Computer Network

Project #2 : Web Client

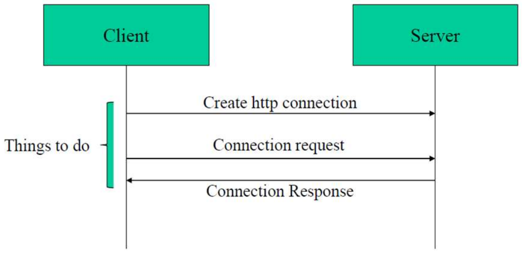
Report

COMPUTER NETWORK PROJECT

Subject	Computer Network
Professor	Jin Seek Choi
Submission Date	November 18th, 2021
University	Hanyang University
School	College of Engineering
Department	Department of Computer Science & Engineering
Student ID	**Name**
2019009261	최가온(CHOI GA ON)

Project Preview

The main goal of this project is to develop a simple Web client for HTTP. This web client must be connected to web server. Also, GET and POST method must be implemented at least.



There are 4 missions in this project.

- MISSION 1

  Set User-Agent in Request header

  : Student Number/Name/Program Name/Subject

- MISSION 2

  Send GET Method to the Server that you are assigned

  Answer how many pictures you received

- MISSION 3

  Post your message to

  http://166.104.143.225:61359/test/postHandleTest

  What message did you receive?

- MISSION 4

  Send GET method to the server that you are assigned and check what you can see

Code Explanation

1. main.java

Step 1.

Import necessary libraries for main.java.

```
package webclient;

import java.io.*;
import java.net.*;
import java.util.*;
```

Step 2.

Create main function, and create a WebClient object. Declare needed variables, such as url, data, picnum.

```
public class main {
    public static void main(String[] args) throws IOException{
        WebClient client = new WebClient();
        String url;
        String data = "2019009261"; // student ID
        Scanner scanner = new Scanner(System.in);
        int picnum;
```

Step 3.

Send GET method to the server, through client.getWebContentByGet method.

```
// MISSION 2
// 2-1: Send Get Method to the Server that you are assigned
String s = client.getWebContentByGet(url + "/test/index.html", "UTF-8",
10000);
System.out.println(s);
```

Step 4.

Get number of images, as a variable named “picnum”. And send POST method to the server, through client.getWebContentByPost method. Then, the server will check whether the number is correct.

```
// MISSION 2
// 2-2: Answer How many Picture you received

// STEP 2
System.out.println("[ STEP 2 ]");
System.out.println("Insert number of valid pictures: ");
picnum = scanner.nextInt();
String ans = client.getWebContentByPost(url + "/test/picResult.html",
data + "/" + Integer.toString(picnum), "UTF-8", 10000);
System.out.println(ans);
```

Step 5.

Send POST method to the server, through client.getWebContentByPost method, and save the message into String s1. Finally, print s1.

```
// MISSION 3
// Post your message to http://166.104.143.225:61359/test/postHandleTest
// What message did you receive?
System.out.println("Answer: ");
String s1 = client.getWebContentByPost(url + "/test/postHandleTest.html",
data, "UTF-8", 10000);
System.out.println(s1);
```

Step 6.

Get url from the user, and save it into String url. Create an ImageGUI object, to implement the GUI for showing an image.

```
        // MISSION 4
        // Send Get Method to the Server that you are assigned and check
what you can see

        // STEP 3
        System.out.println("[ STEP 3 ]");
        System.out.print("URL: ");
        url = scanner.next();
        ImageGUI img_gui = new ImageGUI();
        img_gui.imshow(url);
    }
}
```

2. WebClient.java

Step 1.

Import necessary libraries for WebClient.java.

```
package webclient;

import java.io.*;
import java.net.*;
```

Step 2.

Define WebClient class.

```
public class WebClient {
…
}
```

Step 3-(1).

Define getWebContentByGet method. Its purpose is to implement GET message. First, it checks for the validity of urlString. And make a HTTP connection with the given URL.

```
public String getWebContentByGet(String urlString, final String charset,
int timeout) throws IOException {
    if(urlString == null || urlString.length() == 0) {
        return null;
    }

    urlString = (urlString.startsWith("http://") ||
urlString.startsWith("https://")) ? urlString
             : ("http://" + urlString).intern();
    URL url = new URL(urlString);
    HttpURLConnection conn = (HttpURLConnection) url.openConnection();
    conn.setRequestMethod("GET");
```

Step 3-(2).

Implementing the complete GET message is one of the main point in this method. Set user-agent in request header as a format of “Student

Number/Name/Program Name/Subject”. Also add exception handling for IOException.

```
    // MISSION 1
    // Set User-Agent in Request header : Student Number/Name/Program
    Name/Subject
    conn.setRequestProperty("User-Agent",
    "2019009261/GAONCHOI/WebClient/computer_network");
    conn.setRequestProperty("Accept", "text/html");
    conn.setConnectTimeout(timeout);
    try {
       if(conn.getResponseCode() != HttpURLConnection.HTTP_OK) {
          return null;
       }
    } catch(IOException e) {
       e.printStackTrace();
       return null;
    }
```

Step 3-(3).

