Software Limitations

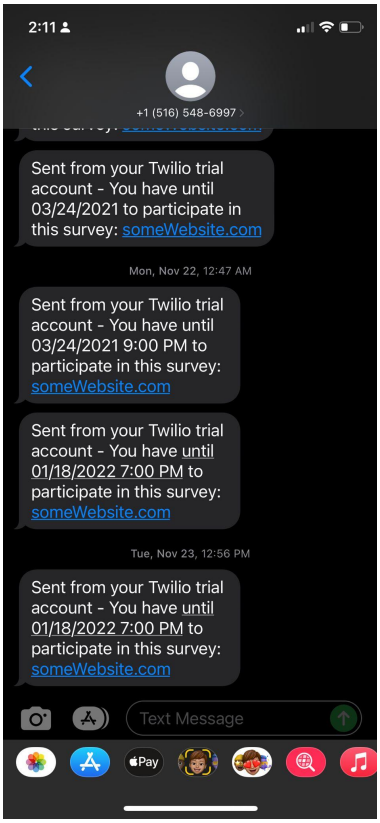
Messaging via Twilio

```
public Boolean messageUser(String phoN, String myText) {
    Twilio.init(ACCOUNT_SID, AUTH_TOKEN);

    try {
        Message message = Message.creator(new PhoneNumber(phoN), new PhoneNumber("+15165486997"), // twilio number
                myText).create();
    } catch (Exception e) {
        // System.out.println(e.getMessage());
        return false;
    }
    // System.out.println(message.getSid());
    return true;
}
```

This is the code that allows our project to be able to send messages via twilio. In other to run this simple code and message a phone number there are several items needed. ACCOUNT_SID and AUTH_TOKEN are generated when creating a twilio account and gives us permission to run the method, but we also need a phone number where we send messages from. In this picture +15165486997 is that phone number.

The limitation we have for messaging users in the current version of our project is that we are not allowed to send messages to any phone number. In the picture, the variable named phoN represents the reciver of the message, and that phone number has to be verified in our twilio account if we wish to send a message to that number.



I simple and effective solution for this, is to upgrade our twilio account to the paid version which will enable our project to be able to send messages to phone numbers without the need of having to verify them in our account first.

Lastly, another limitation encountered while messaging is that the account may get blocked. We are unsure what causes this problem, but we suspect it's caused by trying to message an unverified number repeatedly.

executeSelectedQuery() Method

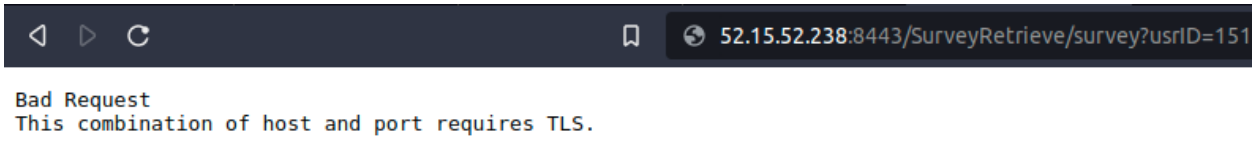
```java
public String executeSelectedQuery(String queryType, String[] infoToProcess) {
    DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
    String strToReturn = "no result yet";//String mainly used for retrieval of information.
    ResultSet rs = null;
    Boolean insert = true;
    //MORE CODE BELLOW
```

The executeSelectedQuery method is in charge of searching through the SQL repository defined inside a XML file. As seen in the picture above, this method is meant to return a String. The main reason for this to return string is due to the need of executing queries that retrieve data from the database. However, in the current version of our project, this method is unable to return a row of a table or an entire table. It only returns an specific entry of a given field.

```xml
<queries>
    <query name="getSystemDateTime">
        <sql paramNum="0" insert="0">SELECT NOW()</sql>
    </query>
    <query name="getSurveyQuestion">
        <sql paramNum="1" insert="0">SELECT * FROM Question WHERE QID=?;</sql>
        <parameters>
            <param>INT</param>
        </parameters>
    </query>
    <query name="depositResponse">
        <sql paramNum="5" insert="1">INSERT INTO QResponce VALUES(?,?,?,?,?);</sql>
        <parameters>
            <param>INT</param>
            <param>INT</param>
            <param>INT</param>
            <param size="10">VARCHAR</param>
            <param>LONGTEXT</param>
        </parameters>
    </query>
```

Servlet

My servlet is unable to connect to HTTPS because I did not install the required TLS to connect to mySQL in tomcat9. However this task can be completed in the future. It also does not provide the desired user interface.



MySQL

My mySQL schema lacks certain uniqueness constraints however this was the ideal setup for testing, in the future further constraints can be made when we are certain of the values. MySQL also enforces SSL however it has not been verified by a sniffer.

SurveyExtIns.jar

The SurveyExtIns.jar is the jar that we have implemented to the servlet. However this jar contains the code that does not implement the query repository. SurveyExtIns2.jar contains the most recent code created to implement the repository, however when exported and used in the servlet, we were having issues with getting the methods to detect the queries from the repository.

Queries

```
String query1 = "Select QID\r\n" +
        "from Question\r\n" +
        "where Survey_ID = '" + S_ID +"'";
PS = connect.prepareStatement(query1);
result = PS.executeQuery();
while (result.next()) {

query1 = "Select QCID\r\n" +
        "from QuestionChoices\r\n" +
        "Where QID = '"+ reference.get("QID")+"'";
PS = connect.prepareStatement(query1);
result2 = PS.executeQuery();
while(result2.next()) {
```

Due to the executeSelectedQueries() method being unable to send the result of multiple rows and columns, my SurveyTest() method is unable to fully rely on the repository. To make sure all of the survey questions and question choices are printed out, my code needs to iterate through each row of the result from a query so i am unable to use the repository for that

JSONObject

```
JSONObject SurveyTest
```

While the JSONObject returned by my SurveyTest method does extract and contain all of the correct information needed by the Survey, if we were to write that JSONObject to a file, we can see that the contents of the file are out of order. All of the objects and information are grouped together correctly, however within those groups, the information is not ordered in the

way the items are put in due to the way JSONObject class is able to order information in a group any way they want.

ResponseInsert

The current ResponseInsert puts information into the database off of a txt file with the information formatted like SurveyID#UserID#QID#TypeID#Response. It does not read from a JSONObject.