



**МОНГОЛ УЛСЫН ШИНЖЛЭХ УХААН
ТЕХНОЛОГИЙН ИХ СУРГУУЛЬ**

MONGOLIAN UNIVERSITY OF SCIENCE AND TECHNOLOGY

АЖЛЫН ТАЙЛАН

“F.CSM360 Програмчлалын дадлага” хичээлийн 6-р өдрийн тайлан

Хичээл заасан багш: Ч.Эрдэнэбат

Оюутан : Н.Төрболд B232270800

Э. Анар B210910840

Б.Дөлгөөн B242270136

Улаанбаатар, 2025

Төслийн зорилго:

Энэхүү модулийн зорилго нь бизнес эрхлэгчдэд хэрэглэгчийн санал хүсэлтийг үр дүнтэй удирдах, хариу өгөх боломжийг олгох юм. Хэрэглэгчийн хариултыг зохион байгуулж, хянах замаар хэрэглэгчийн үйлчилгээ, бүтээгдэхүүний чанарыг сайжруулахад туслах зорилготой юм. Энэхүү систем нь компанид хэрэглэгчийн дэлгэрэнгүй мэдээлэл, санал хүсэлтийг бүртгэж, хянаж үзэх, шаардлагатай бол оруулгыг устгах боломжийг олгодог.

Хэрэглэгчийн шаардлага:

- Үйлчлүүлэгчийн нэр, утасны дугаар, тэдний санал хүсэлтийг оруулах энгийн интерфейс.
- Одоо байгаа бүх хэрэглэгчийн санал хүсэлтийг харах арга.
- Сонгосон санал хүсэлтийг шалгуурт үндэслэн устгах сонголт (жишээ нь, утасны дугаар эсвэл санал хүсэлтийн ID).
- Нэмэлт: тодорхой санал хүсэлтэд хялбар хандах үндсэн шүүлтүүр эсвэл хайлтын функц.

Гол онцлогууд:

- Санал хүсэлт нэмэх: Хэрэглэгчийн нэр, утасны дугаар, бичсэн санал хүсэлтийг оруулж, хадгалах боломжтой.
- Санал хүсэлтийг харах: Системд хадгалагдсан бүх хэрэглэгчийн санал хүсэлтийн жагсаалтыг харуулах.
- Санал хүсэлтийг устгах: Шаардлагагүй болсон эсвэл шийдвэрлэсний дараа тодорхой санал хүсэлтийг устгана.
- Жижиг, дунд бизнест хэрэглэхэд хялбар, хөнгөн жинтэй загвар.

Цаашид сайжруулах боломжууд:

1. Хэрэглэгчийн санал хүсэлтийн ангилал

Санал хүсэлтүүдийг тодорхой ангиллаар (жишээ нь: үйлчилгээ, бүтээгдэхүүн, хүргэлт, техникийн асуудал гэх мэт) шүүж, ангилж бүртгэх боломжийг нэмж өгснөөр мэдээлэл илүү бүтэцтэй болж, дүн шинжилгээ хийхэд хялбар болно.

2. Хариу өгөх, шийдвэрлэсэн төлөв байдал

Санал хүсэлт бүрт “Хариу өгсөн эсэх”, “Шийдвэрлэсэн эсэх” гэсэн статус нэмснээр хэрэглэгчидтэй харилцах процессыг илүү тодорхой болгоно. Үүнд:

- Статус: "Хүлээгдэж буй", "Шийдвэрлэсэн", "Хариу өгсөн"

- Хариу бичих боломж: Системээр дамжуулан хэрэглэгчид хариу мессеж илгээх эсвэл дотоод тэмдэглэл үлдээх

3. Түүх болон лог бүртгэл

Хэрэглэгчийн санал хүсэлттэй холбоотой өөрчлөлт бүр (нэмсэн, зассан, устгасан) системд бүртгэгдэх лог механизмыг нэмснээр хяналт, аудит хийх боломж нэмэгдэнэ.