Create InputStream and BufferedReader object for completing the GET message.

```
    InputStream input = conn.getInputStream();
       BufferedReader reader = new BufferedReader(new
    InputStreamReader(input, charset));
       String line = null;
       StringBuffer sb = new StringBuffer();

       while((line = reader.readLine()) != null) {
          sb.append(line).append("\r\n");
       }
       if(reader != null) {
          reader.close();
       }
       if(conn != null) {
          conn.disconnect();
       }
       return sb.toString();
    }
```

Step 4-(1).

This process is similar to the POST message side. Make a HTTP connection and set various component as it needs to be.

```
public String getWebContentByPost(String urlString, String data, final
String charset, int timeout) throws IOException {
    if (urlString == null || urlString.length() == 0) {
        return null;
    }

    urlString = (urlString.startsWith("http://") ||
urlString.startsWith("https://")) ? urlString
            : ("http://" + urlString).intern();
    URL url = new URL(urlString);
    HttpURLConnection connection = (HttpURLConnection)
url.openConnection();

    connection.setDoOutput(true);
    connection.setDoInput(true);
    connection.setRequestMethod("POST");

    connection.setUseCaches(false);
    connection.setInstanceFollowRedirects(true);

    connection.setRequestProperty("Content-Type", "text/xml;charset=UTF-
8");
```

Step 4-(2).

Set user-agent in request header as a format of "Student Number/Name/Program Name/Subject". Also add exception handling for IOException.

```
    // MISSION 1
    // Set User-Agent in Request header : Student Number/Name/Program
    Name/Subject
    connection.setRequestProperty("User-Agent",
    "2019009261/GAONCHOI/WebClient/computer_network");

    connection.setRequestProperty("Accept", "text/xml");
    connection.setConnectTimeout(timeout);
    connection.connect();
    DataOutputStream out = new
    DataOutputStream(connection.getOutputStream());

    byte[] content = data.getBytes("UTF-8");

    out.write(content);
    out.flush();
    out.close();
```

Step 4-(3).

Create InputStream and BufferedReader object for completing the POST message.

```
try {
       if(connection.getResponseCode() != HttpURLConnection.HTTP_OK) {
          return null;
       }
   } catch(IOException e) {
       e.printStackTrace();
       return null;
   }

   InputStream input = connection.getInputStream();
   BufferedReader reader = new BufferedReader(new
InputStreamReader(input, charset));
   String line = null;
   StringBuffer sb = new StringBuffer();

   while((line = reader.readLine()) != null) {
       sb.append(line).append("\r\n");
   }
   if(reader != null) {
       reader.close();
   }
   if(connection != null) {
       connection.disconnect();
   }
   return sb.toString();
}
```

3. ImageGUI.java

Step 1.

Import necessary libraries for ImageGUI.java.

```
import javax.imageio.ImageIO;
import javax.swing.*;
import java.awt.Image;
import java.awt.image.BufferedImage;
import java.io.IOException;
import java.net.URL;
```

Step 2.

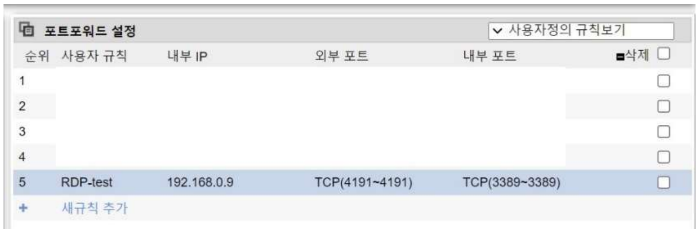
Define ImageGUI class. With ImageIO, read an image with an URL, and save it as “img”. Also, JFrame and JLabel need to be created. Add label_ as a component of JFrame frame. Configure the GUI component sizes.

```
public class ImageGUI {
    public void imshow(String url) {
        BufferedImage img = null;

        try {
            URL url_ = new URL(url);
            img = ImageIO.read(url_);
        }
        catch (IOException e){
            e.printStackTrace();
        }

        JFrame frame_ = new JFrame();
        JLabel label_ = new JLabel(new ImageIcon(img));

        frame_.add(label_);
        frame_.setSize(500, 500);
        frame_.setBounds(100, 100, 700, 500);
        frame_.setVisible(true);
    }
}
```

Instructions / Execution / Results

STEP 1

Configure port forward settings to start this program.



