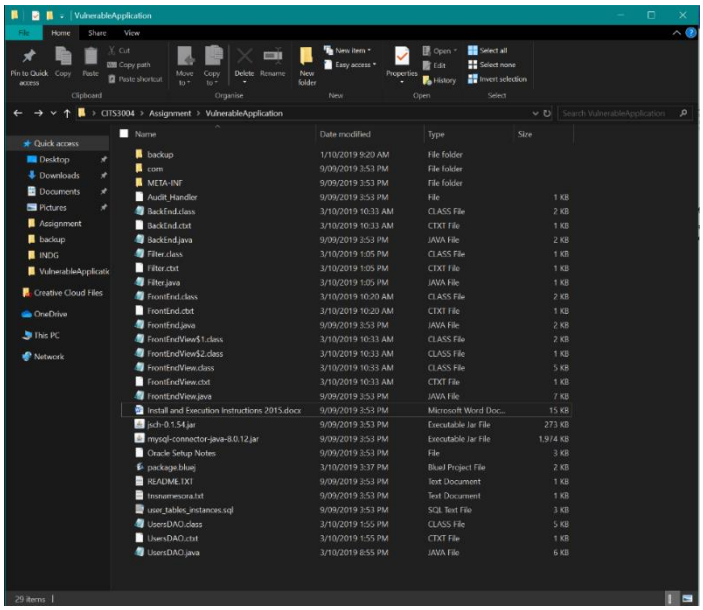
Harwinddranath Muralitheran Student ID: 22254937

CITS3004 Project: Task 3

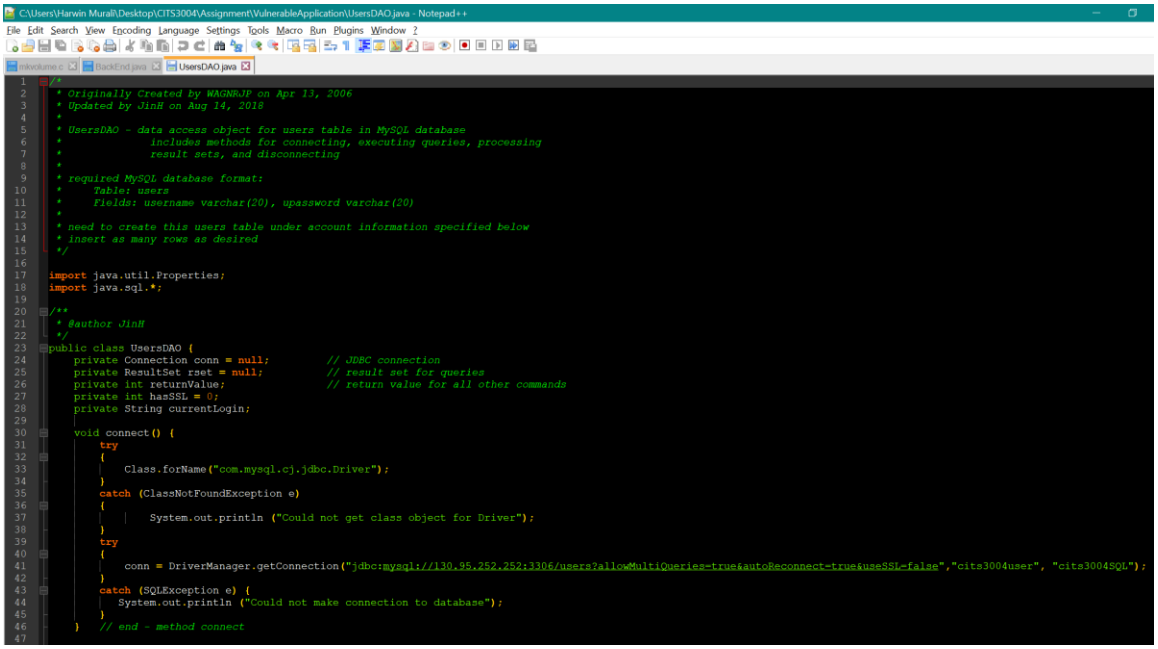
Vulnerability 1: Extracting files from the Vulnerable Application

By extracting the VulnerableApplication.jar file, we are able to access the application's contents and most importantly, the source codes of the file. From this, we can identify that BackEnd, Filter, FrontEnd, FrontEndView and UsersDAO are the main components of the file, and also that they were written in Java. A screenshot of the application's contents is provided below:



Vulnerability 2: Analysing the source code

Examining the source codes, we can get an idea as to how the application was constructed and designed to work. From the UsersDAO.java file, we can identify that the VulnerableApplication is running a MySQL database and extract the connection URL for the database ("jdbc:mysql://130.95.252.252:3306/users?allowMultiQueries=true&autoReconnect=true&useSSL=false") and (Username = "cits3004user", Password = "cits3004SQL").