4. Санал хүсэлтийн тайлан, график үзүүлэлт

Санал хүсэлтийн тоо, төрөл, хариу өгсөн хугацаа гэх мэт мэдээлэлд суурилсан тайлан болон хялбар график харагдац (dashboard) үүсгэж, хэрэглэгчийн үйлчилгээний түвшинг үнэлэхэд тусална.

5. Олон хэрэглэгчийн эрхийн зохицуулалт (User roles)

Админ, ажилтан зэрэг ялгаатай хэрэглэгчийн эрхтэй болгосноор мэдээлэлд нэвтрэх, устгах, засварлах үйлдлийг хязгаарлах боломжтой болно.

6. Мэдэгдэл илгээх систем (Notification)

Шинэ санал ирэхэд хариуцсан ажилтанд и-мэйл, push notification илгээх боломж бүхий мэдэгдлийн системийг нэмэх.

7. Интерфейс / Responsive дизайн

Жижиг, дунд бизнесүүд мэдээлэл харах хэрэгцээ гардаг тул системийн хөнгөн, тохирсон хувилбарыг хөгжүүлэх.

Гол функц тайлбар

1. Абстракт класс ба түүний өргөтгөл

abstract class Subscription

- Үндсэн өгөгдөл хадгалах зориулалттай абстракт (тодорхойгүй) класс.
- toRow() функц – JTable-д харуулах зориулалттай мөр (row) хэлбэрээр Subscription объектын өгөгдлийг буцаана.
- getType() – ямар төрлийн subscription болохыг тодорхойлох abstract функц.

ProductSubscription, ServiceSubscription

- Subscription классыг удамшуулсан хоёр төрөл.

- Төрлөө getType() функцээр "Product", "Service" гэж буцаана.
-

2. Менежмент класс

class SubscriptionManager

Subscription объектуудын жагсаалтыг удирдах функцууд:

- addSubscription(...) – шинэ захиалга нэмнэ.
 - editSubscription(...) – байгаа захиалгыг өөрчилнө.
 - removeSubscription(String id) – ID-гаар устгана.
 - filter(String keyword) – хайлтын түлхүүр үгээр шүүлт хийнэ.
 - generateNewId() – автоматаар шинэ ID үүсгэнэ.
 - findById(String id) – ID-гаар Subscription объект хайна.
-

3. График интерфэйс – GUI

SubscriptionManagerGUI extends JFrame

Сүүлийн хэрэглэгчид харагдах үндсэн GUI цонх.

Компонентууд үүсгэх:

- JTable – бүх захиалгын мэдээллийг хүснэгтэд харуулна.
- JTextField – хайлт хийх текст талбар.
- JButton – нэмэх, засах, устгах товчлуурууд.

Функцууд:

- refreshTable()
→ Хүснэгтийн өгөгдлийг шинэчилнэ. Хайлтын утгатай бол шүүж харуулна.
- showForm(Subscription existing)
→ Захиалга нэмэх эсвэл засах формыг JOptionPane ашиглан харуулна.

→ JTextField, JComboBox зэрэг компонент ашиглан өгөгдөл авна.