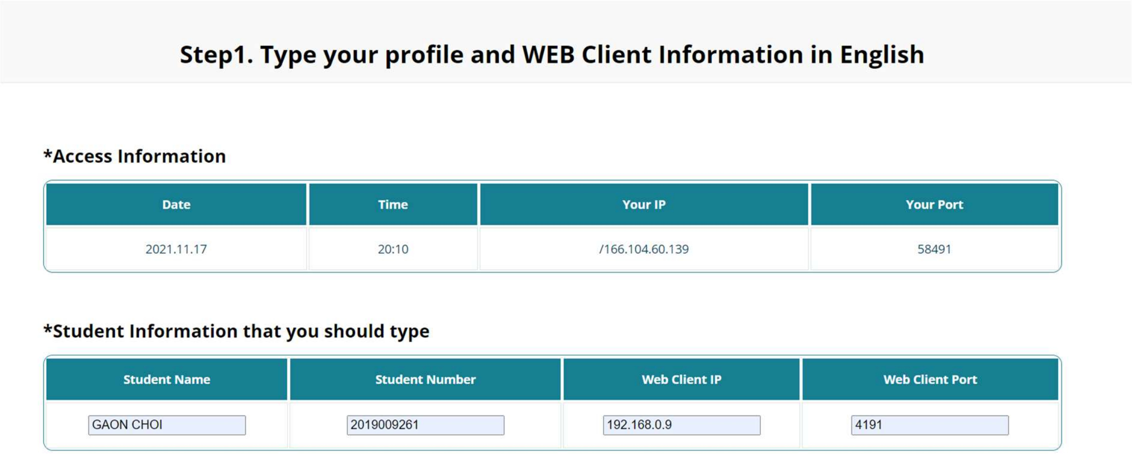
포트포워드 설정

순위	사용자 규칙	내부 IP	외부 포트	내부 포트	삭제 □
1					□
2					□
3					□
4					□
5	RDP-test	192.168.0.9	TCP(4191~4191)	TCP(3389~3389)	□

The inner PORT is 4191, and the outer PORT is 3389.

Open a commercial Internet client software(Chrome, Safari, Internet Explorer …).
Go to : http://166.104.143.225:9090/main.html

Use the auto-marking system, go to web client marking. Insert each information: Student Name, Student Number, Web Client IP, Web Client Port.



Step1. Type your profile and WEB Client Information in English

*Access Information

Date	Time	Your IP	Your Port
2021.11.17	20:10	/166.104.60.139	58491

*Student Information that you should type

Student Name	Student Number	Web Client IP	Web Client Port
GAON CHOI	2019009261	192.168.0.9	4191

- Student Name: GAON CHOI

- Student Number: 2019009261

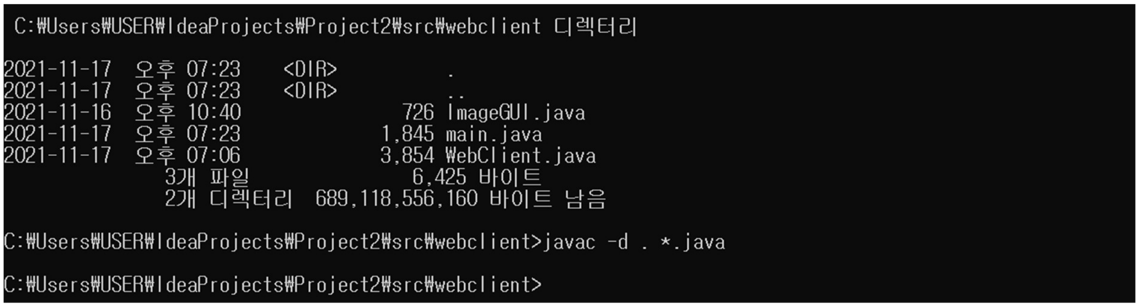
- Web Client IP: 192.168.0.9

- Web Client Port: 4191

STEP 2

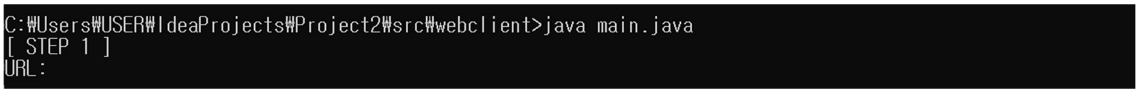
At the command line, go to “Project2₩src₩webclient” to prepare for execution of web client.

And compile the java file to make .class files.



