**Criterion C: Development**

Table of Contents

[1. MySQL Creation (Physical Schema) 3](#_Toc159784560)

[2. List of Complexities 4](#_Toc159784561)

[3. Class-Member Diagram 5](#_Toc159784562)

[4. Complexities 10](#_Toc159784563)

[4.1. MySQL queries 10](#_Toc159784564)

[4.2. OOPL (Java) 11](#_Toc159784565)

[4.3. Try-Catch Block 12](#_Toc159784566)

[4.4. Switchcase 13](#_Toc159784567)

[4.5. TableCellRenderers 14](#_Toc159784568)

[4.6. Class Creation 15](#_Toc159784569)

[4.7. Sending E-mails 16](#_Toc159784570)

[4.8. Logging 17](#_Toc159784571)

[4.9. Search (using Regex API) 18](#_Toc159784572)

[4.10. Error Handling 19](#_Toc159784573)

[4.11. Session Storage 20](#_Toc159784574)

[4.12. Date & Time Capabilities 21](#_Toc159784575)

**Table of Figure Index:**

|  |  |  |
| --- | --- | --- |
| No | Figure Index | Page No |
| 1 | Creating the database "resourceinventorymanagement" in MySQL Workbench | 3 |
| 2 | Database code and table for the Streamliner Application | 4 |
| 3 | Class Member Diagram for the Streamliner Application | 5 |
| 4 | Establishing connection and running queries to MySQL | 10 |
| 5 | Use of If/Else statements and While loops | 11 |
| 6 | Use of Try-Catch Block for uncertain outcomes | 12 |
| 7 | Use of Switchcase for unlockdate functionality | 13 |
| 8 | Code and Outcome of using TableCellRenderers | 14 |
| 9 | Creating the Emailsend class and running the class | 15 |
| 10 | Result of using Emailsend class | 16 |
| 11 | Establishing Logging class and the outcome of running the class. | 17 |
| 12 | Searching feature using Regex API | 18 |
| 13 | Code and Outcome of Error Handling | 19 |
| 14 | Establishing and using Session Storage in the application | 20 |
| 15 | Demonstration of Jcalendar functionality and outcome | 21 |
| 16 | Code and Database outcome of current date & time | 22 |

# MySQL Creation (Physical Schema)

* 1. Database Creation on MySQL Workbench

A screenshot of a computer

Description automatically generated

Figure : Creating the database "resourceinventorymanagement" in MySQL Workbench

* 1. Database Table creation on MySQL Workbench

|  |  |
| --- | --- |
| MySQL Database Coding | Database Table |
| A screen shot of a computer code  Description automatically generated | A screenshot of a password  Description automatically generated |
| A screenshot of a computer  Description automatically generated | A screenshot of a graph  Description automatically generated |
| A screen shot of a computer  Description automatically generated | A graph of a number of gray bars  Description automatically generated with medium confidence |
| A screen shot of a computer code  Description automatically generated | A screenshot of a graph  Description automatically generated |

Figure : Database code and table for the Streamliner Application

# List of Complexities

* 1. MySQL Database
  2. Try-Catch Clause
  3. Switchcase
  4. TableCellRenderers
  5. Class Creation
  6. Sending E-mails
  7. Logging
  8. Search (using Regex API)
  9. Error Handling
  10. Session Storage
  11. Date/Time Capabilities

# Class-Member Diagram

Figure : Class Member Diagram for the Streamliner Application

|  |  |
| --- | --- |
| Class Relationship Diagram | |
|  |  |
|  |  |
| A screenshot of a computer program  Description automatically generated |  |
|  | A screenshot of a computer program  Description automatically generated |
|  |  |
| A screenshot of a computer  Description automatically generated |  |
|  |  |

# Complexities

## MySQL queries

|  |  |
| --- | --- |
| A screen shot of a computer  Description automatically generated | Using java.sql.Connection and java.sql.DriverManager imports and the JDBC driver mysql-connector-j-8.1.0.jar, the code connects to the “resourceinventorymanagement” database. By establishing connection, SQL queries can be read and executed through the JFrame forms. |
| A screen shot of a computer program  Description automatically generated | To retrieve data from a specified database table, the imports: java.sql.PreparedStatement (pstatement) and java.sql.ResultSet (rset) are utilised to execute the query and retrieve the results according to the query respectively. Then the java.sql.Statement library (statement) is used to execute update, add, and delete queries. |

Figure : Establishing connection and running queries to MySQL

## OOPL (Java)

|  |  |  |
| --- | --- | --- |
|  | | The final product is developed fully through Java, where “if” and “else” statements, “while” loops, arrays, and “break”/”continue” statements are utilised throughout the application. This allows us to create conditions on locking the application as shown below. |
| Result | | |
| A screenshot of a computer  Description automatically generated |  | |

Figure : Use of If/Else statements and While loops

## Try-Catch Block

|  |  |
| --- | --- |
|  | The Try-Catch clause helps to establish the driver and MySQL connection, and then execute the following code afterwards. In the event that the JDBC Driver is not found or it has failed to connect to the database, the catch block will be utilised to establish the respective error in  an alert as shown below. This helps address the problem to the administrator. |
| Result | |
|  | |

Figure : Use of Try-Catch Block for uncertain outcomes

## Switchcase

|  |  |
| --- | --- |
|  | The switch case was utilised to determine the unlock duration of the application after 5 attempts. In the worst-case, where this scenario occurs on the last day of a month, the switch case was employed. The switch condition was employed as it is more efficient than an if/else statement as there are multiple months with a different number of days for each month. Hence it helps establish the unlocktime in worst-case scenarios. |

