**CN Project-1**

**GROUP-**

**Raghava Adarsh Mandarapu- 800937296**

**Jeevitha Mahankali – 800966168**

**Http Client and Http Server**

The following assignment was developed using JAVA language. Client sends the Http Request to the server and the server responds through the Http Response. The communication between client and server can be made possible through the Socket Programming using TCP connection.

First, the object for Socket is to be created. The arguments from the console are assigned to variables for clear understanding and processing. We can fetch the data over web by specifying the port number 80 i.e., of worldwide web widely used by Hypertext Transfer Protocol.

**HTTP Client:**

1. The client program is saved as AdclientApp.java.
2. Test whether all arguments are correctly passed by the user, display the usage of its implementation in case of incorrect arguments.
3. In case of GET command client prepares a request line as prescribed in RFC standard and send that request to the server.
4. And then client waits for response from server. When the response is received at client side it displays that response and terminates.
5. In case of PUT command client first verifies whether the file exists in the specified path or not.
6. If exists, then it requests the server as said in step 2. After that it opens the file, and read its contents in byte format and sends to the server.
7. After sending all contents in the file the client waits for the response. When it receives a response it displays that response and terminates.
8. If the file doesn’t exists the client displays an error report informing the issue and then terminates.
9. In all terminating cases, it closes all the resources bind to the client.

**HTTP Server:**

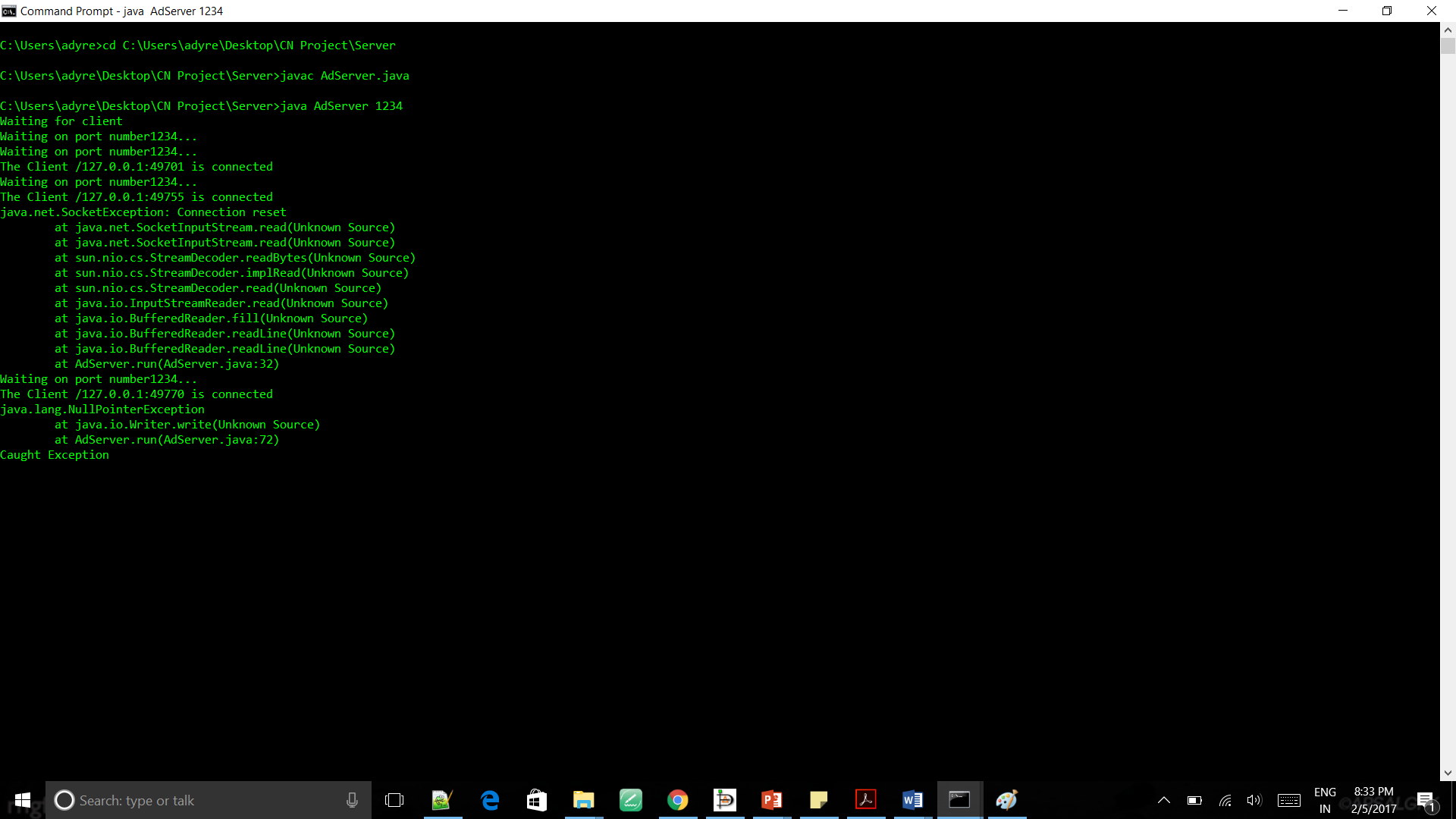
1. The server side program is saved as AdServerApp.java.
2. Test whether all arguments are correctly passed by the user, display the usage of its implementation in case of incorrect arguments.
3. Sever initiates its own socket and then calls a terminate() method, which is used to handle its corresponding resources to properly close in case of termination of the server using ctrl+c.
4. After that it enters into the listening mode and wait for the incoming request. When a request arrives it creates a duplicate socket for handling that request.
5. Using the newly created socket a new thread is started that which handles the entire session of that corresponding request and saves this thread reference to the main thread.
6. If the current request is first one, then server checks for its repository (Server\_Files) exist or not. If the directory is not existed, then it creates.
7. In case of GET method, it checks for the presence of the requested file. If the corresponding file is not present then it replies the corresponding response. Otherwise, it opens that file and write its contents to the client with corresponding status.
8. In case of PUT method, it checks for the requested file in its repository. If not exists it creates the file and copies the data given by the client. Otherwise it overwrites the file with the current receiving data.
9. And replies back to the client with corresponding status/response message.
10. The socket should be closed at the end of the program.