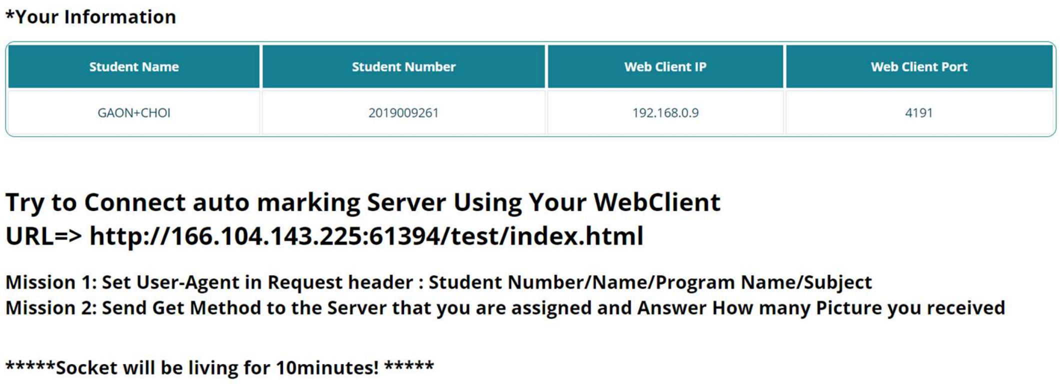
```
 C:₩Users₩USER₩IdeaProjects₩Project2₩src₩webclient 디렉터리

2021-11-17  오후 07:23    <DIR>          .
2021-11-17  오후 07:23    <DIR>          ..
2021-11-16  오후 10:40               726 ImageGUI.java
2021-11-17  오후 07:23             1,845 main.java
2021-11-17  오후 07:06             3,854 WebClient.java
               3개 파일               6,425 바이트
               2개 디렉터리  689,118,556,160 바이트 남음

C:₩Users₩USER₩IdeaProjects₩Project2₩src₩webclient>javac -d . *.java

C:₩Users₩USER₩IdeaProjects₩Project2₩src₩webclient>
```

Execute main.java file to operate the web server.



```
C:₩Users₩USER₩IdeaProjects₩Project2₩src₩webclient>java main.java
[ STEP 1 ]
URL:
```

After clicking “submit” button at the auto-marking system, you will see two missions, as following.



***Your Information**

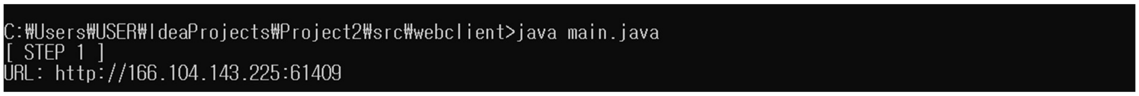
Student Name	Student Number	Web Client IP	Web Client Port
GAON+CHOI	2019009261	192.168.0.9	4191

Try to Connect auto marking Server Using Your WebClient
URL=> http://166.104.143.225:61394/test/index.html

Mission 1: Set User-Agent in Request header : Student Number/Name/Program Name/Subject
Mission 2: Send Get Method to the Server that you are assigned and Answer How many Picture you received

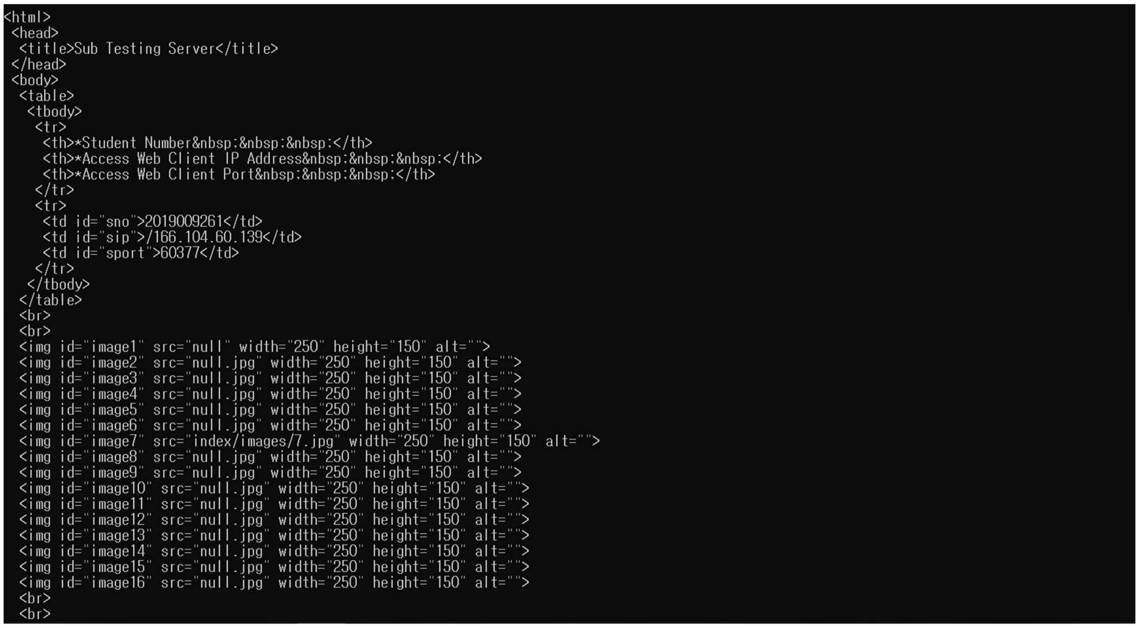
*******Socket will be living for 10minutes! *******

Copy the URL into the web server, executing on the command line. Paste only the IP address and port number.

