

Introduction:

The program I've coded is called IGVA which stands for 'Instagram Virtual Assistant'. This program serves multiple functions to online business owners who conduct their marketing via Instagram (A social media platform). This program includes scraping users from Instagram and automatically commenting on their posts. This solves the problem of paying virtual assistants to do this manually, which has saved it's users hundreds of dollars per month.

Summary of Programming Techniques:

- parameter passing
- random number generation
- for loop
- while loop
- do while loop
- nested loops
- recursion
- method returning a value
- arrays, 2d arrays
- user defined objects made from an OOP "template" class
- encapsulation of private methods that work on public attribute of a "template" class
- Polymorphism, extending a GUI class to JFrame.
- making an array of objects
- simple and compound selection (if/else)
- searching (linear search, binary search...)
- Storing data in a local SQLite database
- Fetching data from local SQLite database
- error handling (for example catching a divide by 0 error, or a null pointer while using an array of object...)
- Option pane generation for communicating with the user
- GUI tabs
- GUI popup menus
- overloaded constructors, which work differently depending on the parameters sent
- use sqlite and selenium libraries
- linked list, or any other ADT, like binary search tree class...
- Translating requests from Java to SQL.
- Using methods within different classes to serve different functions.
- Creating an NSIS (Nullsoft Scriptable Install System) installer for the ease of downloading and running the program.
- Running operations concurrently using Threads object.
- Throwing exceptions
- Working with files as drivers
- Web cookies utilization

Unique Programming Techniques:

- Java communicating to SQLite example:

```
public void addUser(String user){  
    try {  
        String query = " insert into users values ('"+user+"');";  
        PreparedStatement p = con.prepareStatement(query);  
        p.execute();  
    } catch (SQLException ex) {  
        ex.printStackTrace();  
    }  
}
```

- Fetching a Profile object that gets it's values from the SQLite database.

```
public Profile getAccount(SQL sql, int i) throws SQLException {  
    Profile profile = new Profile(sql.getProfile(i), sql.getSession_id(i));  
    return profile;  
}
```

Structure of program

Simply, the program, IGVA, serves the user two functions, scraping and commenting. In the early stages of development, IGVA was run using an IDE and did not include a GUI feature. The current version of IGVA featuring a user interface also allows the user to see the data which has been scraped, the data used in the past, and other information about data that the program needs in order to run.

The data used in this program is directly used from the SQLite local database and not by RAM. The reason the data is not pulled out of the database into a data structure which is temporarily stored in the RAM while the program is running, is to prevent data loss. Since RAM is volatile, if for any reason the computer running the program shuts off, and changes made to the data will be lost.

The program is composed of 2 packages and 14 classes in order to organise sections of the program which allow for development to become a more efficient process. The class 'Do' is used by many of the classes as it holds general functions that are used for the automation, for example, waiting a certain number of seconds, setting the property for the web driver, generating random numbers within a range and modifying strings.

OOP features

The Profile class allows for the ease of working with Instagram profiles by holding the username and sessionid cookie value for a profile from a particular web session, and making it easier to fetch the data from the SQL feature using the getProfile and getSession_id methods.

Data Structures Used

There were no other data structures they needed to handle the data in the program as it was stored using a local SQLite database. The only data structure used was array and arrays of objects for fast execution of the scraping method. The methods were all run in parallel using the Thread object as an array of threads.

Unique Algorithms

- When adding a username, the user is prompted with a window and after logging in, the program should close the window automatically. It does this by looking for an element which only appears on the login page, so when this element is no longer visible it will close the window.

```
private String sessionIdRetriever() throws InterruptedException {
    Do.setProperty();
    String session_id = "";
    WebDriver driver = new ChromeDriver();
    String url = "https://instagram.com";

    driver.get(url);
    Do.waitSec(5);
    driver.findElement(By.name("username")).sendKeys(usernameTF.getText());
    while (onLoginPage(driver)) {
        Do.waitSec(1);
        System.out.println("waiting");
    }
    Do.waitSec(3);
    try {
        session_id = driver.manage().getCookieNamed("sessionid").getValue();
    } catch (Exception e) {}
    driver.quit();
    return session_id;
}

private boolean onLoginPage(WebDriver driver) {
    try {
        driver.findElement(By.name("username"));
        return true;
    } catch (Exception e) {
        return false;
    }
}
```

- It is unnecessary to check when a scraper or commenter thread is running, therefore I wrote this method which tries to see if a the thread is alive, if it isn't an error will be thrown, so in the catch statement it can simply be returned that the thread is not alive, otherwise, the boolean isAlive will equate to 'true' and be returned after the statement is complete.

```
private boolean scraperIsAlive() {  
    boolean isAlive = false;  
    try {  
        isAlive = scraper.isAlive();  
    } catch (Exception ignored) {  
        return scraperRunning;  
    }  
    return isAlive;  
}
```

