

Development

Techniques Used:

1. Database and SQL Queries
2. ArrayList
3. Data Validation
4. Nested Loops
5. File Open/Save & Printer Output
6. Interactive Interface (GUI) & MouseEvent
7. Additional Libraries

1. Database and SQL Queries

A database was found to be the most effective system to store all teacher and student information. Both database tables, 'teacher' and 'subject', were formatted using phpMyAdmin¹ with data constraints. Using the domain constraint, this allows for the storage of only the correct data type, preventing repetition and errors when registering teachers and subjects.

Both databases must first be initialised prior to the running of the program on the client's desktop, with the assistance of Wampserver² to continue running. This is to ensure that there is a reliable connection between the Java program and the database tables.

Two database tables, for teachers and subjects, are seen in Figure 1.

¹ phpMyAdmin. "PhpMyAdmin." PhpMyAdmin, 2019, www.phpmyadmin.net/.

² WampServer. "WampServer." WampServer, 2012, www.wampserver.com/en/.

Figure 1

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	subjectCode	int(10)			No	None			Change Drop More
<input type="checkbox"/> 2	subjectName	varchar(50)	latin1_swedish_ci		No	None			Change Drop More

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	TeacherID	int(15)			No	None			Change Drop More
<input type="checkbox"/> 2	Name	varchar(70)	latin1_swedish_ci		No	None			Change Drop More
<input type="checkbox"/> 3	Email	varchar(70)	latin1_swedish_ci		No	None			Change Drop More
<input type="checkbox"/> 4	PhoneNumber	int(15)			No	None			Change Drop More

In order to add and delete entries in the database, SQL Language must be used. The SQL statement that is utilized when inserting the teacher information into the database table, is seen below:

```
"INSERT INTO `teacher`(`TeacherID`, `Name`, `Email`, `PhoneNumber`)
VALUES ([value-1],[value-2],[value-3],[value-4])"
```

This is then incorporated into the program code by importing `JavaSQL3`, the package that allows for a connection to the database:

Figure 2

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
```

³ Java. "Java.Sql (Java Platform SE 8)." Docs.Oracle.Com, docs.oracle.com/javase/8/docs/api/java/sql/package-summary.html.

In Figure 3, it shows the execution of the code when the “Insert Teacher” button is clicked. A try-catch statement has been utilized. The try-catch statement helps in defining a segment of code to be tested for errors when executed. If an error occurs, the catch statement will be executed (an error message). This will be beneficial in meeting the needs of the client, to ensure teacher and subject registration are successful.

Figure 3

```
private void insertTActionPerformed(java.awt.event.ActionEvent evt) {  
  
    try {  
        //the sql used to insert a teacher entry  
        String sql = "INSERT INTO teacher"  
            + "(TeacherID, Name, Email, PhoneNumber)"  
            + "VALUES (?, ?, ?, ?)";  
        //connect to database  
        con = DriverManager.getConnection("jdbc:mysql://localhost/teacher", "root", "");  
        //prepare sql  
        pst = con.prepareStatement(sql);  
        //acquire and collect text from textboxes  
        pst.setString(1, idboxT.getText());  
        pst.setString(2, nameboxT.getText());  
        pst.setString(3, emailboxT.getText());  
        pst.setString(4, phoneboxT.getText());  
        //execute sql  
        pst.executeUpdate();  
        //show alert message  
        JOptionPane.showMessageDialog(null, "Inserted Successfully!");  
    } catch (SQLException | HeadlessException ex) {  
        //unsuccessful insertion message:  
        JOptionPane.showMessageDialog(null, "Invalid Details. Try Again.", "Insertion Error", JOptionPane.ERROR_MESSAGE);  
    }  
    showTableData();  
}
```

If an addition is made to the database table, the “showTableData” class (Figure 4) is executed. This class will prompt the execution of another SQL query to fetch the new data, updating the table.

Figure 4