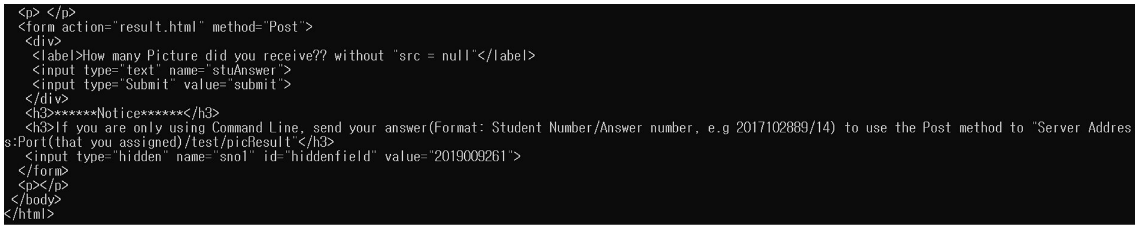


```
C:₩Users₩USER₩IdeaProjects₩Project2₩src₩webclient>java main.java
[ STEP 1 ]
URL: http://166.104.143.225:61409
```

Then,

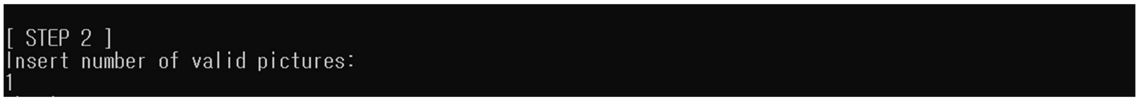


```
<html>
 <head>
  <title>Sub Testing Server</title>
 </head>
 <body>
  <table>
   <tbody>
    <tr>
     <th>*Student Number&nbsp;&nbsp;&nbsp;</th>
     <th>*Access Web Client IP Address&nbsp;&nbsp;&nbsp;</th>
     <th>*Access Web Client Port&nbsp;&nbsp;&nbsp;</th>
    </tr>
    <tr>
     <td id="sno">2019009261</td>
     <td id="sip">/166.104.60.139</td>
     <td id="sport">60377</td>
    </tr>
   </tbody>
  </table>
  <br>
  <br>
  <img id="image1" src="null" width="250" height="150" alt="">
  <img id="image2" src="null.jpg" width="250" height="150" alt="">
  <img id="image3" src="null.jpg" width="250" height="150" alt="">
  <img id="image4" src="null.jpg" width="250" height="150" alt="">
  <img id="image5" src="null.jpg" width="250" height="150" alt="">
  <img id="image6" src="null.jpg" width="250" height="150" alt="">
  <img id="image7" src="index/images/7.jpg" width="250" height="150" alt="">
  <img id="image8" src="null.jpg" width="250" height="150" alt="">
  <img id="image9" src="null.jpg" width="250" height="150" alt="">
  <img id="image10" src="null.jpg" width="250" height="150" alt="">
  <img id="image11" src="null.jpg" width="250" height="150" alt="">
  <img id="image12" src="null.jpg" width="250" height="150" alt="">
  <img id="image13" src="null.jpg" width="250" height="150" alt="">
  <img id="image14" src="null.jpg" width="250" height="150" alt="">
  <img id="image15" src="null.jpg" width="250" height="150" alt="">
  <img id="image16" src="null.jpg" width="250" height="150" alt="">
  <br>
  <br>
```



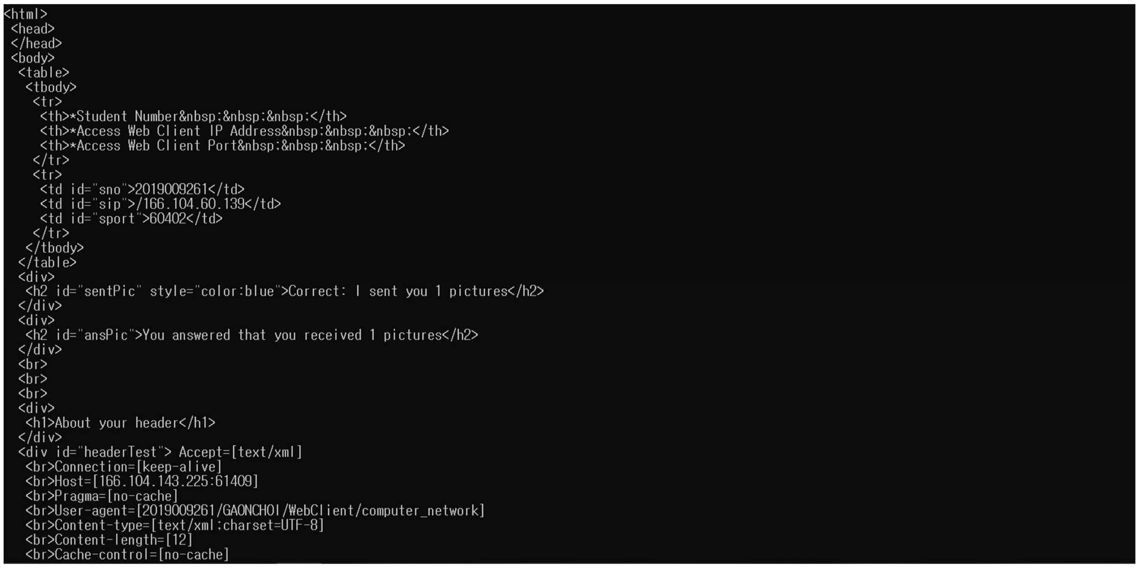
```
  <p> </p>
  <form action="result.html" method="Post">
   <div>
    <label>How many Picture did you receive?? without "src = null"</label>
    <input type="text" name="stuAnswer">
    <input type="Submit" value="submit">
   </div>
   <h3>******Notice******</h3>
   <h3>If you are only using Command Line, send your answer(Format: Student Number/Answer number, e.g 2017102889/14) to use the Post method to "Server Addres
s:Port(that you assigned)/test/picResult"</h3>
   <input type="hidden" name="sno1" id="hiddenfield" value="2019009261">
  </form>
  <p></p>
 </body>
</html>
```

