IT 351 Design Document for Individual Projects

Zach Sharpe

2/4/18

Table of Contents

Discussion Board 2 3

Individual Project 2 3

Individual Project 3 6

Individual Project 4 7

Individual Project 5 9

## Discussion Board 2

* **Explain how you would modify your server to make it multithreaded, allowing more than one client to connect and request product and customer data.**

The server application will be made to start a thread for each connected client.

My current server class will convert into one that implements threading:

Implements runnable interface, utilizes thread class, methods in conjunction, run method.

The established Run() method will take a socket as the argument to allow spinning new threads each time a new client requests.

Lastly, a new socket variable will be established that allows a new socket per needed client connection.

* **Be sure you specifically identify which Java API classes you intend to use.**

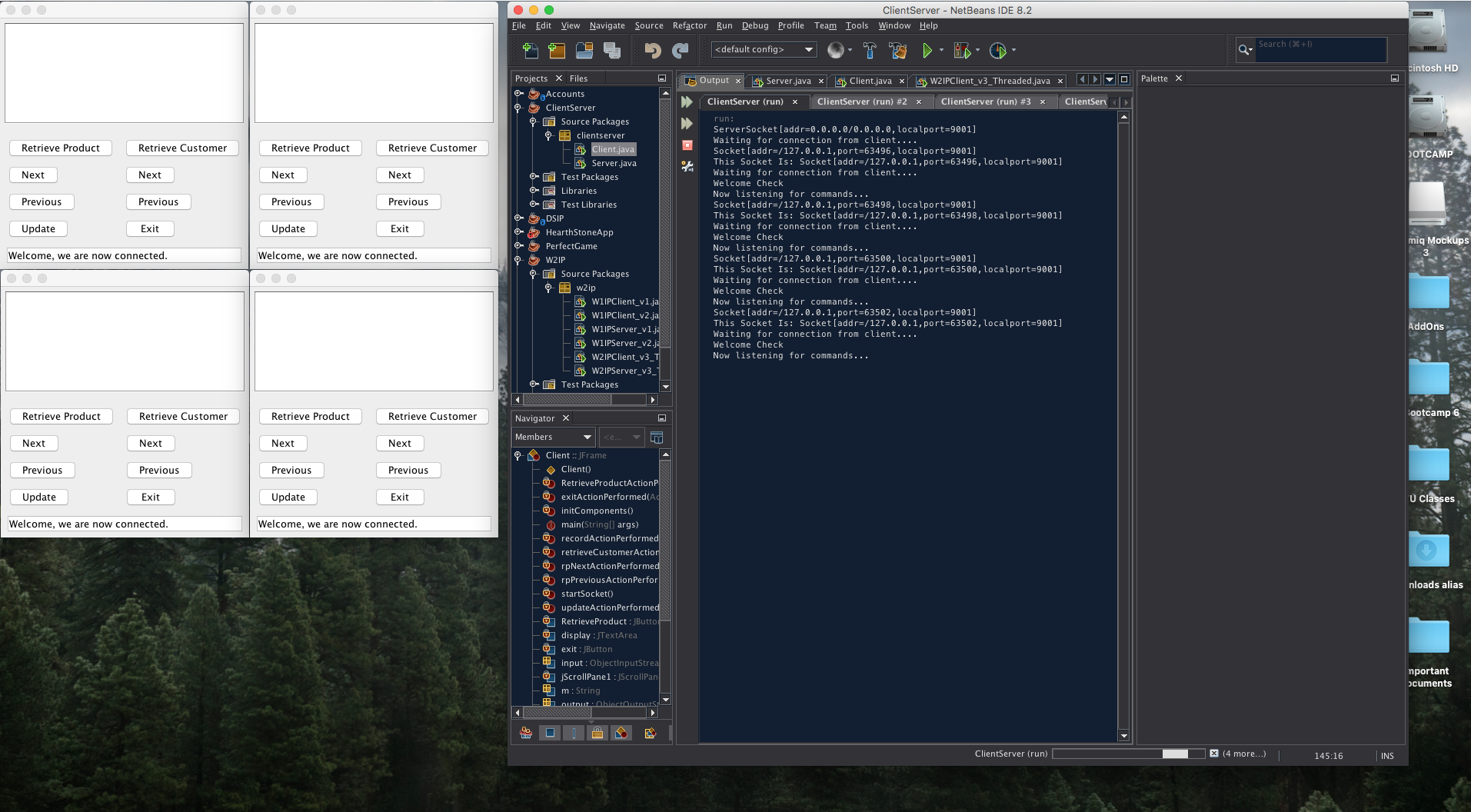
I will use the Thread class for establishing new threads.

