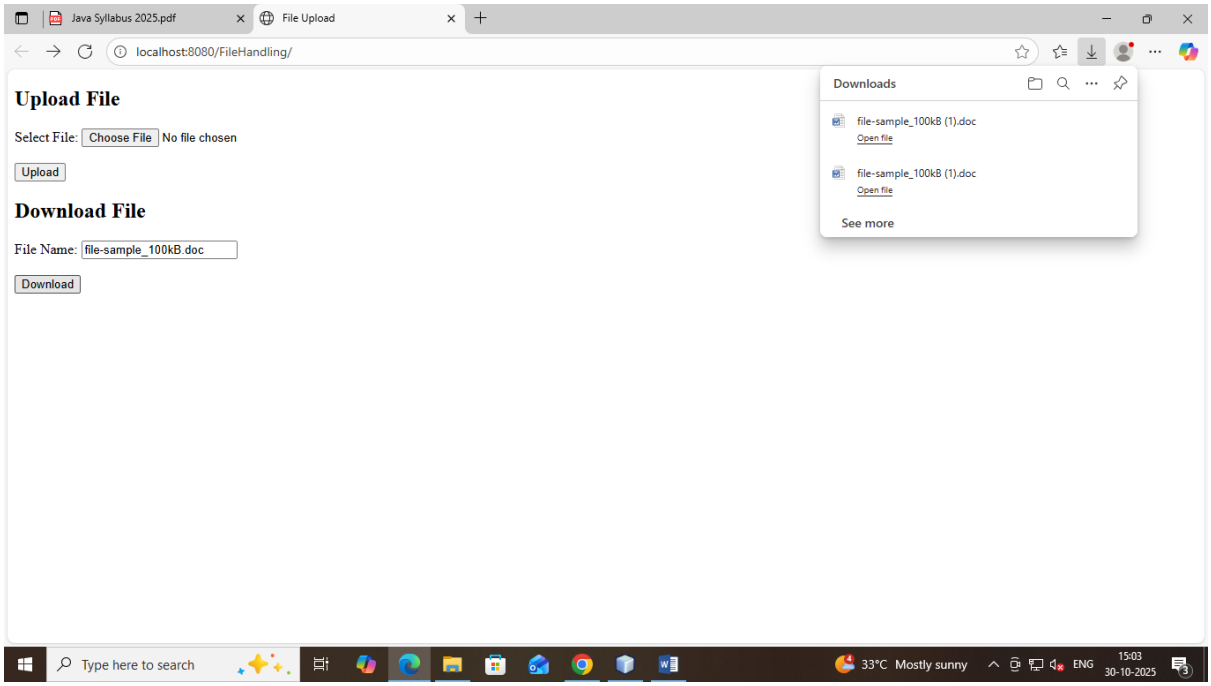
You can see the following result in the Browser.

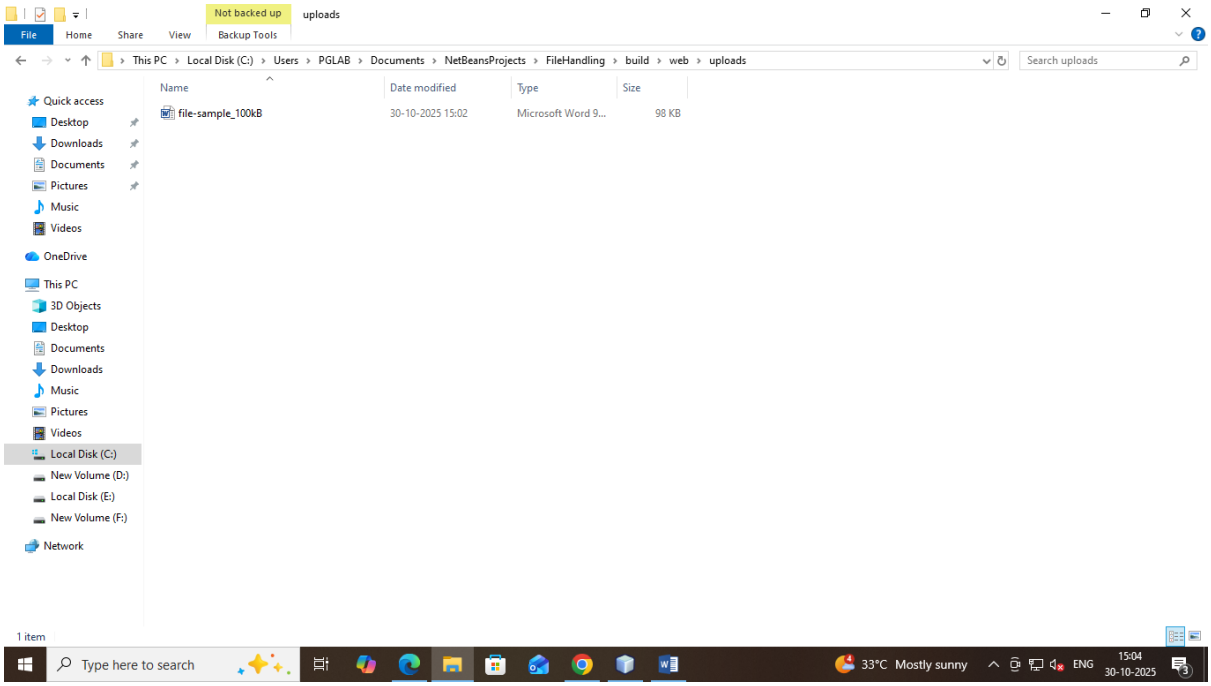





Java Syllabus 2025.pdf
localhost:8080/FileHandling/FileU
localhost:8080/FileHandling/FileUploadDownloadServlet
File file-sample_100kB.doc uploaded successfully to C:\Users\PGLAB\Documents\NetBeansProjects\FileHandling\build\web\uploads
Type here to search
33°C Mostly sunny
ENG
15:02
30-10-2025


Java Syllabus 2025.pdf
File Upload
localhost:8080/FileHandling/
Upload File
Select File: Choose File No file chosen
Upload
Download File
File Name: file-sample_100kB.doc
Download
Type here to search
33°C Mostly sunny
ENG
15:03
30-10-2025


Java Syllabus 2025.pdf
File Upload
localhost:8080/FileHandling/
Upload File
Select File: Choose File No file chosen
Upload
Download File
File Name: file-sample_100kB.doc
Download
Downloads
file-sample_100kB (1).doc
Open file
file-sample_100kB (1).doc
Open file
See more
Type here to search
33°C Mostly sunny
ENG
15:03
30-10-2025


Not backed up
uploads
File
Home
Share
View
Backup Tools
This PC › Local Disk (C:) › Users › PGLAB › Documents › NetBeansProjects › FileHandling › build › web › uploads
Search uploads
Name
Date modified
Type
Size
file-sample_100kB
30-10-2025 15:02
Microsoft Word 9...
98 KB
Quick access
Desktop
Downloads
Documents
Pictures
Music
Videos
OneDrive
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
New Volume (D:)
Local Disk (E:)
New Volume (F:)
Network
1 item
Type here to search
33°C Mostly sunny
ENG
15:04
30-10-2025

Result

Thus, a servlet for uploading files to the server and downloading files using Multipart Config and File Output Stream has been implemented.