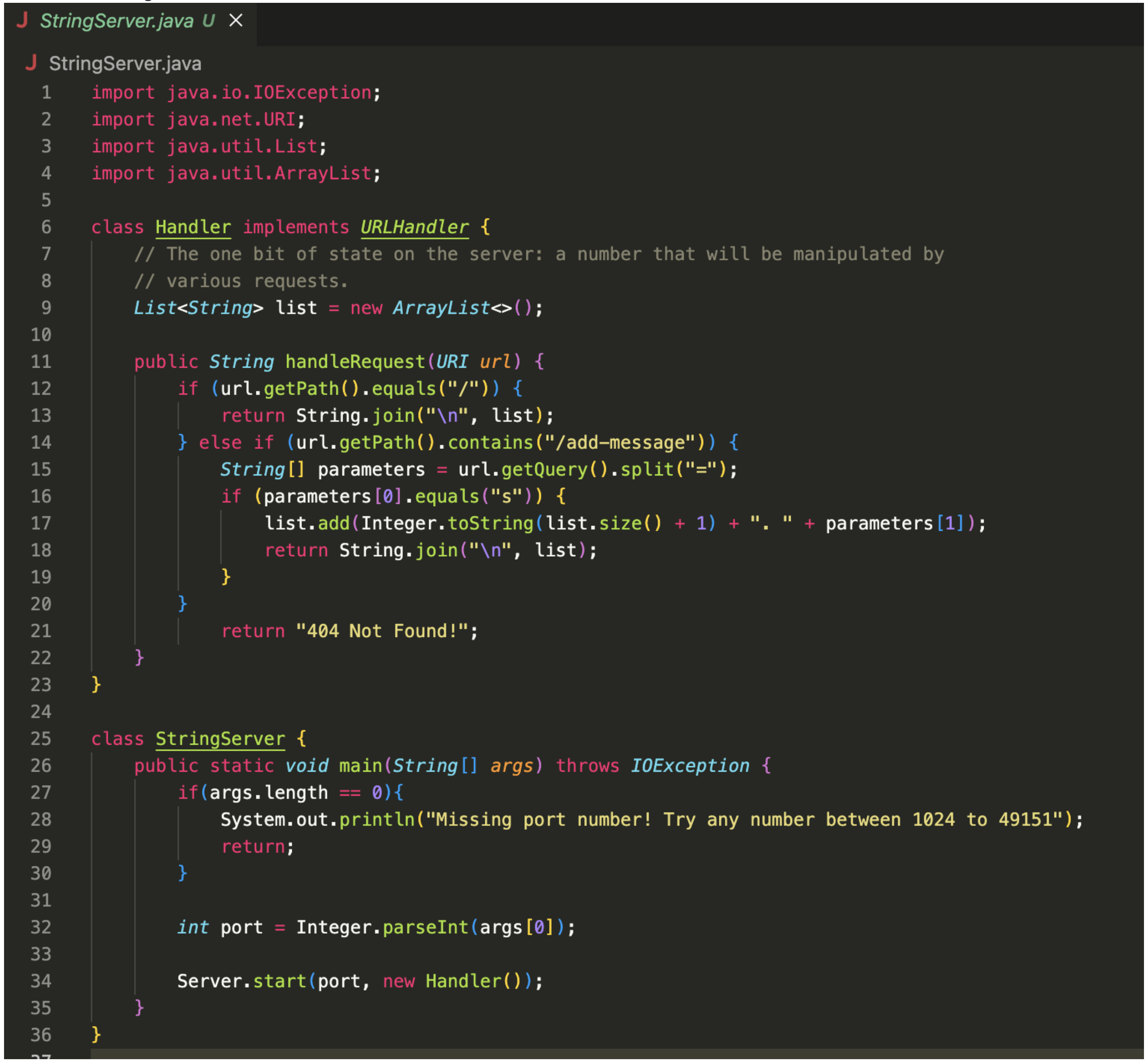
cse15l-lab-report2

Part 1

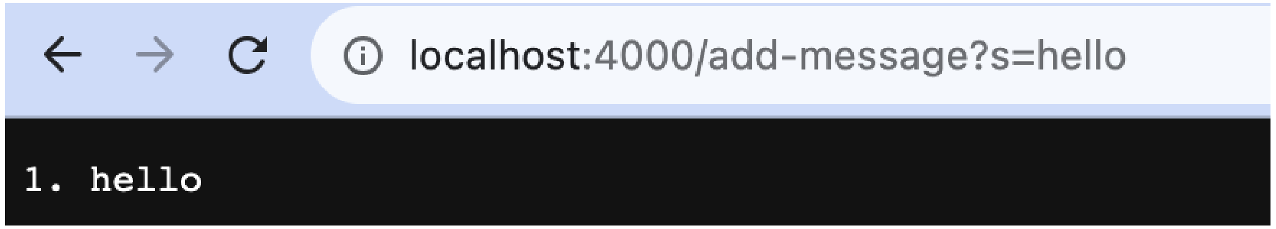
The code of StringServer is as follows:



StringServer.java U

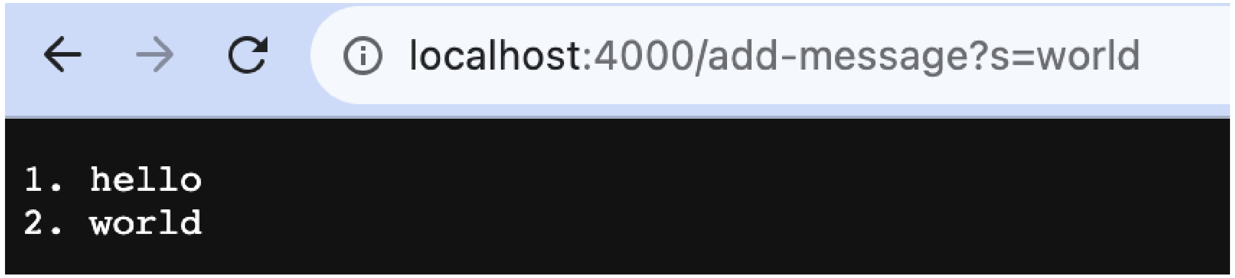
StringServer.java

```java
import java.io.IOException;
import java.net.URI;
import java.util.List;
import java.util.ArrayList;

class Handler implements URLHandler {
    // The one bit of state on the server: a number that will be manipulated by
    // various requests.
    List<String> list = new ArrayList<>();

    public String handleRequest(URI url) {
        if (url.getPath().equals("/")) {
            return String.join("\n", list);
        } else if (url.getPath().contains("/add-message")) {
            String[] parameters = url.getQuery().split("=");
            if (parameters[0].equals("s")) {
                list.add(Integer.toString(list.size() + 1) + ". " + parameters[1]);
                return String.join("\n", list);
            }
        }
            return "404 Not Found!";
    }
}

class StringServer {
    public static void main(String[] args) throws IOException {
        if(args.length == 0){
            System.out.println("Missing port number! Try any number between 1024 to 49151");
            return;
        }

        int port = Integer.parseInt(args[0]);

        Server.start(port, new Handler());
    }
}
```



```
1. hello
```

1. When we start our server, a `Handler` object is created and `list` is initialized, which is a ArrayList to storage string.
2. When we do a request `http://localhost:4000/add-message?s=hello`, the `handleRequest(URI url)` is called by the whole url as an argument. the value of `url.getPath()` equals to `/add-message?s=hello`, so `url.getPath().contains("/add-message")` is true.
3. The value of `url.getQuery()` is equal to `"s=hello"`. `split("=")` will splits the String into multiple Strings by separator `"="` and return an array `["s", "hello"]` to `parameters`.
4. Since `parameters[0].equals["s"]` is true, `list.add()` is called.
5. `list.add(Integer.toString(list.size() + 1) + ". " + parameters[1]);`
   - `(Integer.toString(list.size() + 1) + ". " + parameters[1])` Every time we get the size of the list plus 1 and convert it to String from Integet. Add `.` and `parameters[1]` after that. The result of the value equals to `"1. hello"`.
   - `list.add("1. hello)` means add the String `"1. hello"` to list and `list.size()` equals to 1 now.
6. `return String.join("\n", list)`
   - `join("\n", list)` method concatenates the given `list` with `\n` and returns the concatenated string.
   - Since there is only one String in `list`, it returns `"1. hello"`.