ActionListener ашигласан үйлдлүүд

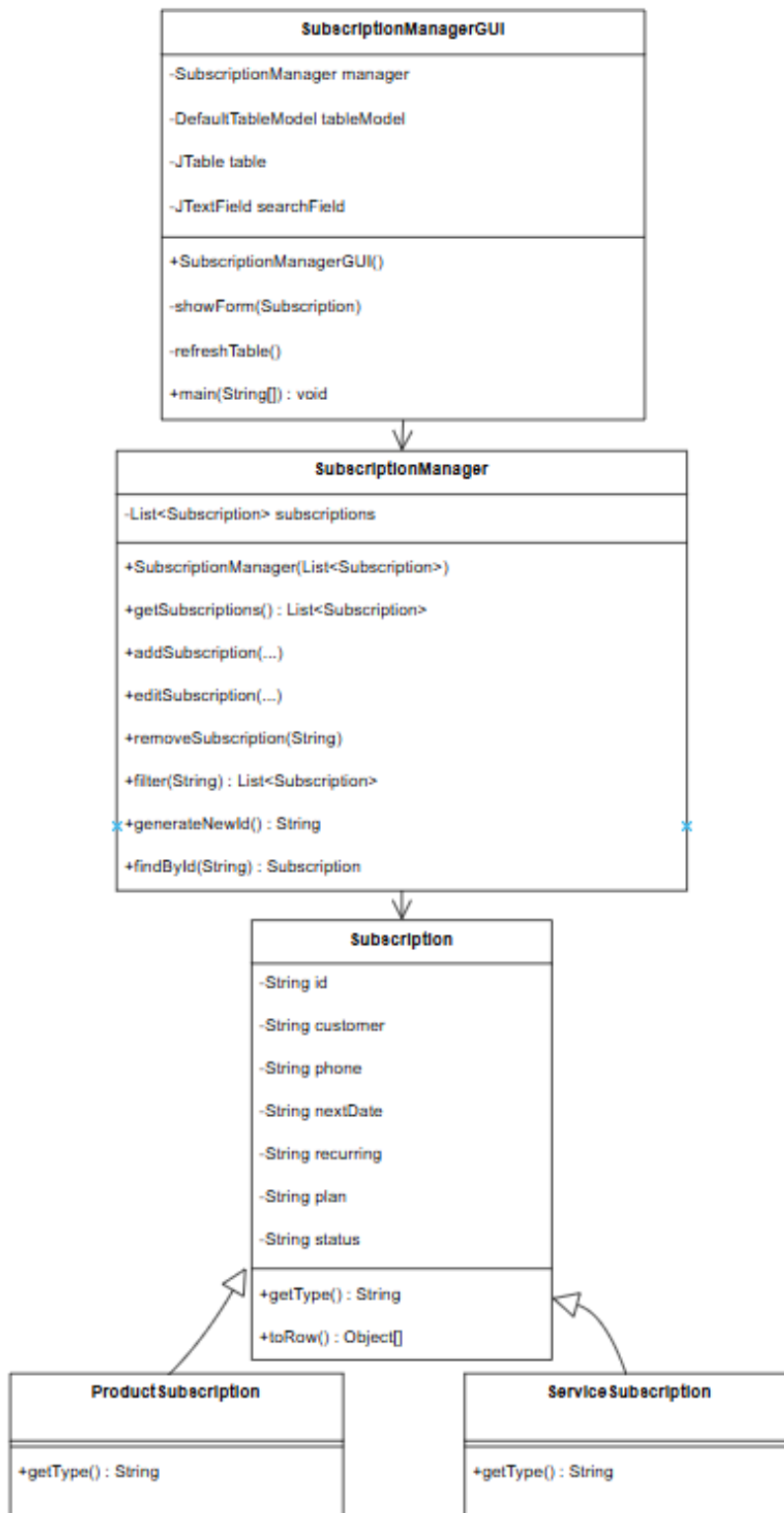
- editBtn.addActionListener()
→ Сонгосон мөрийн ID-г олж, editSubscription дуудан шинэчлэнэ.
- removeBtn.addActionListener()
→ Сонгосон мөрийн ID-г ашиглан removeSubscription хийж хүснэгтээс хасна.
- searchField.getDocument().addDocumentListener(...)
→ Текст өөрчлөгдөх бүрт refreshTable() автоматаар ажиллаж шүүж харуулна.

4 . Алдаа шалгалт

төслийн үеэр бид системийн найдвартай ажиллагааг хангах, хэрэглэгчийн оруулсан мэдээллийн алдааг бууруулах зорилгоор алдааг олж засварлах ажлуудыг амжилттай хийлээ.