```
public void showTableData() //refresh the data in the table
{
    try{
        //connection to database
        con = DriverManager.getConnection("jdbc:mysql://localhost/subject", "root", "");
        //sql for reading the database table
        String sql = "SELECT * FROM subject";
        pst = con.prepareStatement(sql);
        rs = pst.executeQuery();
        //refresh the table
        TableT.setModel(DbUtils.resultSetToTableModel(rs));
    }
    catch (Exception ex) {
        JOptionPane.showMessageDialog(null, ex);
    }
}
```

The execution of this class “showTableData” can also be found when there is an deletion of an entry, where the JTable must be updated. This is clever because an update button is no longer required for the interface, and the code will not have to be written or repeated multiple times. Moreover, this meets the needs of the client to display all necessary teacher information when desired.

Furthermore, the database entries seen in Figure 5 are linked to the choice boxes in the timetable interface, allowing the client to choose the desired option (Figure 6).

Figure 5










				subjectCode	subjectName
<input type="checkbox"/>	 Edit	 Copy	 Delete	25	Biology
<input type="checkbox"/>	 Edit	 Copy	 Delete	34	Chemistry
<input type="checkbox"/>	 Edit	 Copy	 Delete	59	Physics

Figure 6

un - Feb 16	Mon - Feb 17	Tue - Feb 18

Choose A Teacher:

jonathan

Choose A Subject:

Biology
Biology
Chemistry
Physics

Choose A Class:

Year 9

In Figure 7, the subject choice box will be updated using JavaSQL to perform a query and get the most updated information in the database table. The selected option in the choice box would then be imprinted on the reservation slot.

Figure 7

```
private void initializeSubjects() {  
  
    try {  
        //connection to database  
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost/subject", "root", "");  
        Statement stmt = con.createStatement();  
        //execution of SQL to read subject database table  
        ResultSet rs = stmt.executeQuery("SELECT * FROM subject");  
        while (rs.next()) {  
            //adding the subject from database table to the dropbox in timetable interface  
            Task subject = new Task();  
            String name = rs.getString("subjectName");  
            subject.setName(name);  
            subjects.add(subject.getName());  
            calendar.getTasks().add(subject);  
            subjectsList.add(subject);  
        }  
    } catch (SQLException | HeadlessException ex) {  
    }  
}
```

This is an example of ingenuity because it is a creative method to avoid the tedious action of typing the subject, teacher and class when making each reservation, meeting the needs of the client.

2. ArrayList

An ArrayList⁴ has been the data structure of choice for the timetable section, holding all subject and teacher data. An ArrayList provides for many advantages over a regular array. This includes the ability for it to be resizable, and dynamically create and delete elements within it. Furthermore, the use of an ArrayList fulfills the requests of the client who will continue to add or remove teachers and subjects as the school continues to grow or expand. This demonstrates ingenuity because the future extensibility of the product has been considered with a more versatile data structure. The initialisation is seen in Figure 8.

Figure 8

```
contactsList = new ArrayList<Contact>();  
subjectsList = new ArrayList<Task>();
```

⁴Java. "ArrayList (Java Platform SE 8)." Oracle.Com, 11 Sept. 2019, docs.oracle.com/javase/8/docs/api/java/util/ArrayList.html.

3. Nested Loops

Nested loops are a crucial component of the interactive timetable and is required for several functions. Nested loops are useful when dealing with multiple iterations or conditions, in which memory size will be reduced. This will in the long-term benefit the client by saving memory and increasing the efficiency in code. Particularly, when a space on the timetable is clicked, algorithmic creativity through a nested loop is utilized to handle the certain activity, seen below:

Figure 9

```
private void onCalendarClicked(MouseEvent e) {
    if (ignoreNextClick)
    {
        //ignore the click
        ignoreNextClick = false;
        return;
    }
    if (currentColumnBounds != null)
    {
        //if bound selected
        Point point = new Point(e.getXOnScreen() - calendar.getLocationOnScreen().x,
            //collecting the locations that the mouse has clicked on the screen
            e.getYOnScreen() - calendar.getLocationOnScreen().y);
        if (currentColumnBounds.contains(point))
        {
            //if within the specific bound:
            Point p = currentColumnBounds.getLocation();
            for (Item item : allItems)
            {
                //reservation box padding
                int padding = 2;
                //new rectangle created between the specific bounds
                Rectangle itemBounds = new Rectangle(p.x + padding, p.y + padding,
                    currentColumnBounds.width - 2 * padding, calendar.getItemSettings().getSize());
                //within the reservation contains the object itself
                if (itemBounds.contains(point))
                {
                    JOptionPane.showMessageDialog(this, item.getHeaderText() + " was clicked!");
                }
                //creation of a new point that is used in the creation of the reservation
                p = new Point(p.x, p.y + calendar.getItemSettings().getSize() + padding);
            }
            return;
        }
    }
}
```


4. Data Validation

The client must be made aware if there is an error when wanting to create or delete entries from the database. For integers in particular, this is incorporated through an error message seen above the text field using `NumberFormatException`⁵ (Figure 10). This displays ingenuity because instead of a standard error message, this does not get in the way of the client, while also ensuring a successful insertion of an entry.

Figure 10