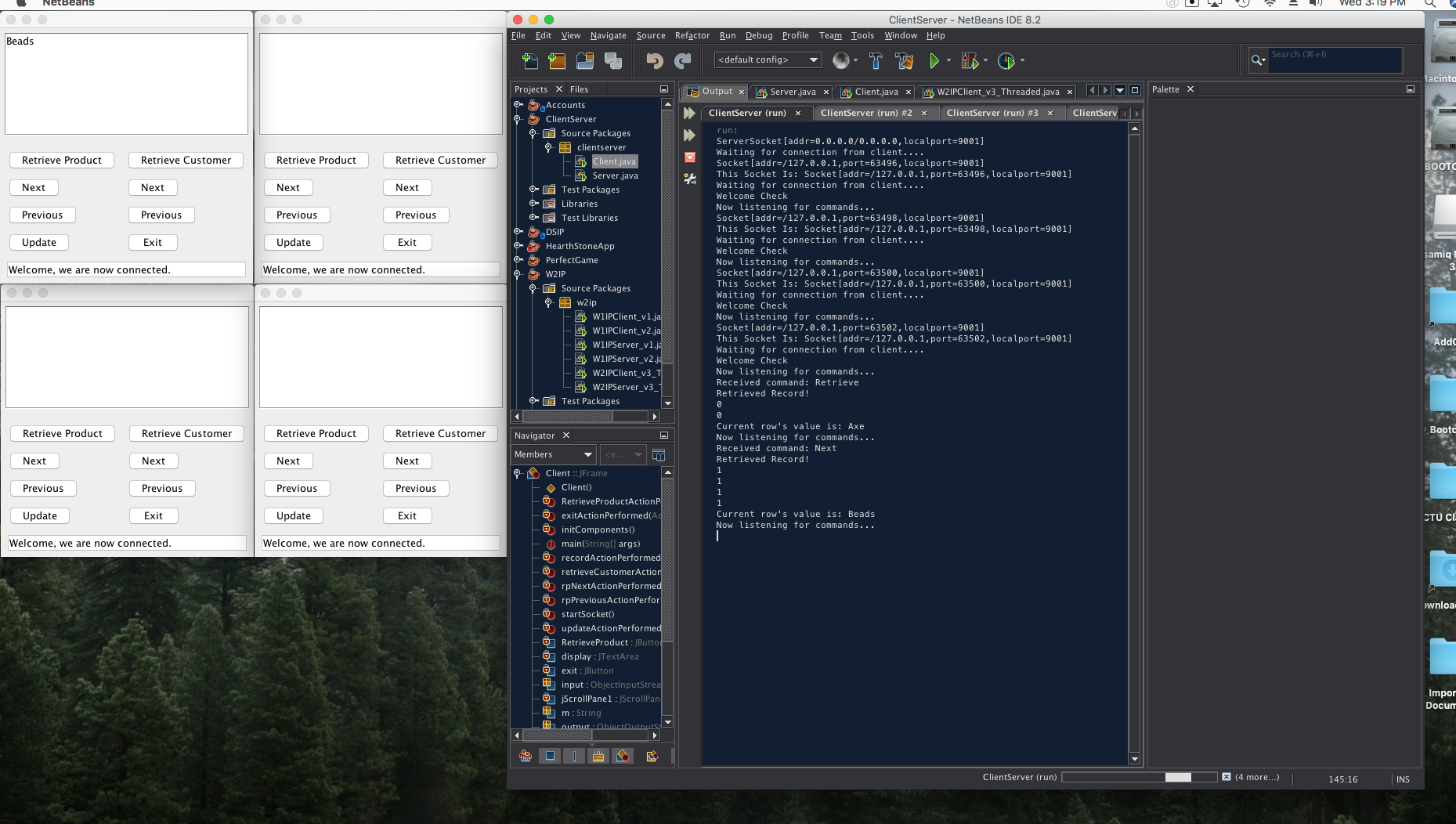
Socket class to establish connection points for both the server and client.

