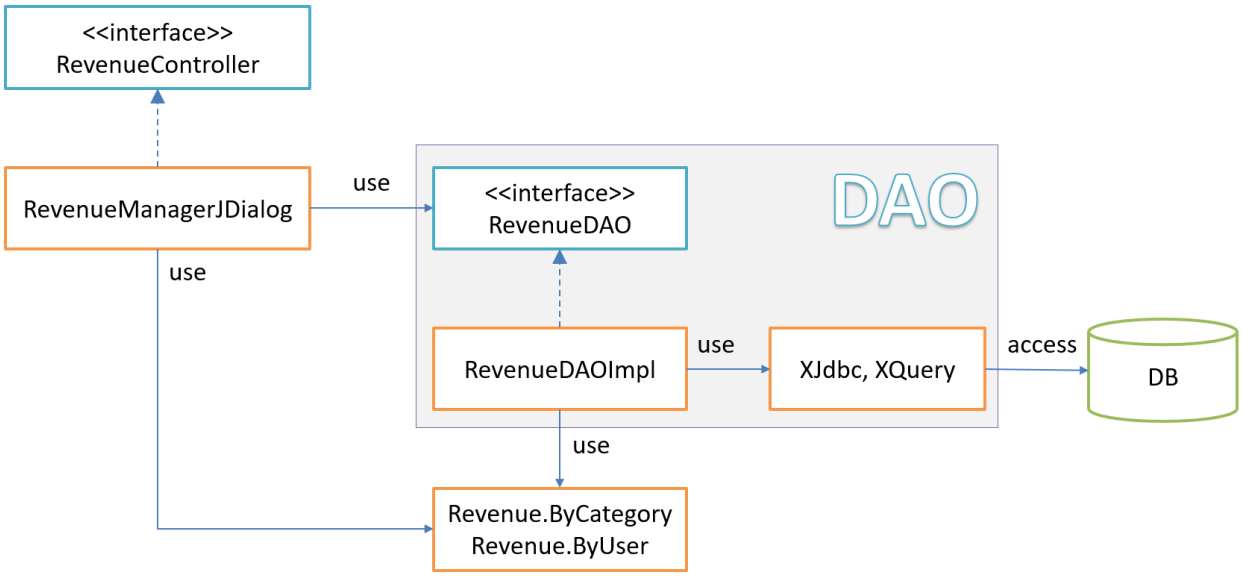
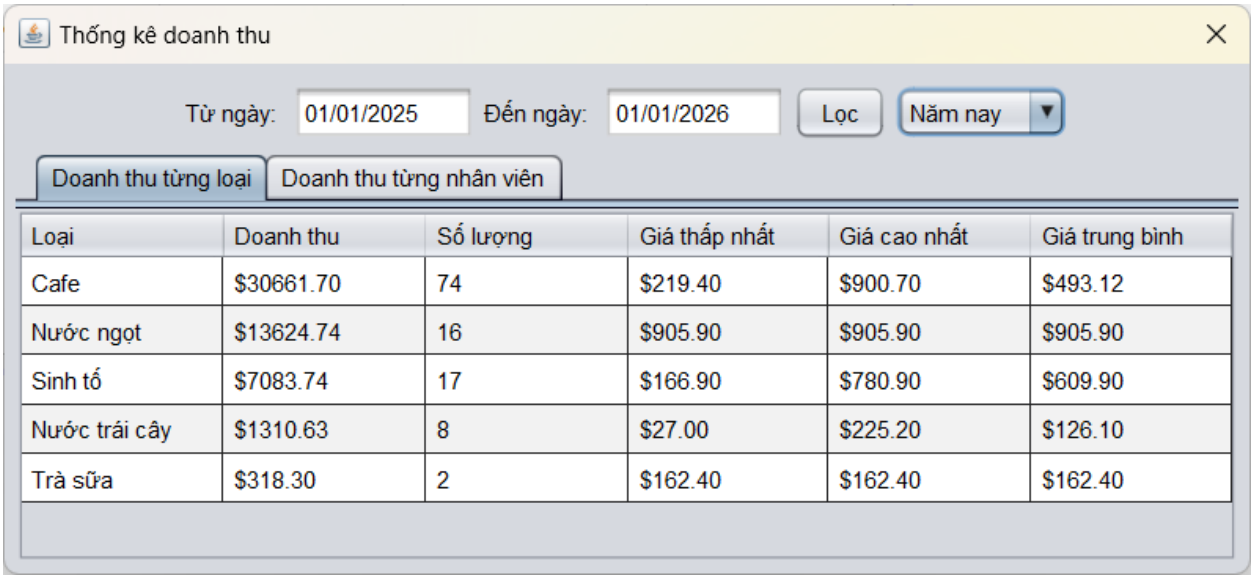
THỐNG KÊ DOANH THU BÁN HÀNG