In this case, there are only one valid images (image/7.jpg). Then, insert 1 as the number of images, on the command line.

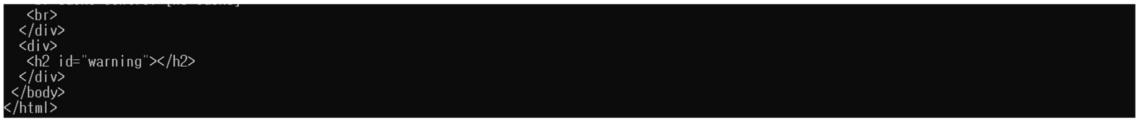


```
[ STEP 2 ]
Insert number of valid pictures:
1
```

As this number is sent to the server, the auto-marking program checks that the number is correct, compared to actual number of valid sent images. After checking it, the client gets

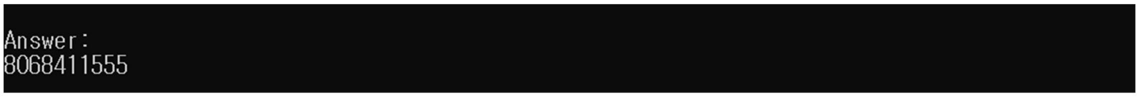


```
<html>
 <head>
 </head>
 <body>
  <table>
   <tbody>
    <tr>
     <th>*Student Number&nbsp;&nbsp;&nbsp;</th>
     <th>*Access Web Client IP Address&nbsp;&nbsp;&nbsp;</th>
     <th>*Access Web Client Port&nbsp;&nbsp;&nbsp;</th>
    </tr>
    <tr>
     <td id="sno">2019009261</td>
     <td id="sip">/166.104.60.139</td>
     <td id="sport">60402</td>
    </tr>
   </tbody>
  </table>
  <div>
   <h2 id="sentPic" style="color:blue">Correct: I sent you 1 pictures</h2>
  </div>
  <div>
   <h2 id="ansPic">You answered that you received 1 pictures</h2>
  </div>
  <br>
  <br>
  <br>
  <div>
   <h1>About your header</h1>
  </div>
  <div id="headerTest"> Accept=[text/xml]
   <br>Connection=[keep-alive]
   <br>Host=[166.104.143.225:61409]
   <br>Pragma=[no-cache]
   <br>User-agent=[2019009261/GAONCHOI/WebClient/computer_network]
   <br>Content-type=[text/xml;charset=UTF-8]
   <br>Content-length=[12]
   <br>Cache-control=[no-cache]
```

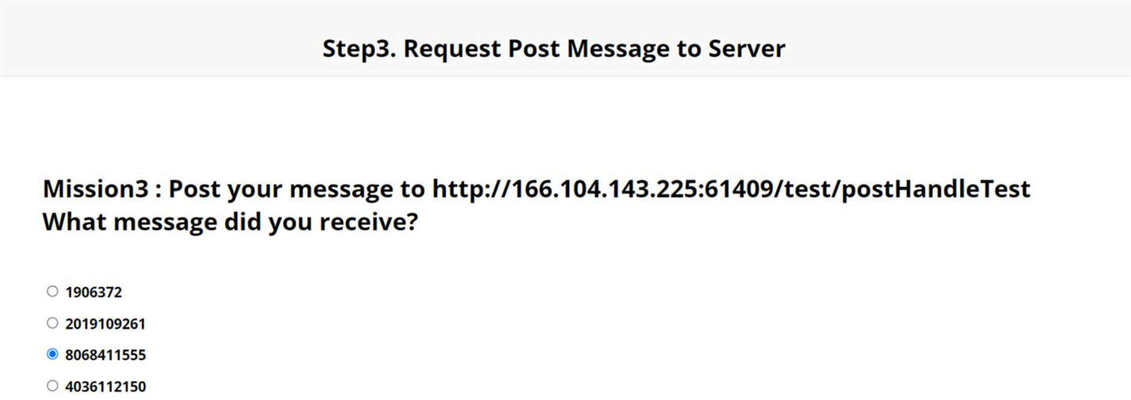


```
   <br>
  </div>
  <div>
   <h2 id="warning"></h2>
  </div>
 </body>
</html>
```