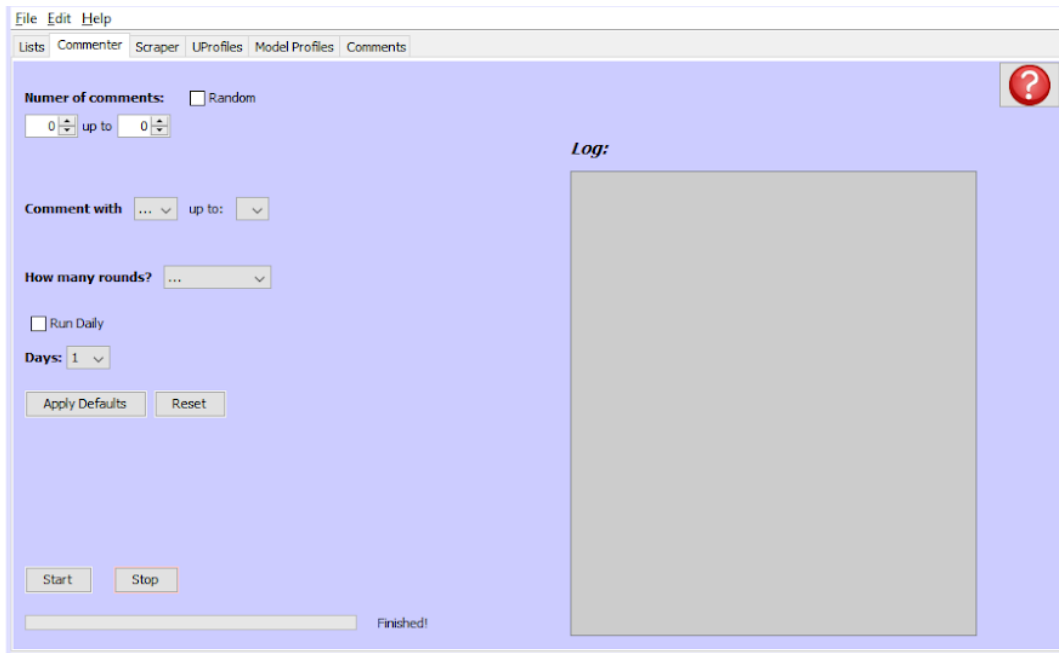
Software Tools Used

- Chrome
- IntelliJ
- Command Prompt
- NetBeans
- Nullsoft Scriptable Install System (NSIS)

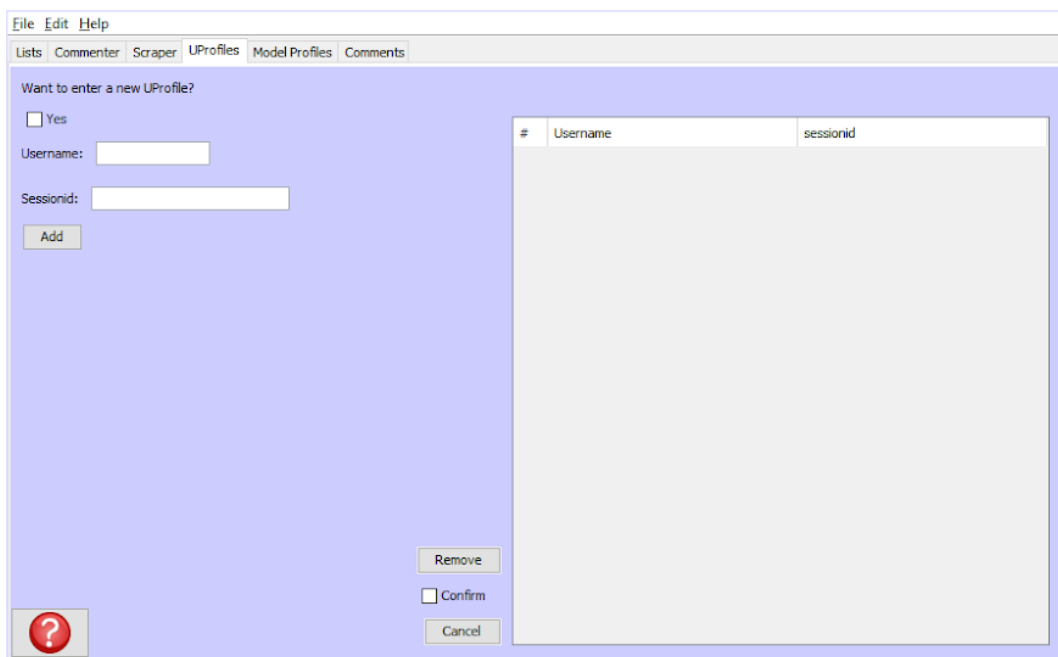
The main software I used was IntelliJ to write the backend of the program. I used IntelliJ to test and modify the code before adding it to my project in NetBenas. NetBeans provided ease of working with GUI components, so it was used to create the frontend.

Command Prompt is a windows command-line interpreter which was used to communicate with the database when it was first created. The tables were created using this application before the SQLite database was added to the program.

User interface work



Swing GUI elements such as JLabels, JTextAreas, JComboBoxes and more were used in the program to provide the user with a modern interactive experience.



Additionally, JLabels and JTables were used to get user input, send it to the local SQLite database, and finally, display it on the table. JTables allow the user to have a visual representation of their data.