**Requirements**:

* Node.js / JavaScript
* Make a server side and client side
* UI optional
* Client should authenticate with a key stored securely
* Verify a message they send is signed with their private key
* Comments
* Don’t reinvent the wheel
* Keep it simple

**To Run:**

1. Unzip package.
2. Open CMD.
3. cd into the unzipped directory.
4. Enter npm start. You should see “BitPay Assignment listening on port 3000.”
5. Open your browser and navigate to <http://localhost:3000/>. There you should see the front end.
6. On a 2nd tab, navigate to <http://localhost:3000/> again so that you may open two sessions.
7. Be sure that both tabs are in the same room channel as shown in the text box on the top right so that both tabs can communicate to each other.
8. Chat with other tab back and forth to test it. To be honest, there’s a little lag or delay in communicating between both but it shouldn’t be longer than 20 seconds.
9. While still having the CMD up, you should be able to see the output messages are not readable because they are encrypted.

**Libraries Used:**

* Vue.JS - <https://vuejs.org/>
* Node.JS - <https://nodejs.org/en/>
* Socket.io - <https://socket.io/>
* JSEncrypt (Stanford U.) – <http://travistidwell.com/jsencrypt/>
* Stanford Javascript Crypto Library - <http://bitwiseshiftleft.github.io/sjcl/>
* ExpressJS - <https://expressjs.com/>

**Resources Referred To:**

* <https://www.w3schools.com/nodejs/ref_crypto.asp>
* <https://socket.io/get-started/chat/>
* <https://crypto.stackexchange.com/questions/39940/should-signature-verification-work-using-two-different-rsa-libraries>
* <https://github.com/travist/jsencrypt/issues>
* <https://code.tutsplus.com/tutorials/real-time-chat-with-nodejs-socketio-and-expressjs--net-31708>
* <https://bitwiseshiftleft.github.io/sjcl/demo/>
* StackOverflow + GitHub Gists + CodePens

**Overview:**

In this application, 2048 bit RSA encryption is implemented to create a functioning end to end encrypted chat room that fill the above requirements.

**Concerns:**

Client side browser JavaScript encryption is a touchy subject in InfoSec due to the vulnerabilities and exploits that are inherited in web applications. Most of these issues can be resolved by using HTTPS to prevent man-in-the-middle attacks, but it is central to stay aware of potential vulnerabilities if this is ever hosted online. As it stands this app is good to run locally on your PC.