At the bottom side, there is a sequence of numbers. Check the number in the auto-marking server, that matches exactly.



```
Answer:
8068411555
```

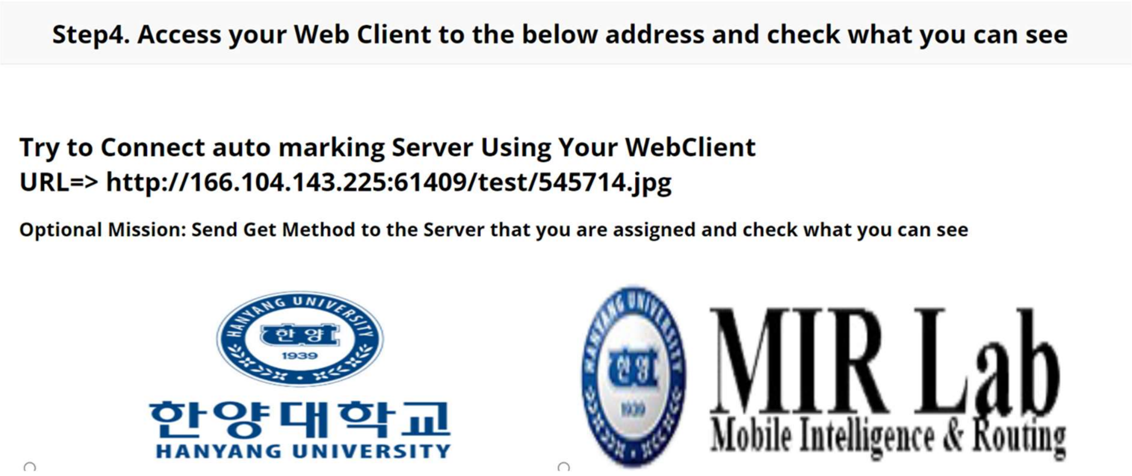


Step3. Request Post Message to Server

Mission3 : Post your message to http://166.104.143.225:61409/test/postHandleTest
What message did you receive?

○ 1906372
○ 2019109261
● 8068411555
○ 4036112150

Then, you will see the following result at the auto-marking system side.



Step4. Access your Web Client to the below address and check what you can see

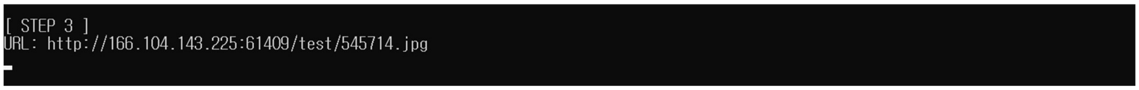
Try to Connect auto marking Server Using Your WebClient
URL=> http://166.104.143.225:61409/test/545714.jpg

Optional Mission: Send Get Method to the Server that you are assigned and check what you can see

○ ○

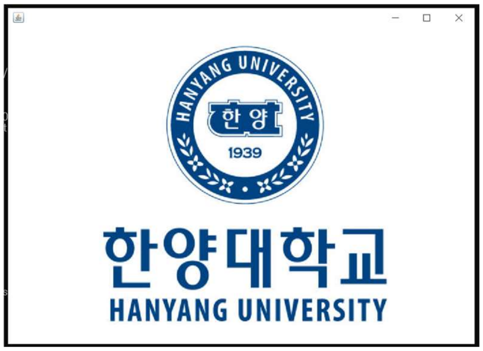
Copy the URL, and paste it onto the command line.

※ URL: http://166.104.143.225:61409/test/545714.jpg

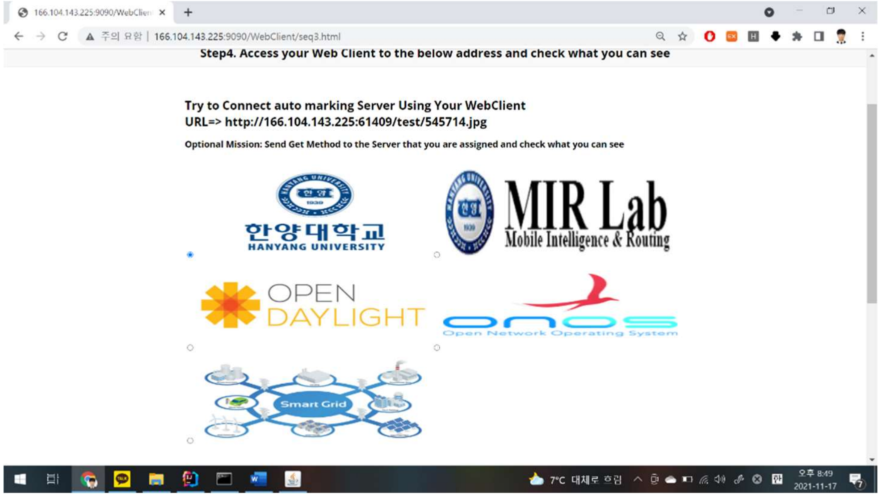


```
[ STEP 3 ]
URL: http://166.104.143.225:61409/test/545714.jpg
_
```

Then, you will get a GUI window that shows an Image.



