Step 1. Deploy on Windows operating system:

Clone the project into **your laptop**, open git bash and write the instruction:
"git clone https://github.com/cjhwa96/DS-Chat-201904.git".
Open Eclipse jee/ IntelliJ (java web IDE), import the project folder into the IDE. Then, wait for a couple of minutes, until the maven dependency shows up at the left side.
After this process, go to "src/main/java" and please right click "WebsocketDemoApplication.java" and choose "run as → "java application", wait about 10 seconds and open your browser, type in "localhost:8080" in the address bar, the web application will show in your browser's page.

Step 2. Package the *.jar (runable java package) on your windows laptop

Note: Make sure the step 1 has been finished.

Open the terminal, go to the directory of this project, type in the command "mvn clean package" (if this process fails, please check your JAVA_HOME, PATH, MAVEN_HOME those kinds of various system environment variables are set up successfully). Wait for a couple of seconds and the runnable *.jar package will appear in the "target" folder.

Step 3. Deploy on Linux operating system:

Note: Make sure the step 2 has been finished.

Transmit the runnable *.jar to your linux server (Using SSH technique and FTP technique), make sure the port (port 21 & port 22, etc.) on firewall has been shut down. Type the following command in your linux terminal: *java -jar xxxx.jar*, "xxxx" should be specified as the name of your *.jar file (which is the runnable jar package of our web application). If the step occurs failure, please make sure the following conditions have been fulfilled:

- JAVA_HOME has been set up clearly in your linux server, type in "*echo $JAVA_HOME*" and "*java -version*" to make sure your java environment has been deployed successfully on your linux server.
- The port of our web application (default port is 8080) has been opened on the firewall settings.
- The linux server in the public network (which means that it has a public IP rather than private IP), this is for ensuring that client server can access the server side directly

Step 4. Deploy the load balancing network structure (On Windows OS)

Note: If you want to deploy reload balancing on Windows OS, make sure the step 1 has been finished.

Download Nginx server on your Windows laptop.

Open the directory of the nginx and open the "conf" folder, and modify the conf file into the following code (several areas are selected):

```
worker_processes  1;

events {
    worker_connections  1024;
}


http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;

    #Server Clustering
    upstream tomcat {
        server 127.0.0.1:9999 weight=10;
        server 127.0.0.1:8888 weight=10;#Weight value
    }

    server {
        listen       80;
        server_name  localhost;

        location / {
            #root   html;
            #index  index.html index.htm;
            proxy_pass http://tomcat;
            proxy_redirect default;
        }

        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }
```

Figure 1. Conf file of nginx

Area 3 and area 4 the name (tomcat, in this figure) can be modified but must be the same name. Area 1 is to configure the ip address of servers, which run the web application. Area 2 is the weight of the servers, the higher weight value, the higher possibility to redirect to that server. The most important part is the upstream part, it registers the server (redirect destination).

```
spring.mvc.view.prefix: /WEB-INF/jsp/
spring.mvc.view.suffix: .jsp
server.port=8888
```

Figure 2. The configuration in "application.properties"

For testing in the localhost, you should remember to configure the server port in our web application. In the “src/main/resource” you should add the line here to configure which desired port to run this program.

Open cmd (Win+R+input cmd), then go to the directory of the nginx, and type the code “nginx -s reload”. Nginx will be restarted and listen to the port of 80 (Thus also make sure the 80 port of your firewall also be opened).

Now access to the address *localhost:80*, the nginx server (plays as the proxy server), it will begin to redirect to the destination server successfully. If in this stage, the white page error occurs (JSP NOT FOUND ERROR occurs), make sure you runnable *.jar file has package all the files in the project including admin.jsp, chatroom.jsp and login.jsp.

Step 5. Deploy the load balancing network structure (On Ubuntu OS)
Note: If you want to deploy reload balancing on Ubuntu OS, make sure the step 3 has been finished.
Install the nginx server on your linux server, and access to the installation of directory of Nginx. Now type the following instruction in the terminal: *“cd conf/”, “sudo vim conf”*. modify the code in conf file (this configuration part should be refer to Step 4). After modification, reload your nginx server, and now your successfully deploy the load-balancing network structure.