Figure : Use of Switchcase for unlockdate functionality

## TableCellRenderers

|  |  |
| --- | --- |
| A screen shot of a computer program  Description automatically generated | The javax.swing.table.DefaultTableCellRenderer library helps return the component used for each cell and alter the respective component. The “@override” helps override the method of the parent class (JTable).  By using an if/else statement, the products in the product table which have a quantity lower than 11 are set to red to alert the product is close to being out of stock. |
| Result | |
| A screenshot of a product list  Description automatically generated | |

Figure : Code and Outcome of using TableCellRenderers

## Class Creation

|  |  |
| --- | --- |
| A screen shot of a computer program  Description automatically generated | A class is created for this process in order to have organised code. Then, by calling the class name and void provided in the JFrame form, values obtained from the JFrame form can be passed to the class to be executed. |
|  |

Figure : Creating the Emailsend class and running the class

## Sending E-mails

Figure : Result of using Emailsend class

|  |  |
| --- | --- |
| Result | |
|  | With the code displayed in **4.6.**, using the javax.mail.jar (Javamail) API and its respective imports, the SMTP is established by providing the corresponding host and ports for the e-mail service used (Outlook) as well as the e-mail address and password for the designated e-mail used for the product. This was used for the scenario where a staff member forgot their password. Then, the “useremail” and “pw” retrieved from the database which is assigned to the username is utilised to send the email to the individual about their account password. |
| A close up of a screen  Description automatically generated |

## Logging

|  |  |  |
| --- | --- | --- |
|  | | The java.util.logging library is utilised to output application logs onto a file using a FileHandler, where any logs are appended to the file after it is created. Any changes in the database or database tables are logged by calling the “Streamlinerlog” class, where each is configured to log the time, the severity of the log, the AdminID of who did the action and the method (how it was achieved). |
| A screen shot of a computer  Description automatically generated | |
| Result | | |
|  | A screenshot of a computer  Description automatically generated | |

Figure : Establishing Logging class and the outcome of running the class.

## Search (using Regex API)

|  |  |  |
| --- | --- | --- |
| A screen shot of a computer  Description automatically generated | | Utilising the javax.swing.table.TableRowSorter library for the DefaultTableModel, it filters using regexFilter and provides any matches in the table with the text provided in the search bar. Any other rows are filtered out in the JTable (tblModel) when the search contains text. |
| Result | | |
| A screenshot of a computer  Description automatically generated | A screenshot of a computer  Description automatically generated | |

Figure : Searching feature using Regex API

## Error Handling

|  |
| --- |
|  |
| Using the try-catch clause before, a pop-up alert is sent towards the user/staff regarding the error using the catch block and JOptionPane. Therefore the staff can interpret the error message and correspondingly ask the developers on troubleshooting this error. |
| Result |
| A screenshot of a computer error message  Description automatically generated |

Figure : Code and Outcome of Error Handling

## Session Storage

|  |  |
| --- | --- |
| A computer screen shot of a program  Description automatically generated | As soon as a user is logged into the application, their userID is acquired from the database and shared with all tabs of the product. This allows for logging to be done more efficiently, rather than acquiring the userID from the database every time an action has been performed by the user. |
| A screen shot of a computer  Description automatically generated |
| A screen shot of a computer program  Description automatically generated |
| Result | |
| A screenshot of a computer  Description automatically generated | |

Figure : Establishing and using Session Storage in the application

## Date & Time Capabilities

|  |  |
| --- | --- |
| A screenshot of a calendar  Description automatically generated | Using the jcalendar-1.4.jar driver, a calendar component can be added into JFrame to determine the manufacturing date for a product. Then by converting the data into a string and respectively making substrings and concatenating them (to get an appropriate date format), the manufacturing date can be added onto the database table for the product. |
| A screen shot of a computer code  Description automatically generated |
| Result | |
| A screenshot of a computer  Description automatically generated | |

Figure : Demonstration of Jcalendar functionality and outcome

|  |  |
| --- | --- |
| A computer screen shot of a code  Description automatically generated | Then, using the java.text.SimpleDateFormat and java.util.Date libraries, the time and date of each transaction can be recorded into the database to provide assurance on the validity of each transaction. Moreover, it can be used to identify a transaction made and alter that transaction accordingly, making it more easier for the user. |
| A screenshot of a computer  Description automatically generated |
| Result | |
| A screenshot of a graph  Description automatically generated | |

Figure : Code and Database outcome of current date & time

**Word Count: 884**

**Bibliography:**

Javatpoint. “How to Get Current Date and Time in Java.” *Javatpoint*,

https://www.javatpoint.com/java-get-current-date. Accessed 15 September 2023.

NetBeans. *Logging in NetBeans*, https://bits.netbeans.org/dev/javadoc/org-openide-

util/org/openide/util/doc-files/logging.html. Accessed 17 September 2023.

Pankaj et al. “JavaMail Example - Send Mail in Java using SMTP.” *DigitalOcean*, 3 August

2022, https://www.digitalocean.com/community/tutorials/javamail-example-send-mail-in-java-smtp. Accessed 17 September 2023.

Raja et al. “How to implement the search functionality of a JTable in Java?” *Tutorialspoint*,

10 February 2020, https://www.tutorialspoint.com/how-to-implement-the-search-functionality-of-a-jtable-in-java. Accessed 19 September 2023.

Salas, Charles, and Step Guide. “Java: Modify the color of a specific row in a JTable.” *Copy*

*Programming*, 7 April 2023, https://copyprogramming.com/howto/change-color-of-particular-row-in-jtable. Accessed 22 September 2023.

W3Schools. “Java Arrays.” *W3Schools*, https://www.w3schools.com/java/java\_arrays.asp.

Accessed 20 September 2023.