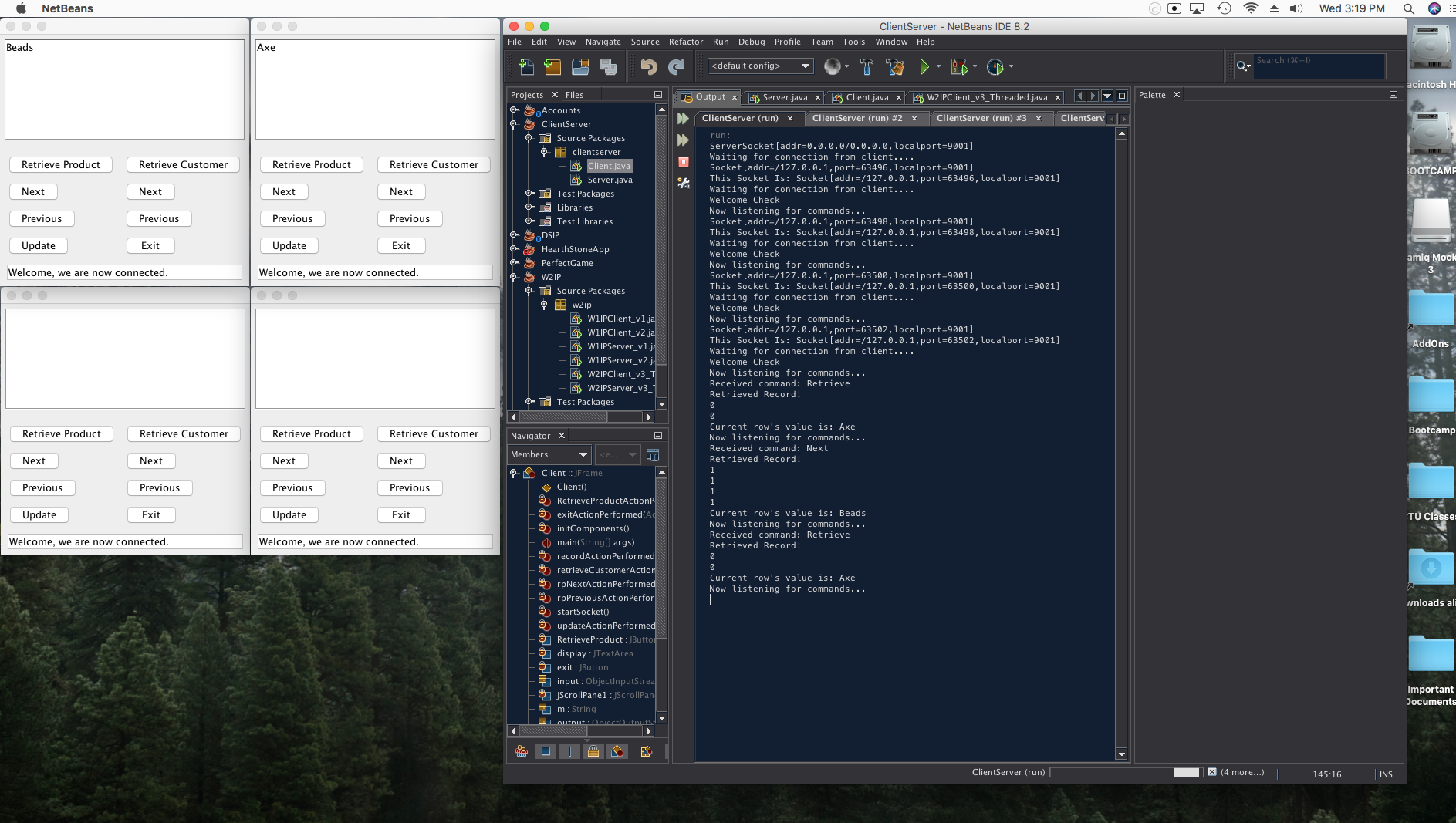
Run interface to implement the thread itself.