- **Оруулах өгөгдлийн шалгалт:** Хэрэглэгчийн нэр, утасны дугаар, дахин төлбөрийн хэмжээ болон огнооны форматыг хязгаарлах, зөвшөөрөгдөөгүй утга оруулах үед мэдээллийг зөв оруулахыг сануулсан алдааны мэдэгдэл гаргадаг болсон.
- **Алдаатай огнооны форматтой холбоотой асуудлыг шийдвэрлэв:** Огнооны форматыг зөв оруулахгүй бол систем алдаа зааж, хэрэглэгчдэд огноог зөв форматаар оруулах талаар тодорхой зааварчилгаа өгдөг болсон.
- **Алдааны мессежүүдийг илүү ойлгомжтой болгов:** Алдааны үед хэрэглэгчдэд үзүүлэх мэдээлэлд тодорхой, хялбар ойлгогдох мессежүүдийг ашиглан хэрэглэгчийн туршлагыг сайжруулсан.
- **Хоёрдогч алдааг урьдчилан сэргийлэх:** Буруу өгөгдөлтэй захиалгыг бүртгэхгүй байх, системийн бусад хэсгүүдэд дамжин алдаа үүсгэх боломжийг багасгасан.
- **Жишээ:** Захиалгын огноо “ММ-dd-ууу” бус өөр форматтай орж ирвэл `DateTimeParseException` гарч, хэрэглэгчид анхааруулга үзүүлж, дахин оруулахыг шаарддаг болсон.

Эдгээр засварууд нь системийн найдвартай байдлыг дээшлүүлж, хэрэглэгчийн оруулах өгөгдлийн алдааг багасгах гол хүчин зүйл болсон.



Код:

```
import javax.swing.*.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*.*;
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.time.format.DateTimeParseException;
import java.util.ArrayList;
import java.util.List;

abstract class Subscription {
    String id, customer, phone, nextDate;
    LocalDate nextDateObj;
    String recurring, plan, status;

    public Subscription(String id, String customer, String phone, LocalDate nextDateObj, String
recurring, String plan, String status) {
        this.id = id;
        this.customer = customer;
        this.phone = phone;
        this.nextDateObj = nextDateObj;
        this.nextDate = nextDateObj.format(DateTimeFormatter.ofPattern("MM-dd-yyyy"));
        this.recurring = recurring;
        this.plan = plan;
        this.status = status;
    }

    public abstract String getType();

    public Object[] toRow() {
        return new Object[]{id, customer, phone, nextDate, recurring, plan, status, getType()};
    }
}

class ProductSubscription extends Subscription {
    public ProductSubscription(String id, String customer, String phone, LocalDate nextDateObj,
String recurring, String plan, String status) {
        super(id, customer, phone, nextDateObj, recurring, plan, status);
    }

    @Override
    public String getType() {
        return "Бүтээгдэхүүн";
    }
}
```

```

class ServiceSubscription extends Subscription {
    public ServiceSubscription(String id, String customer, String phone, LocalDate nextDateObj,
String recurring, String plan, String status) {
        super(id, customer, phone, nextDateObj, recurring, plan, status);
    }

    @Override
    public String getType() {
        return "Үйлчилгээ";
    }
}

```

```

class SubscriptionManager {
    private List<Subscription> subscriptions = new ArrayList<>();

    public SubscriptionManager(List<Subscription> initialData) {
        subscriptions.addAll(initialData);
    }

    public List<Subscription> getSubscriptions() {
        return subscriptions;
    }

    public void addSubscription(String customer, String phone, LocalDate nextDateObj, String
recurring, String plan, String status, String type) {
        String id = generateNewId();
        Subscription sub = type.equalsIgnoreCase("Бүтээгдэхүүн") ?
            new ProductSubscription(id, customer, phone, nextDateObj, recurring, plan, status) :
            new ServiceSubscription(id, customer, phone, nextDateObj, recurring, plan, status);
        subscriptions.add(sub);
    }

    public String generateNewId() {
        return "S" + String.format("%04d", subscriptions.size() + 1);
    }

    public Subscription findById(String id) {
        return subscriptions.stream().filter(s -> s.id.equals(id)).findFirst().orElse(null);
    }

    public void editSubscription(String id, String customer, String phone, LocalDate nextDateObj,
String recurring, String plan, String status, String type) {
        Subscription old = findById(id);
        if (old != null) {
            subscriptions.remove(old);
            Subscription updated = type.equalsIgnoreCase("Бүтээгдэхүүн") ?