Xây dựng chức năng thống kê doanh thu bán hàng từng loại và từng nhân viên theo thời gian.

SƠ ĐỒ TỔ CHỨC

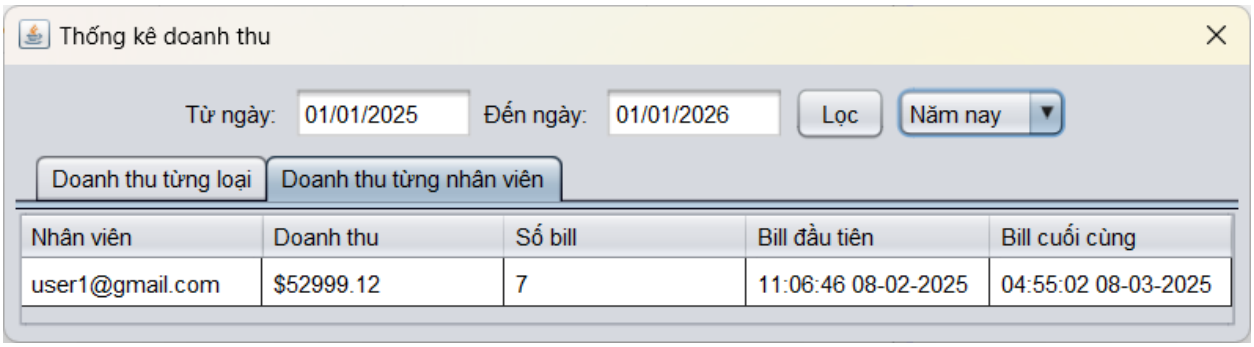


GIAO DIỆN



Loại	Doanh thu	Số lượng	Giá thấp nhất	Giá cao nhất	Giá trung bình
Cafe	$30661.70	74	$219.40	$900.70	$493.12
Nước ngọt	$13624.74	16	$905.90	$905.90	$905.90
Sinh tố	$7083.74	17	$166.90	$780.90	$609.90
Nước trái cây	$1310.63	8	$27.00	$225.20	$126.10
Trà sữa	$318.30	2	$162.40	$162.40	$162.40

Hình 1: Doanh thu từng loại theo thời gian



Hình 2: Doanh thu từng nhân viên theo thời gian

CÁC THÀNH PHẦN CẦN PHẢI XÂY DỰNG

- GUI: RevenueJDialog
- Entity:
  - Revenue.ByCategory
  - Revenue.ByUser
- DAO
  - RevenueDAO, RevenueDAOImpl
  - XJdbc/XQuery
- Controller: RevenueController
-

GUI: REVENUEJDAILOG

- Thiết kế giao diện cho RevenueJDialog như hình 1 và 2
- Tên các thành phần liên quan đến mã nguồn
  - Lọc:
    - Từ ngày (txtBegin)
    - Đến ngày (txtEnd)
    - Nút Lọc (btnFilter)
    - Khoảng thời gian sẵn có (cboTimeRanges)
  - Bảng chứa dữ liệu doanh thu
    - Doanh thu từng loại (tblByCategory)
    - Doanh thu từng nhân viên (tblByUser)