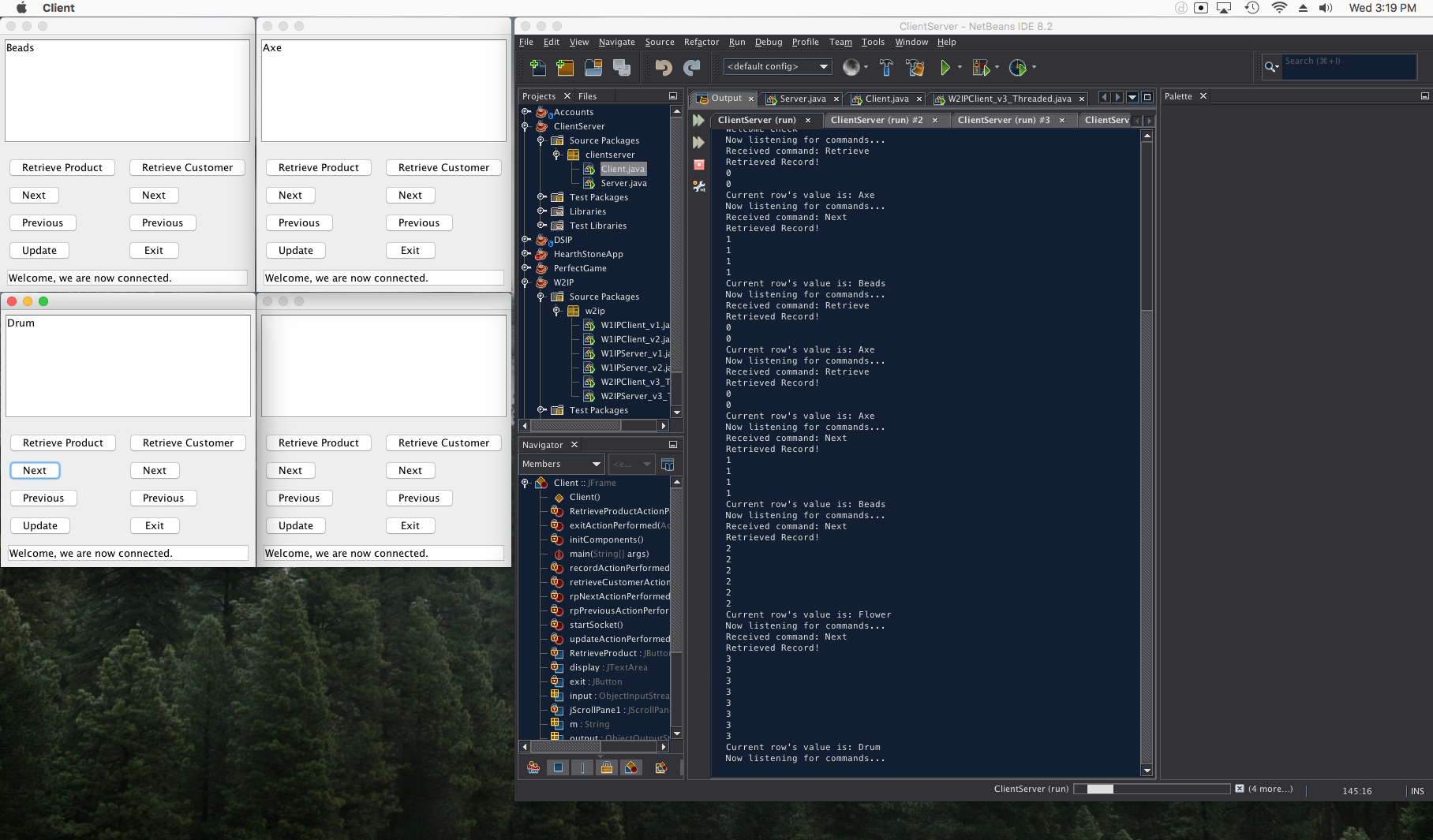
IO classes for input / display for the client side.