```
private void idboxTKeyPressed(java.awt.event.KeyEvent evt) {  
    try {  
        int i = Integer.parseInt(idboxT.getText());  
        validid.setText("");  
    }  
    catch (NumberFormatException e) {  
        validid.setText("Please input an integer!");  
    }  
}
```

TeacherID
1234
126
124
1234455
1234444

⁵ Java. "NumberFormatException (Java Platform SE 7)." Docs.Oracle.Com, docs.oracle.com/javase/7/docs/api/java/lang/NumberFormatException.html.

5. File Open/Save & Printer Output

In order to successfully save, the JFileChooser has been utilised in combination with the Mindfusion⁶ external library to collect all necessary bookings and collate it into an XML file. The “Save” and “Open” file option can be found in the menu bar (Figure 11), involving the use of JMenuBar for ease of access. The save feature is crucial as it has been highly requested by the client, to ensure progress can be saved and worked on at another time.

Figure 11

```
//create menubar
JMenuBar menuBar = new JMenuBar();
//menubar heading
JMenu mFile = new JMenu("Menu");
//menubar "Open" function
JMenuItem mIFOpen = new JMenuItem("Open");
mIFOpen.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        openFileClicked();
    }
});
//menubar "Save" function
JMenuItem mIFSave = new JMenuItem("Save");
mIFSave.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        saveFileClicked();
    }
});

menuBar.add(mFile);
mFile.add(mIFOpen);
mFile.add(mIFSave);
setJMenuBar(menuBar);
```

⁶ Mindfusion. “MindFusion UI Controls for Web, Mobile and Desktop Applications.” Mindfusion.Eu, 2001, mindfusion.eu/.

Figure 12 displays the open and save functions:

Figure 12

```
//when save button is clicked
protected void saveFileClicked() {
    if (fileChooser.showSaveDialog(this) == JFileChooser.APPROVE_OPTION)
    {
        //save to XML and obtain the path
        calendar.getSchedule().saveTo(fileChooser.getSelectedFile().getAbsolutePath(), ContentType.Xml);
    }
}

// when file button is clicked in menu bar
protected void openFileClicked() {
    if (fileChooser.showOpenDialog(this) == JFileChooser.APPROVE_OPTION)
    {
        //path for selecting file, and the file chooser menu
        calendar.getSchedule().loadFrom(fileChooser.getSelectedFile().getAbsolutePath(), ContentType.Xml);
        if (calendar.getGroupType() != GroupType.None)
        {
            //collect all the necessary information on the calendar
            calendar.beginInit();
            calendar.getContacts().clear();
            calendar.getResources().clear();
            calendar.getLocations().clear();
            calendar.getTasks().clear();
            calendar.getContacts().addAll(calendar.getSchedule().getContacts());
            calendar.getResources().addAll(calendar.getSchedule().getResources());
            calendar.getLocations().addAll(calendar.getSchedule().getLocations());
            calendar.getTasks().addAll(calendar.getSchedule().getTasks());
            calendar.endInit();
        }
    }
}
```

The Mindfusion external library has been utilized for the other two functions found in the menu bar: Print and Print Preview (Figure 13). The print feature meets the needs of the client, where the timetable of the week can be printed for teachers to view. This displays ingenuity because less code had to be written, thus saving development time.

Figure 13

```
JMenuBar menu = getJMenuBar();
if (menu != null)
{
    JMenu menuFile = menu.getMenu(0);

    menuFile.addSeparator();

    JMenuItem menuPrintPreview = new JMenuItem("Print Preview");
    menuPrintPreview.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            calendar.printPreview();
        }
    });
    menuFile.add(menuPrintPreview);

    JMenuItem menuPrint = new JMenuItem("Print");
    menuPrint.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            calendar.print();
        }
    });
    menuFile.add(menuPrint);
}
```

6. Interactive Interface (GUI) & Mouse Event

An interactive interface was also a priority for the client, where the program can be operated without hassle. The GUI for the login screen, main menu, and registration have all been achieved with JavaSwing. This includes the use of JTitles, JButtons, JLabels, JPanels and JTables. JavaSwing was chosen because of its wide component palette that helped achieve the designs desired by the client. The use of NetBeans and its design view aided in the positioning of components.

Figure 14

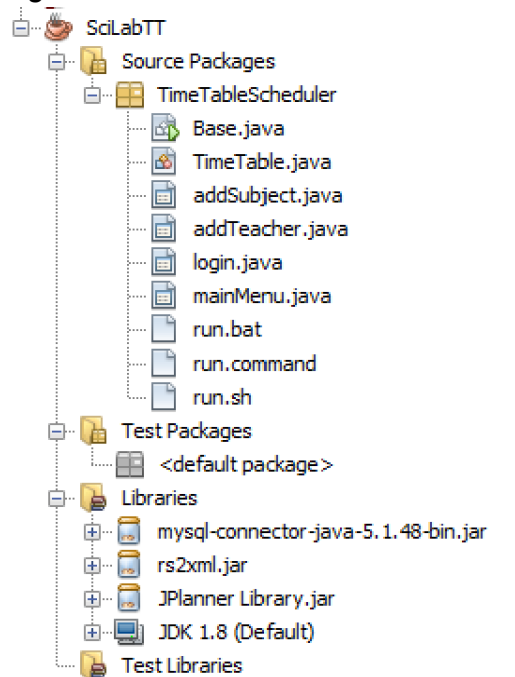
```
private void initComponents() {  
  