Check the radio button that matches exactly with the upper image, at the auto-marking side.

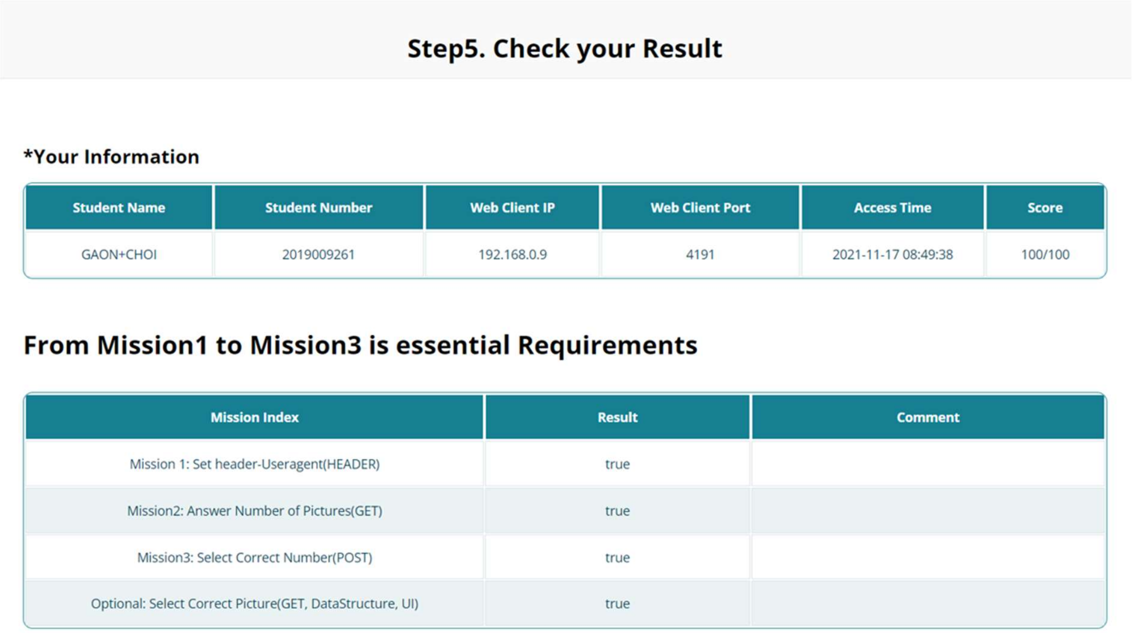


Step4. Access your Web Client to the below address and check what you can see

Try to Connect auto marking Server Using Your WebClient
URL=> http://166.104.143.225:61409/test/545714.jpg

Optional Mission: Send Get Method to the Server that you are assigned and check what you can see

The final result is as follows:



Step5. Check your Result

*Your Information

Student Name	Student Number	Web Client IP	Web Client Port	Access Time	Score
GAON+CHOI	2019009261	192.168.0.9	4191	2021-11-17 08:49:38	100/100

From Mission1 to Mission3 is essential Requirements

Mission Index	Result	Comment
Mission 1: Set header-Useragent(HEADER)	true	
Mission2: Answer Number of Pictures(GET)	true	
Mission3: Select Correct Number(POST)	true	
Optional: Select Correct Picture(GET, DataStructure, UI)	true	

Opinions

With project1 & project2, I fully understood the concept of the server and the client, and their relationship. We usually use Internet applications and become clients to get a lot of information. I think the experience of making these processes one by one has been a valuable experience. In particular, GET methods and POST methods were correctly implemented on my own, and when they were successfully connected to the server, it allowed me to feel a sense of accomplishment on their own.

In addition, I experienced implementing GUI(Graphic User Interface) for the first time in Java, which was one of the biggest challenges for me who was somewhat afraid of trying new things. However, as with everything, I could feel another sense of accomplishment when I saw the picture being successfully printed on the new windows!

I would like to thank the professor and assistant for preparing practical assignments. The most convenient and helpful part of performing the assignments was the introduction of an automatic scoring system. However, in reality, I think it is difficult to exist an automatic scoring program in the development stage. When developing in the actual industry field, we will actively identify the cause of problems and make efforts during the undergraduate period to develop the ability to actively solve numerous problems.