ENTITY

Cần tạo 2 thực thể có cấu trúc dữ liệu phù hợp để chứa dữ liệu hiển thị lên các bảng dữ liệu doanh thu. Để dễ quản lý, khai báo 2 lớp nội tuyến bên trong lớp Revenue:

```
public class Revenue {
   @AllArgsConstructor
   @NoArgsConstructor
   @Builder
   @Data
   public static class ByCategory {
```

```
        private String category; // Tên loại
        private double revenue; // Doanh thu
        private int quantity; // Số lượng đồ uống đã bán
        private double minPrice; // Giá bán cao nhất
        private double maxPrice; // Giá bán thấp nhất
        private double avgPrice; // Giá bán trung bình
    }
    @AllArgsConstructor
    @NoArgsConstructor
    @Builder
    @Data
    public static class ByUser {
        private String user; // Tên đăng nhập của nhân viên bán hàng
        private double revenue; // Doanh thu
        private int quantity; // Số lượng đơn hàng đã bán
        private Date firstTime; // Thời điểm bán đơn hàng đầu tiên
        private Date lastTime; // Thời điểm bán đơn hàng sau cùng
    }
}
```

DAO

Cần tạo RevenueDAO khai báo các phương thức truy vấn dữ liệu cần thiết cho báo cáo và lớp RevenueDAOImpl cài đặt mã nguồn cho các phương thức của RevenueDAO.

```
public interface RevenueDAO{
    /**
     * Truy vấn doanh thu từng loại theo khoảng thời gian
     *
     * @param begin thời gian bắt đầu
     * @param end thời gian kết thúc
     * @return kết quả truy vấn
     */
    List<Revenue.ByCategory> getByCategory(Date begin, Date end);
    /**
     * Truy vấn doanh thu từng nhân viên theo khoảng thời gian
     *
     * @param begin thời gian bắt đầu
     * @param end thời gian kết thúc
     * @return kết quả truy vấn
     */
    List<Revenue.ByUser> getByUser(Date begin, Date end);
}
```

```
public class RevenueDAOImpl implements RevenueDAO {
```

```
    @Override
    public List<ByCategory> getByCategory(Date begin, Date end) {
        String revenueByCategorySql
                = "SELECT category.Name AS Category, "
                + "   sum(detail.UnitPrice*detail.Quantity*(1-detail.Discount)) AS Revenue,"
                + "   sum(detail.Quantity) AS Quantity,"
                + "   min(detail.UnitPrice) AS MinPrice,"
                + "   max(detail.UnitPrice) AS MaxPrice,"
                + "   avg(detail.UnitPrice) AS AvgPrice "
                + "FROM SOF2042_BillDetails detail "
                + "   JOIN SOF2042_Drinks drink ON drink.Id=detail.DrinkId"
                + "   JOIN SOF2042_Categories category ON category.Id=drink.CategoryId"
                + "   JOIN SOF2042_Bills bill ON bill.Id=detail.BillId "
                + "WHERE bill.Status=1 "
                + "   AND bill.Checkout IS NOT NULL "
                + "   AND bill.Checkout BETWEEN ? AND ? "
                + "GROUP BY category.Name "
                + "ORDER BY Revenue DESC";
        return XQuery.getBeanList(ByCategory.class, revenueByCategorySql, begin, end);
    }
    @Override
    public List<ByUser> getByUser(Date begin, Date end) {
        String revenueByUserSql
                = "SELECT bill.Username AS [User], "
                + "   sum(detail.UnitPrice*detail.Quantity*(1-detail.Discount)) AS Revenue,"
                + "   count(DISTINCT detail.BillId) AS Quantity,"
                + "   min(bill.Checkin) AS FirstTime,"
                + "   max(bill.Checkin) AS LastTime "
                + "FROM SOF2042_BillDetails detail "
                + "   JOIN SOF2042_Bills bill ON bill.Id=detail.BillId "
                + "WHERE bill.Status=1 "
                + "   AND bill.Checkout IS NOT NULL "
                + "   AND bill.Checkout BETWEEN ? AND ? "
                + "GROUP BY bill.Username "
                + "ORDER BY Revenue DESC";
        return XQuery.getBeanList(ByUser.class, revenueByUserSql, begin, end);
    }
}
```