    jLabel14 = new javax.swing.JLabel();  
    jPanel11 = new javax.swing.JPanel();  
    jLabel13 = new javax.swing.JLabel();  
    jLabel15 = new javax.swing.JLabel();  
    jpasswordbox = new javax.swing.JPasswordField();  
    jusernamebox = new javax.swing.JTextField();  
    jBlogin = new javax.swing.JButton();  
    jBreset = new javax.swing.JButton();  
    jBexit = new javax.swing.JButton();  
  
    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);  
  
    jLabel14.setFont(new java.awt.Font("Arial", 1, 34)); // NOI18N  
    jLabel14.setText("Science Lab Timetable Scheduler");  
  
    jPanel11.setBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 0, 0), 3));  
  
    jLabel13.setFont(new java.awt.Font("Arial", 1, 18)); // NOI18N  
    jLabel13.setText("Username:");  
  
    jLabel15.setFont(new java.awt.Font("Arial", 1, 18)); // NOI18N  
    jLabel15.setText("Password:");  
  
    jBlogin.setFont(new java.awt.Font("Arial", 3, 14)); // NOI18N  
    jBlogin.setText("Login");  
    jBlogin.setBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 0, 0)));  
    jBlogin.addActionListener(new java.awt.event.ActionListener() {  
        public void actionPerformed(java.awt.event.ActionEvent evt) {  
            jBloginActionPerformed(evt);  
        }  
    });  
  
    jBreset.setFont(new java.awt.Font("Arial", 3, 14)); // NOI18N  
    jBreset.setText("Reset ");  
    jBreset.addActionListener(new java.awt.event.ActionListener() {  
        public void actionPerformed(java.awt.event.ActionEvent evt) {  
            jBresetActionPerformed(evt);  
        }  
    });  
}
```

Meanwhile, the GUI for the timetable had been formed with assistance from the JPlanner library⁷, where multiple classes and tools were incorporated. The use of these classes also aided in the simple creation of the rectangles for the timetable, where MouseEvent was utilized when dragging the mouse to create a reservation slot, which was familiar.

7. Additional Libraries

Additional libraries were utilized to improve functionality and save development time. This includes: Rs2xml⁸, for the manipulation of JTables, and the JPlanner library from Mindfusion⁹, for aid in the formation of the timetable interface. The JPlanner library also paved the way for the handling of reservation slots and the setting up the dates and timings (using in-built classes). The program's source code can be found in Appendix 7.

Figure 15



Word Count: 1124

⁷ Mindfusion. "MindFusion UI Controls for Web, Mobile and Desktop Applications." Mindfusion.Eu, 2001, mindfusion.eu/. Accessed 15 Mar. 2020.

⁸ Hack Smile. "Rs2xml.Jar." Hack Smile, 17 June 2019, hacksmile.com/rs2xml-jar-free-download/.

⁹ Mindfusion. "MindFusion UI Controls for Web, Mobile and Desktop Applications." Mindfusion.Eu, 2001, mindfusion.eu/.

Bibliography:

Hack Smile. "Rs2xml.Jar." Hack Smile, 17 June 2019, hacksmile.com/rs2xml-jar-free-download/.

Java. "ArrayList (Java Platform SE 8)." Oracle.Com, 11 Sept. 2019, docs.oracle.com/javase/8/docs/api/java/util/ArrayList.html.

Java. "Java.Sql (Java Platform SE 8)." Docs.Oracle.Com, docs.oracle.com/javase/8/docs/api/java/sql/package-summary.html.

Java. "NumberFormatException (Java Platform SE 7)." Docs.Oracle.Com, docs.oracle.com/javase/7/docs/api/java/lang/NumberFormatException.html.

Mindfusion. "MindFusion UI Controls for Web, Mobile and Desktop Applications." Mindfusion.Eu, 2001, mindfusion.eu/.

phpMyAdmin. "PhpMyAdmin." PhpMyAdmin, 2019, www.phpmyadmin.net/.

WampServer. "WampServer." WampServer, 2012, www.wampserver.com/en/.