```



```

        new ProductSubscription(id, customer, phone, nextDateObj, recurring, plan, status)
:
        new ServiceSubscription(id, customer, phone, nextDateObj, recurring, plan, status);
    subscriptions.add(updated);
    }
}

public void removeSubscription(String id) {
    subscriptions.removeIf(s -> s.id.equals(id));
}

public List<Subscription> filter(String keyword) {
    keyword = keyword.toLowerCase();
    List<Subscription> result = new ArrayList<>();
    for (Subscription s : subscriptions) {
        if (s.id.toLowerCase().contains(keyword) ||
            s.customer.toLowerCase().contains(keyword) ||
            s.phone.contains(keyword) ||
            s.plan.toLowerCase().contains(keyword) ||
            s.status.toLowerCase().contains(keyword)) {
            result.add(s);
        }
    }
    return result;
}
}

```

```

public class SubscriptionManagerGUI extends JFrame {
    private SubscriptionManager manager;
    private DefaultTableModel tableModel;
    private JTable table;
    private JTextField searchField;
    private final DateFormatter dateFormatter =
        DateFormatter.ofPattern("MM-dd-yyyy");

    public SubscriptionManagerGUI() {
        List<Subscription> initialSubs = List.of(
            new ProductSubscription("S0001", "Бат Энх", "99119911",
                LocalDate.parse("09-25-2025", dateFormatter), "$35.00", "Cap бүр", "Идэвхтэй"),
            new ServiceSubscription("S0002", "Эрхэс Бат", "99112233",
                LocalDate.parse("09-25-2025", dateFormatter), "$35.00", "Cap бүр", "Идэвхтэй")
        );
        manager = new SubscriptionManager(initialSubs);

        setTitle("Захиалгын Менежер");
        setSize(900, 400);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }
}

```

```

setLayout(new BorderLayout());

tableModel = new DefaultTableModel(new String[]{"Дугаар", "Хэрэглэгч", "Утас",
"Дараагийн огноо", "Дахин төлбөр", "Төлөвлөгөө", "Төлөв", "Төрөл"}, 0);
table = new JTable(tableModel);
add(new JScrollPane(table), BorderLayout.CENTER);

JPanel topPanel = new JPanel(new FlowLayout());
JButton addBtn = new JButton("Нэмэх");
JButton editBtn = new JButton("Засах");
JButton removeBtn = new JButton("Устгах");
searchField = new JTextField(15);

topPanel.add(addBtn);
topPanel.add(editBtn);
topPanel.add(removeBtn);
topPanel.add(new JLabel("Хайх:"));
topPanel.add(searchField);
add(topPanel, BorderLayout.NORTH);

addBtn.addActionListener(e -> showForm(null));
editBtn.addActionListener(e -> {
    int row = table.getSelectedRow();
    if (row != -1) {
        String id = (String) tableModel.getValueAt(row, 0);
        Subscription sub = manager.findById(id);
        showForm(sub);
    }
});
removeBtn.addActionListener(e -> {
    int row = table.getSelectedRow();
    if (row != -1) {
        String id = (String) tableModel.getValueAt(row, 0);
        manager.removeSubscription(id);
        refreshTable();
    }
});

searchField.getDocument().addDocumentListener(new
javax.swing.event.DocumentListener() {
    public void insertUpdate(javax.swing.event.DocumentEvent e) { refreshTable(); }
    public void removeUpdate(javax.swing.event.DocumentEvent e) { refreshTable(); }
    public void changedUpdate(javax.swing.event.DocumentEvent e) { refreshTable(); }
});

refreshTable();
}

