SAMPLE CODE

1. Trigger for Out of Stock:

```
DELIMITER ;;
CREATE TRIGGER `outofstocktrigger`
BEFORE UPDATE ON `medicine`
FOR EACH ROW
begin
   if NEW.quantity <= 0
   then
     insert into out_of_stock
     set id = NEW.id;
   elseif NEW.quantity > 0
   then
     delete from out_of_stock where out_of_stock.id = NEW.id;
   end if;
end;
DELIMITER ;
```

2. **Add selected medicine with given quantity to cart:**

```
private void btn_addtocartActionPerformed(java.awt.event.ActionEvent
evt) {
            String quantity_validate = txt_quantity.getText();

            int row = table_medicines.getSelectedRow();
            String table_click =
    table_medicines.getModel().getValueAt(row, 0).toString(); //
    medicine id is taken into table_click variable

            try {
                String sql = "select * from medicine where id =" +
    table_click + ";";
                pst = conn.prepareStatement(sql);
                rs = pst.executeQuery();

                if (rs.next()) {
                    price = rs.getString("price");
                    quantity = rs.getString("quantity");
                }
            } catch (Exception e) {
                JOptionPane.showMessageDialog(null, e);
            } finally {
                try {
                    rs.close();
                    pst.close();
                } catch (Exception e) {
                    JOptionPane.showMessageDialog(null, e);
                }
            }
            if (quantity_validate.equals("")){
                JOptionPane.showMessageDialog(null, "Enter
    quantity");
            }
            else if (Integer.parseInt(quantity_validate) >
    Integer.parseInt(quantity)) {
```

```
            JOptionPane.showMessageDialog(null, "Not enough stock
available!");
         }

         else {
            try {
                String sql = "insert into cart (id, price,
quantity)values(?,?,?);";
                pst = conn.prepareStatement(sql);
                String currentSelectedID = medID;
                pst.setString(1, medID);
                pst.setString(2, price);
                pst.setString(3, txt_quantity.getText());


                pst.execute();
                UpdateCart();

           updateQuantityAfterAddingToCart(currentSelectedID);
                btn_bill.setEnabled(true);
                btn_clear.setEnabled(true);
                txt_quantity.setText("");
                noincart++;
             } catch (Exception e) {
                JOptionPane.showMessageDialog(null, e);
             } finally { // gurantees this section will be
executed, even if error is encountered
                try {
                    rs.close();
                    pst.close();
                } catch (Exception e) {
                    JOptionPane.showMessageDialog(null, e);
                }
             }
        }
    }
```

3. **Filtering Report using given dates on clicking 'Filter' button:**

```
private void btn_filterActionPerformed(java.awt.event.ActionEvent
evt) {
        java.util.Date from_date = date_from.getDate();
        java.util.Date to_date = date_to.getDate();

        java.util.Date date=new java.util.Date();
        java.sql.Date sqlDate=new java.sql.Date(date.getTime());

        if (from_date == null) {
            JOptionPane.showMessageDialog(null, "Enter 'From'
Date");
        } else if (from_date.compareTo(sqlDate) >= 0) {
            JOptionPane.showMessageDialog(null, "Enter past date
in 'from' date chooser");
        }
        else if (to_date == null) {
            JOptionPane.showMessageDialog(null, "Enter 'to'
Date");
        } else {
            try {
                SimpleDateFormat sdf = new
SimpleDateFormat("yyyy-MM-dd");
                String str_from_date = sdf.format(from_date);
                String str_to_date = sdf.format(to_date);
                String sql = "select * from bill where date >= ?
and date <= ?;";
                pst = conn.prepareStatement(sql);
                pst.setString(1, str_from_date);
                pst.setString(2, str_to_date);
                rs = pst.executeQuery();

table_report.setModel(DbUtils.resultSetToTableModel(rs));

            } catch (Exception e) {
                JOptionPane.showMessageDialog(null, e);
            } finally {
                try {
```

```
                    rs.close();
                    pst.close();
                } catch (Exception e) {
                    JOptionPane.showMessageDialog(null, e);
                }
            }
        }
    }
```

CONCLUSION

This **Pharmacy Store Management System** has been computed and tested successfully. It is user friendly and has various required options, which has been utilized by the user to perform the desired operations.

The software is developed using Java as front end and MySQL as backend in Windows environment. The goals that are achieved by the software are:

- Efficient management of records of medicines.
- Simplification of the operations.
- Less processing time and getting required information.
- User friendly.
- Portable and flexible for further enhancement.

REFERENCES

Books:

1. Database System Concepts 7th edition – Abraham Silberschatz
2. Java The Complete Reference 9th edition – Herbert Schildt

Links:

1. https://en.wikipedia.org/wiki/Java_(programming_language)
2. https://www.tutorialspoint.com/dbms/
3. https://netbeans.org/kb/docs/web/mysql
4. https://stackoverflow.com