**Instructions for Starting the Server and Client Setup**

**Overview**

You can connect multiple clients at a time and perform operations simultaneously. Exceptions and error messages are handled explicitly.

**Server Startup**

1. **Start RMI registry:**

* start rmiregistry

1. **Start the server:**

* java -cp pa2.jar FileServer start <portnumber>

**Client Startup**

1. **Set the server environment variable:**

* set PA1\_SERVER=localhost:<portnumber>
* Replace <portnumber> with the port number you want to use (e.g., 4098).

**File Operations**

**Upload a File**

* java -cp pa2.jar Client upload <client\_path> <server\_path>

**Download a File**

* java -cp pa2.jar Client download <server\_path> <client\_path>

**Directory Operations**

**List Directory Contents**

* java -cp pa2.jar Client dir <server\_path>

**Create a Directory**

* java -cp pa2.jar Client mkdir <server\_path>

**Remove an Empty Directory**

* java -cp pa2.jar Client rmdir <server\_path>

**Remove a File**

* java -cp pa2.jar Client rm <server\_path>

**Shutdown the Server**

* java -cp pa2.jar Client shutdown

**File Sharing System (FSS) Requirements**

**Functional Requirements**

1. **Upload Files:** The client must be able to upload files to the file server. If the file already exists, it should be replaced.
2. **Download Files:** The client must be able to download files from the file server. If the file does not exist, an error message should be returned.
3. **List Directory Contents:** The client must be able to list files and directories on the file server.
4. **Directory Management:** The client must be able to create and remove directories.
5. **Remove Files:** The client must be able to remove files from the file server.
6. **Multiple Client Support:** The file server must support multiple simultaneous client connections.
7. **Shutdown Server:** Clients must be able to cleanly shut down the file server.
8. **Resume Transfers:** The system must support resuming interrupted uploads and downloads.

**Environment Variable**

The client needs to know how to contact the server. This is provided via the environment variable PA1\_SERVER, which includes the server name and port number (e.g., localhost:8000).

**Development Requirements**

* Use Java for implementation.
* Use only Java sockets libraries for networking.
* No third-party software unless explicitly permitted.

**Deliverables**

1. Executable and self-contained jar files.
2. A read-me text file with instructions.
3. Source Code in Java.

**Commands**

**Server Commands**

* java -cp pa1.jar server start <portnumber>

**Client Commands**

1. **Set server:**

* export PA1\_SERVER=<computername:portnumber>

**On Windows**

* set PA1\_SERVER=...

1. **Upload a file:**

* java -cp pa1.jar client upload <path\_on\_client> </path/filename/on/server>

1. **Download a file:**

* java -cp pa1.jar client download </path/existing\_filename/on/server> <path\_on\_client>

1. **List directory contents:**

* java -cp pa1.jar client dir </path/existing\_directory/on/server>

1. **Create a directory:**

* java -cp pa1.jar client mkdir </path/new\_directory/on/server>

1. **Remove an empty directory**

* java -cp pa1.jar client rmdir </path/existing\_directory/on/server>

1. **Remove a file:**

* java -cp pa1.jar client rm </path/existing\_filename/on/server>

1. **Shutdown the server:**

* java -cp pa1.jar client shutdown

**Example Usage**

1. **Set up server and clients:**

* mkdir –p server ; cd server
* java -cp pa1.jar server start 8000
* cd .. ; mkdir –p client1 ; cd client1
* export PA1\_SERVER=localhost:8000
* java -cp pa1.jar client upload text1.pdf /folders/text1.pdf

1. **Start another client and download a file:**

* cd ..; mkdir –p client2 ; cd client2
* java -cp pa1.jar client download /folders/text1.pdf copy\_of\_text1.pdf

1. **Shutdown the server:**

* java -cp pa1.jar client shutdown