## Individual Project 2

**Client Connects**

**Client 1 Commands**

**Client 2 Commands**

**Client 3 Commands**

**GIF of Clients Cycling**

[**http://recordit.co/iMCogwtcQw**](http://recordit.co/iMCogwtcQw)

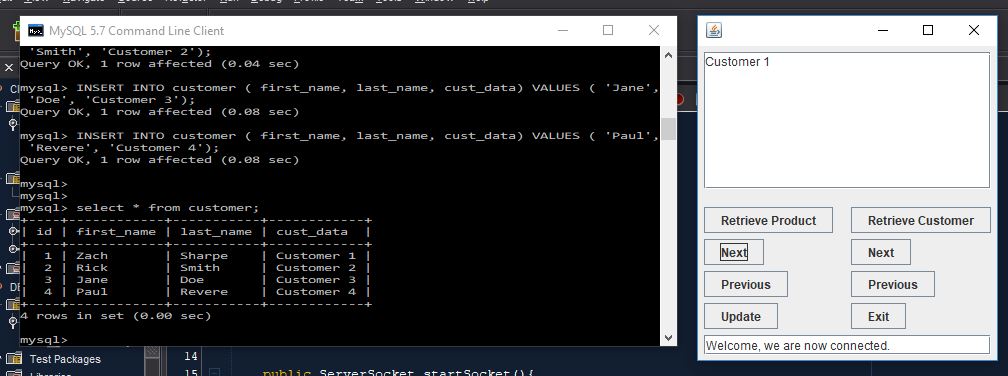
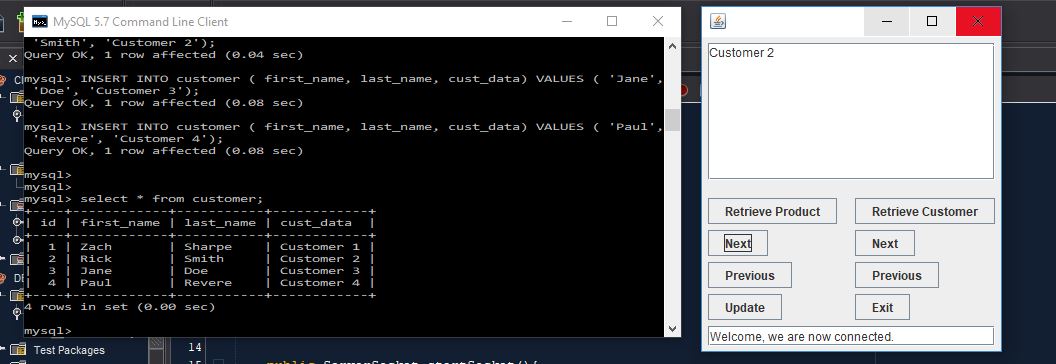
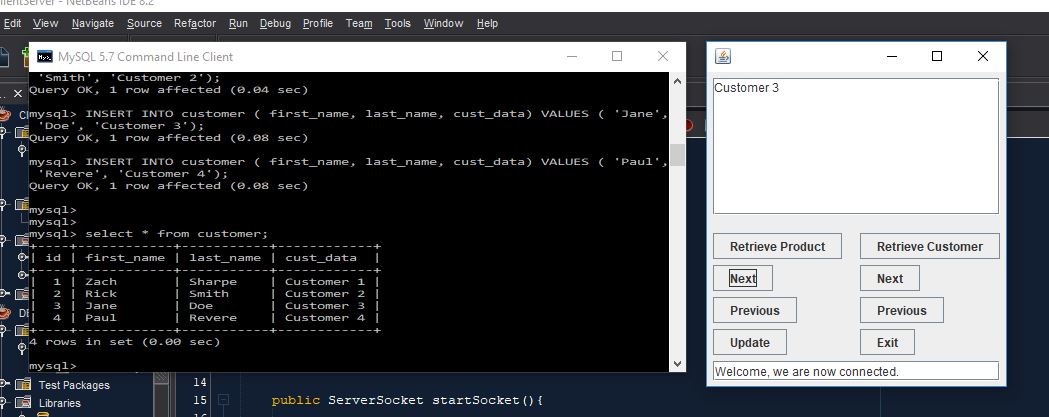
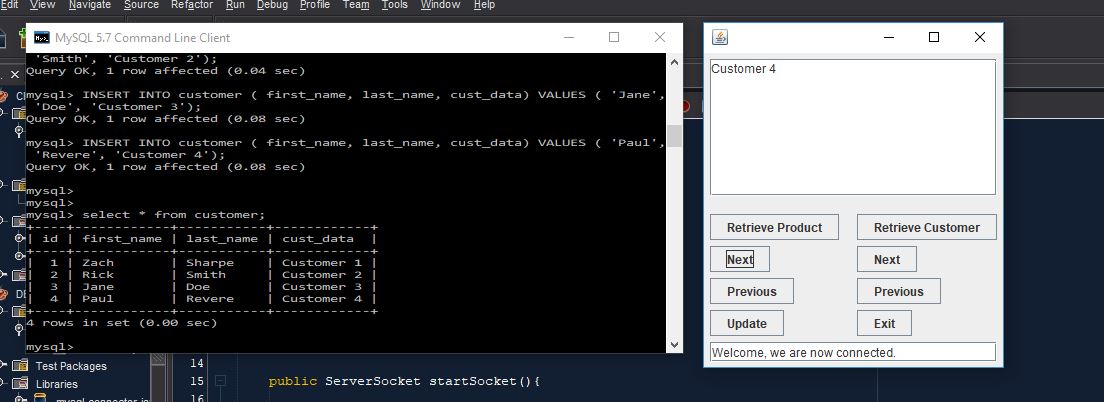
## Individual Project 3