REVENUECONTROLLER

- Khai báo các phương thức xử lý tương tác lên các thành phần giao diện để kết nối với các event handler trên giao diện

```
public interface RevenueController {
    void open(); // hiển thị doanh thu từng loại trong ngày
```

```
    void selectTimeRange(); // hiển thị doanh thu theo khoảng thời gian được chọn
    void fillRevenue(); // hiển thị doanh thu
}
```

CÀI ĐẶT MÃ NGUỒN CHO REVENUECONTROLLER

CẤU TRÚC TỔ CHỨC

```
public class RevenueManagerJDialog extends JDialog implements RevenueController {
        ....
        RevenueDAO dao = new RevenueDAOImpl();

        @Override
        public void open() {...}

        @Override
        public void selectTimeRange() {...}

        @Override
        public void fillRevenue() {...}
}
```

CÀI ĐẶT MÃ HOÀN THIỆN

```
@Override
public void open() {
   this.setLocationRelativeTo(null);
   this.selectTimeRange();
}

@Override
public void selectTimeRange() {
   TimeRange range = TimeRange.today();
   switch (cboTimeRanges.getSelectedIndex()) {
      case 0 -> range = TimeRange.today();
      case 1 -> range = TimeRange.thisWeek();
      case 2 -> range = TimeRange.thisMonth();
      case 3 -> range = TimeRange.thisQuarter();
      case 4 -> range = TimeRange.thisYear();
   }
   txtBegin.setText(XDate.format(range.getBegin(), "MM/dd/yyyy"));
   txtEnd.setText(XDate.format(range.getEnd(), "MM/dd/yyyy"));

   this.fillRevenue();
}

@Override
public void fillRevenue() {
```

```
    Date begin = XDate.parse(txtBegin.getText(), "MM/dd/yyyy");
    Date end = XDate.parse(txtEnd.getText(), "MM/dd/yyyy");
    switch(tabs.getSelectedIndex()){
        case 0 -> this.fillRevenueByCategory(begin, end);
        case 1 -> this.fillRevenueByUser(begin, end);
    }
}

private void fillRevenueByCategory(Date begin, Date end) {
    List<Revenue.ByCategory> items = dao.getByCategory(begin, end);

    DefaultTableModel model = (DefaultTableModel) tblByCategory.getModel();
    model.setRowCount(0);
    items.forEach(item -> {
        Object[] row = {
            item.getCategory(),
            String.format("$%.2f", item.getRevenue()),
            item.getQuantity(),
            String.format("$%.2f", item.getMinPrice()),
            String.format("$%.2f", item.getMaxPrice()),
            String.format("$%.2f", item.getAvgPrice())
        };
        model.addRow(row);
    });
}

private void fillRevenueByUser(Date begin, Date end) {
    List<Revenue.ByUser> items = dao.getByUser(begin, end);

    DefaultTableModel model = (DefaultTableModel) tblByUser.getModel();
    model.setRowCount(0);
    items.forEach(item -> {
        Object[] row = {
            item.getUser(),
            String.format("$%.2f", item.getRevenue()),
            item.getQuantity(),
            XDate.format(item.getFirstTime(), "hh:mm:ss dd-MM-yyyy"),
            XDate.format(item.getLastTime(), "hh:mm:ss dd-MM-yyyy")
        };
        model.addRow(row);
    });
}
```

Gắn kết các phương thức của RevenueController vào các event handler

```
private void formWindowOpened(java.awt.event.WindowEvent evt) {
    // TODO add your handling code here:
    this.open();
}

private void tabsStateChanged(javax.swing.event.ChangeEvent evt) {
    // TODO add your handling code here:
    this.fillRevenue();
}

private void btnFilterActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    this.fillRevenue();
}

private void cboTimeRangesActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    this.selectTimeRange();
}
```