**Result:**

**Server-**

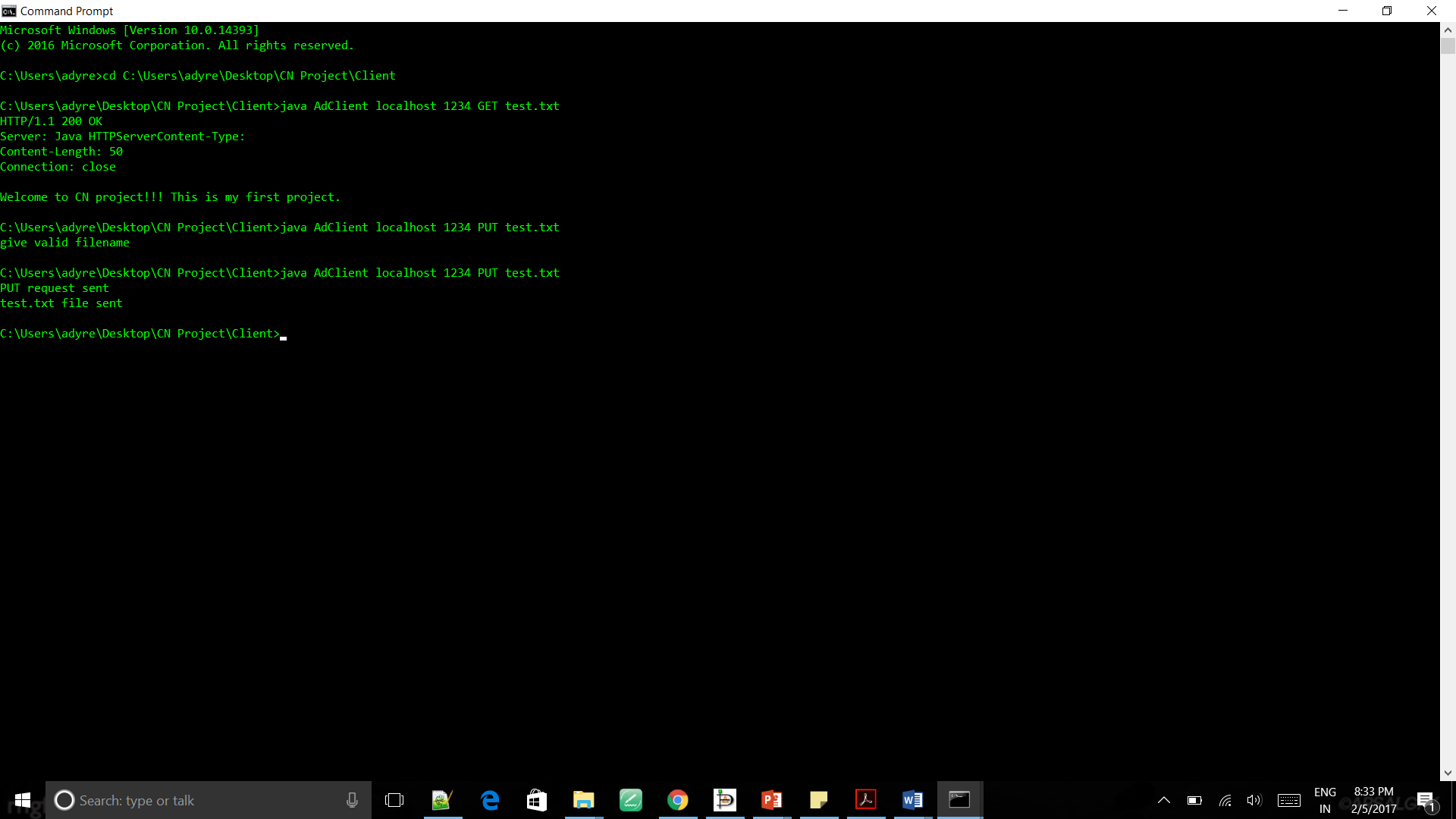
1. Compile the Server program: **javac Adserver.java**

2. Execute the server with some port number: **java Adserver 1234**

****

**Client-**

1. Compile the Client program: **javac Adclient.java**
2. Execute the Client: **java Adclient localhost 1234 get [put] filename**

****