```

```

private void showForm(Subscription existing) {
    JTextField customer = new JTextField(existing != null ? existing.customer : "");
    JTextField phone = new JTextField(existing != null ? existing.phone : "");
    JTextField nextDate = new JTextField(existing != null ? existing.nextDate : "");
    JTextField recurring = new JTextField(existing != null ? existing.recurring : "");
    JComboBox<String> planBox = new JComboBox<>(new String[]{"Сар бүр", "Жил бүр"});
    JComboBox<String> statusBox = new JComboBox<>(new String[]{"Идэвхтэй",
"Төлөвлөсөн", "Хаагдсан"});
    JComboBox<String> typeBox = new JComboBox<>(new String[]{"Бүтээгдэхүүн",
"Үйлчилгээ"});

    if (existing != null) {
        planBox.setSelectedItem(existing.plan);
        statusBox.setSelectedItem(existing.status);
        typeBox.setSelectedItem(existing.getType());
    }

    JPanel panel = new JPanel(new GridLayout(0, 1));
    panel.add(new JLabel("Хэрэглэгчийн нэр:")); panel.add(customer);
    panel.add(new JLabel("Утас (8 оронтой):")); panel.add(phone);
    panel.add(new JLabel("Дараагийн огноо (MM-dd-yyyy):")); panel.add(nextDate);
    panel.add(new JLabel("Дахин төлбөр (жишээ: $35.00):")); panel.add(recurring);
    panel.add(new JLabel("Төлөвлөгөө:")); panel.add(planBox);
    panel.add(new JLabel("Төлөв:")); panel.add(statusBox);
    panel.add(new JLabel("Төрөл:")); panel.add(typeBox);

    while (true) {
        int result = JOptionPane.showConfirmDialog(this, panel, existing == null ? "Захиалга
нэмэх" : "Захиалга засах", JOptionPane.OK_CANCEL_OPTION);
        if (result != JOptionPane.OK_OPTION) {
            break;
        }
        try {
            String custVal = customer.getText().trim();
            String phoneVal = phone.getText().trim();
            String nextDateVal = nextDate.getText().trim();
            String recurringVal = recurring.getText().trim();

            if (custVal.isEmpty() || custVal.length() < 2) {
                throw new IllegalArgumentException("Хэрэглэгчийн нэр дутуу байна.");
            }
            if (!phoneVal.matches("\\d{8}")) {
                throw new IllegalArgumentException("Утасны дугаарыг зөв оруулна уу (8
оронтой).");
            }
            if (!recurringVal.matches("\\d+(\\.\\d{2})?")) {