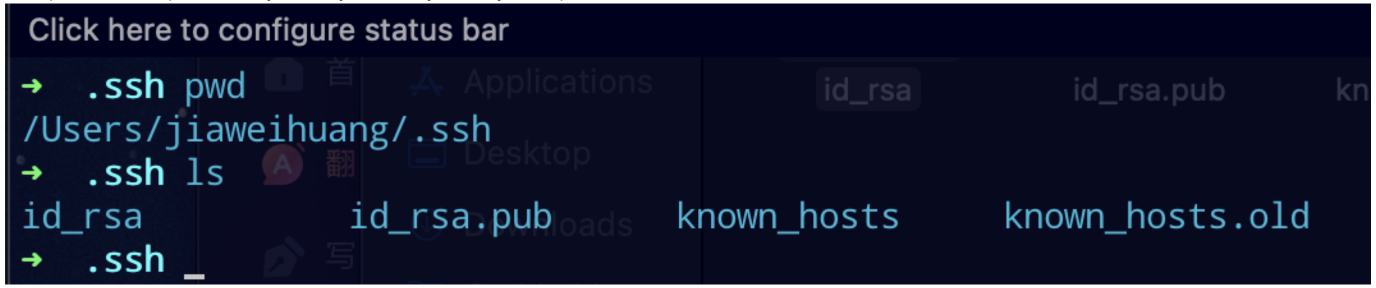


```
1. hello
2. world
```

1. When we do a request `http://localhost:4000/add-message?s=world`, because it has the same path with last request, it is the same as the preceding runing steps up until the 5 step.
2. `list.add(Integer.toString(list.size() + 1) + ". " + parameters[1]);`
   - Since `list` already have one String and the size is 1, `list.size() + 1` equals to 2 and `paramters[1]` equals to `"world"`.
   - `"2. world"` is added to the `list` and `list.size()` equals to 2 now.
3. `return String.join("\n", list)`
   - `list` contains `("1. hello", "2. world")` and we concatenates them with `\n`, so we return the String `"1. hello\n2.world"`.