Vulnerability 3: Manipulating source code

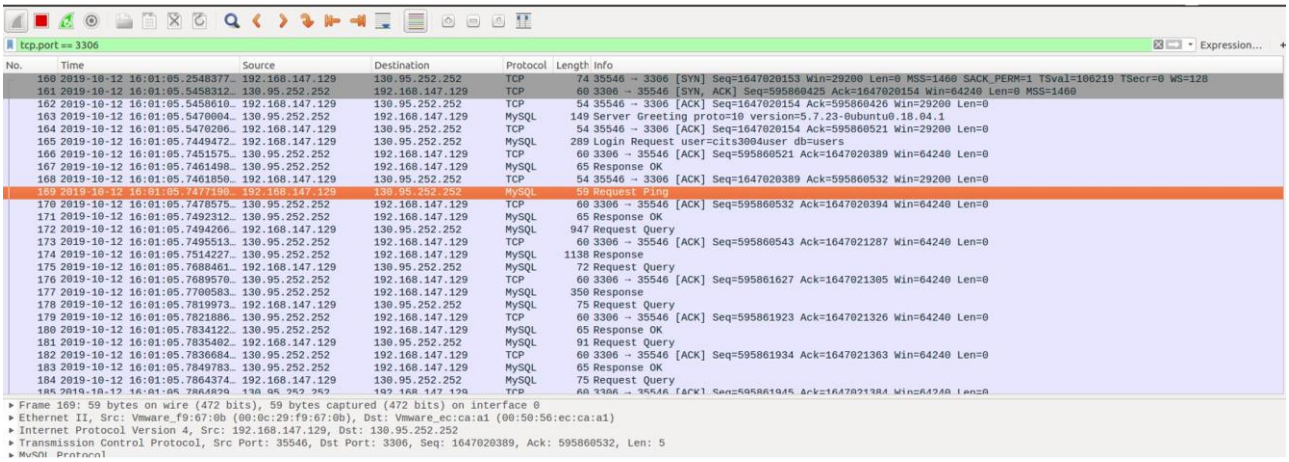
```
        Statement stmt = null;      // SQL statement object

        sqlQuery = "select * from users where " +
            "username = " + "'" + fe.getUsername() + "'" + " and " +
            "upassword = " + "'" + fe.getPassword() + "'";
        try {
           stmt = conn.createStatement();
           rset = stmt.executeQuery(sqlQuery);
        }
        catch (SQLException e) {
            System.out.println("Could not execute SQL statement: " + sqlQuery);
        }
    }
```

By manipulating the SQL variable template (shown in picture above), hackers can alter the ways queries are processed. For example, hackers could create an additional field that requests for the user's email address. By doing so, they can send phishing emails to the users, potentially tricking them into providing more personal information.

Within the source code, hackers may implement malicious codes that infects the user's system with viruses. Another option would be to implement a keylogger program within the source codes. As an example scenario, a hacker could imitate a lecturer (using the login details extracted from SQL injections, from earlier tasks) and send out an email (spear phishing) to their students with the infected program attached, and requesting them to use this updated version and thus be able to track the keyboard inputs of the users.

Vulnerability 4: Packet Sniffing



Using the Wireshark Packet Sniffer, we are able to capture the network traffic as they are transmitted over a TCP protocol and identify the platform that the database is running on, in this case, MySQL. We can also identify the source and destination ports (35546 and 3306 respectively), and the destination IPv4 address of the MySQL database (130.95.252.252 - consistent with the MySQL database connection URL mentioned above).

Vulnerability 5: Network/TCP Session Hijacking

With the information extracted from packet sniffing earlier, hackers can perform IP Spoofing to hijack the TCP session. Using the IP addresses, hackers can impersonate a different user's identity and send spoofed packets, compromising the integrity of the connection.