While this week has been extremely challenging switching to a new coding environment (OSX to Windows 10) – I was able to get my jar added to the library and NetBeans connecting with my DB that was setup via MySQL.

Next / Previous are incrementing and decrementing with a few snags. I’m also experiencing a NullPointerException in my methods that handle iterating or finding row number that I was unable to resolve before turn in.

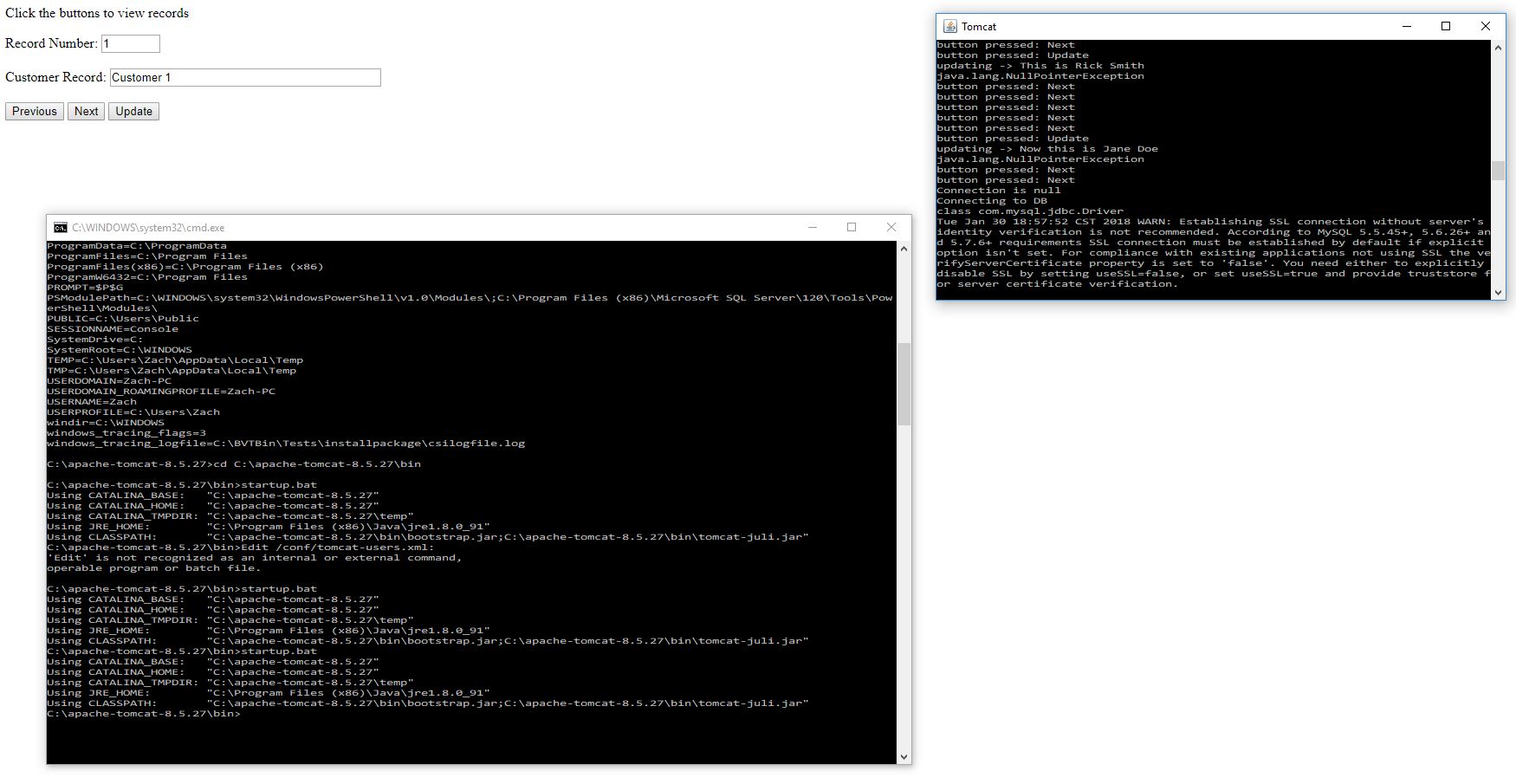
Lastly, the update interface will need to just be changed to have one set of next / previous / update buttons for customer data (not product).

Feedback appreciated! The following screenshots are the app rotating through my created DB table.



## Individual Project 4

**Tomcat / Command Line / Web App Screenshot**



**Video of application cycling using next / previous as well as updating then cycling**

<http://g.recordit.co/r5fGoaOMzH.gif>

**Embedded Source Code Object**



## Individual Project 5

* Explain how web services are used in the real world. Provide at least 3 real world examples.

In my current position we use AWS or Amazon Web Services to provide service to our customers. We provide email-marketing software that pulls from a database. AWS makes this possible.

In addition to AWS we provide an API documentation (api.myemma.com) that provides end points for third party applications to integrate with.

Lastly, I use the web version of Pandora often. This provides quick access to Internet radio.

* What are the benefits of web services?

Web services allow us to communicate using different languages to a browser for interaction or integration.

* How would you troubleshoot and test a web service? Would you use a debugger? Why or why not?

I would debug something like a JSP or XML in an editor like IntelliJ. I would also debug the Java associated with these in a respective editor. Troubleshooting JSON would likely be something I could look at. To test an application, I would just interface with it using a web browser most of the time. I would also use the console output for the web server that’s hosting it.

* Do you think that your client/server application could be enhanced to consume a web service or be used as a web service? Why or why not?

Certainly, I believe the JSP that is setup could already be converted to another Apache program that provides SOAP. Axis and CXF are both examples of this.

* Discuss the Java EE technologies you used and the reasons for your choice. Explain how these technologies achieved data persistence.

I used JSP and JDBC technologies primarily. Database technologies allowed for connection to a central database using MySQL. This allowed for data persistence outside of the life of the app. This is contrasted with internal data structures that automatically end as the program ends.

This was originally achieved by utilizing a .txt file but connecting to a true database server has provided a much smoother experience in terms of persistence.