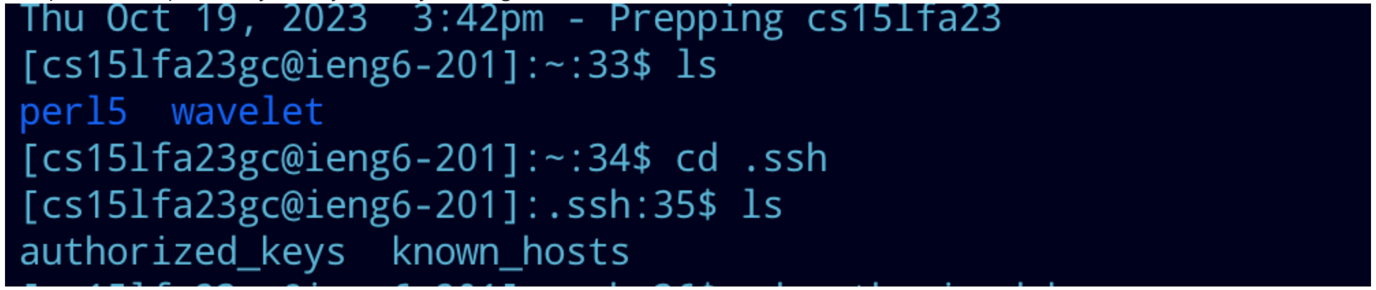
Part 2

1. The path to the private key for my SSH key on my computer.



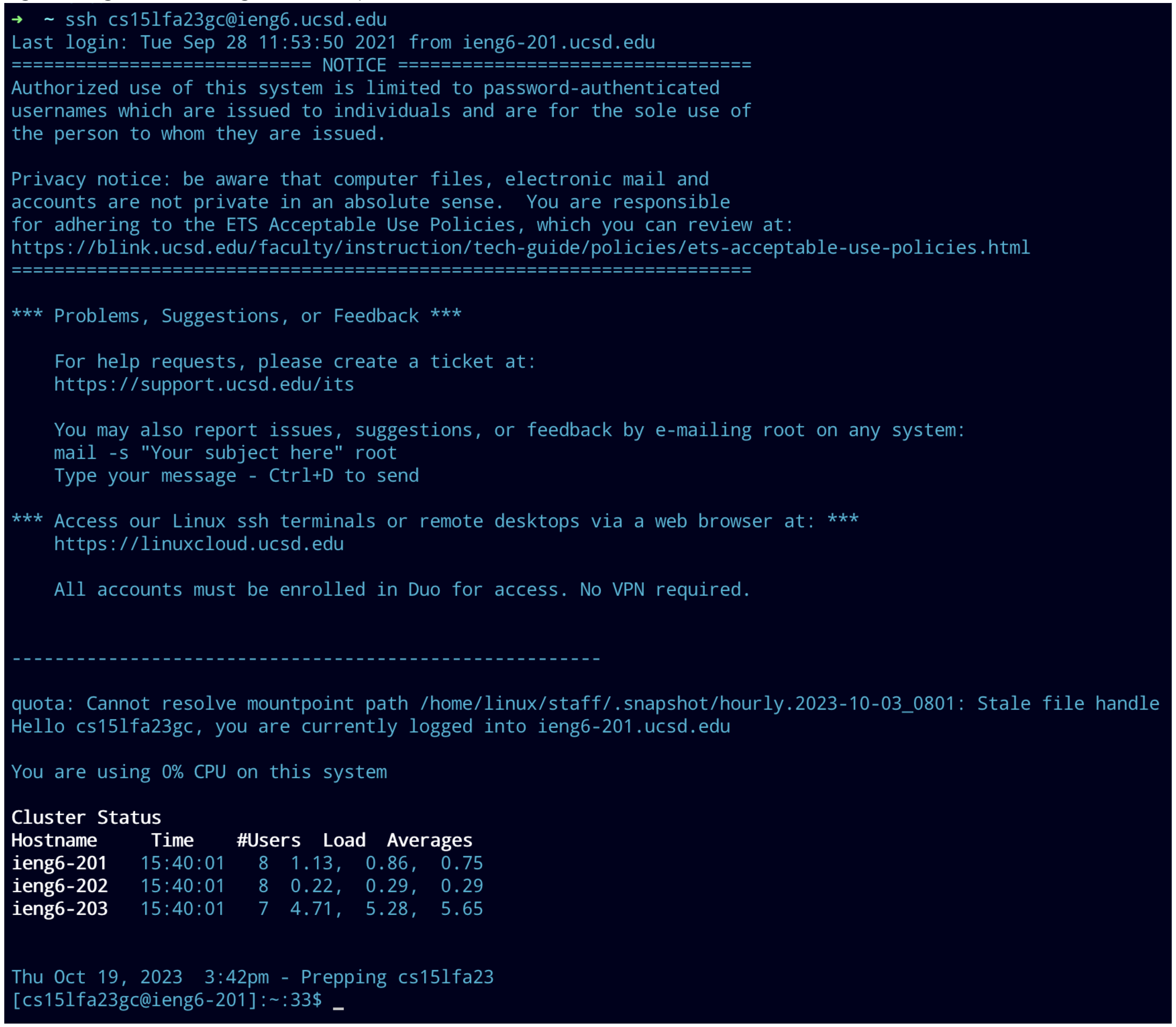
```
Click here to configure status bar
➜  .ssh pwd
/Users/jiaweihuang/.ssh
➜  .ssh ls
id_rsa          id_rsa.pub      known_hosts     known_hosts.old
➜  .ssh _
```

2. The path to the public key for my SSH key on ieng6.



```
Thu Oct 19, 2023  3:42pm - Prepping cs15lfa23
[cs15lfa23gc@ieng6-201]:~:33$ ls
perl5  wavelet
[cs15lfa23gc@ieng6-201]:~:34$ cd .ssh
[cs15lfa23gc@ieng6-201]:.ssh:35$ ls
authorized_keys  known_hosts
```

3. log into ieng6 without being asked for a password.



```
➜  ~ ssh cs15lfa23gc@ieng6.ucsd.edu
Last login: Tue Sep 28 11:53:50 2021 from ieng6-201.ucsd.edu
============================ NOTICE =================================
Authorized use of this system is limited to password-authenticated
usernames which are issued to individuals and are for the sole use of
the person to whom they are issued.

Privacy notice: be aware that computer files, electronic mail and
accounts are not private in an absolute sense.  You are responsible
for adhering to the ETS Acceptable Use Policies, which you can review at:
https://blink.ucsd.edu/faculty/instruction/tech-guide/policies/ets-acceptable-use-policies.html
=====================================================================

*** Problems, Suggestions, or Feedback ***

    For help requests, please create a ticket at:
    https://support.ucsd.edu/its

    You may also report issues, suggestions, or feedback by e-mailing root on any system:
    mail -s "Your subject here" root
    Type your message - Ctrl+D to send

*** Access our Linux ssh terminals or remote desktops via a web browser at: ***
    https://linuxcloud.ucsd.edu

    All accounts must be enrolled in Duo for access. No VPN required.


-------------------------------------------------------

quota: Cannot resolve mountpoint path /home/linux/staff/.snapshot/hourly.2023-10-03_0801: Stale file handle
Hello cs15lfa23gc, you are currently logged into ieng6-201.ucsd.edu

You are using 0% CPU on this system

Cluster Status
Hostname     Time    #Users  Load  Averages
ieng6-201   15:40:01    8   1.13,   0.86,   0.75
ieng6-202   15:40:01    8   0.22,   0.29,   0.29
ieng6-203   15:40:01    7   4.71,   5.28,   5.65


Thu Oct 19, 2023  3:42pm - Prepping cs15lfa23
[cs15lfa23gc@ieng6-201]:~:33$ _
```

Part 3

1. Understand how to connect the remote server with ssh, and log in without password through SSH key.
2. learn how to use VS Code.
3. learn related methods of processing URL in Java.