```

```

        throw new IllegalArgumentException("Дахин төлбөрийг зөв оруулна уу (жишээ:
35.00).");
    }
    double amount = Double.parseDouble(recurringVal.replace("$", ""));
    if (amount < 0.01) {
        throw new IllegalArgumentException("Дахин төлбөрийн дүн $0.01-аас бага байж
болохгүй.");
    }

    LocalDate nextDateObj = LocalDate.parse(nextDateVal, dateFormatter);

    if (existing == null) {
        manager.addSubscription(custVal, phoneVal, nextDateObj, recurringVal,
            planBox.getSelectedItem().toString(), statusBox.getSelectedItem().toString(),
typeBox.getSelectedItem().toString());
    } else {
        manager.editSubscription(existing.id, custVal, phoneVal, nextDateObj, recurringVal,
            planBox.getSelectedItem().toString(), statusBox.getSelectedItem().toString(),
typeBox.getSelectedItem().toString());
    }
    refreshTable();
    break;
} catch (IllegalArgumentException ex) {
    JOptionPane.showMessageDialog(this, ex.getMessage(), "Алдаа",
JOptionPane.ERROR_MESSAGE);
} catch (DateTimeParseException ex) {
    JOptionPane.showMessageDialog(this, "Дараагийн огноог зөв оруулна уу (жишээ:
09-25-2025).", "Алдаа", JOptionPane.ERROR_MESSAGE);
} catch (Exception ex) {
    JOptionPane.showMessageDialog(this, "Алдаа гарлаа: " + ex.getMessage(),
"Алдаа", JOptionPane.ERROR_MESSAGE);
}
}

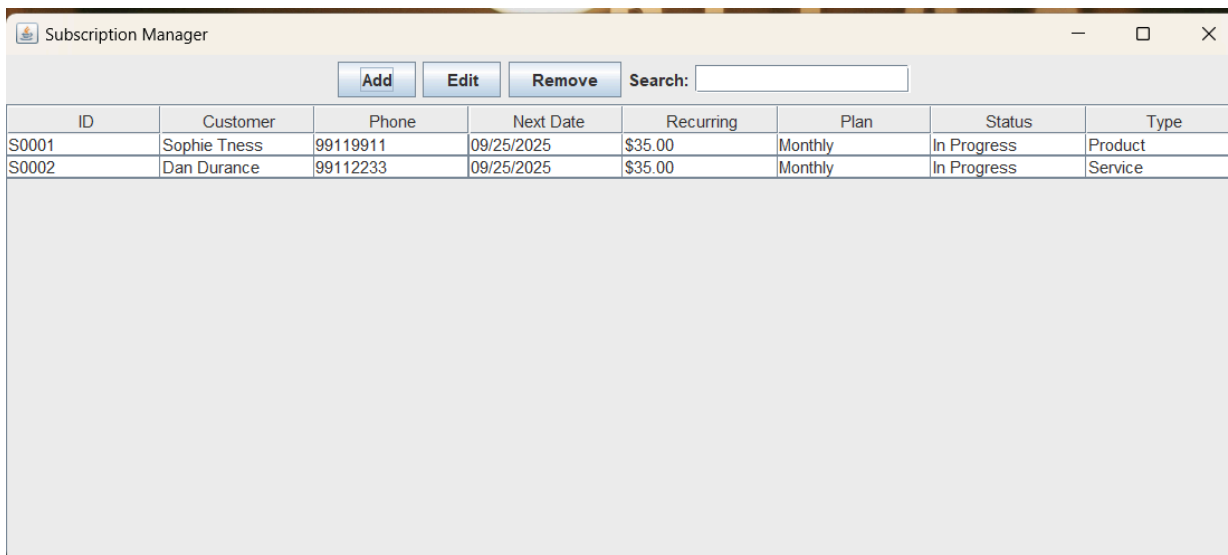
private void refreshTable() {
    tableModel.setRowCount(0);
    String keyword = searchField.getText();
    List<Subscription> list = keyword.isEmpty() ? manager.getSubscriptions() :
manager.filter(keyword);
    for (Subscription s : list) {
        tableModel.addRow(s.toRow());
    }
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(() -> new SubscriptionManagerGUI().setVisible(true));
}

```

```
}  
}
```

Кодийг compile хийсэн нь



ID	Customer	Phone	Next Date	Recurring	Plan	Status	Type
S0001	Sophie Tness	99119911	09/25/2025	\$35.00	Monthly	In Progress	Product
S0002	Dan Durance	99112233	09/25/2025	\$35.00	Monthly	In Progress	Service

Тестжүүлэлт

Ашигласан туршилтын хэрэгсэл: JUnit 5 (org.junit.jupiter.api)

```
import org.junit.jupiter.api.Test;
```

```
import java.time.format.DateTimeParseException;
```

```
import static org.junit.jupiter.api.Assertions.*;
```

```
public class SubscriptionValidatorTest {
```

```
    @Test
```

```
    public void testValidInput() {
```

```
        assertDoesNotThrow(() -> SubscriptionValidator.validate("Бат Энх", "99119911",  
"09-25-2025", "$35.00"));
```

```
    }
```

```
    @Test
```

```
    public void testInvalidCustomer() {
```

```
        Exception e = assertThrows(IllegalArgumentException.class, () ->
```

```
            SubscriptionValidator.validate("", "99119911", "09-25-2025", "$35.00"));
```

```
        assertEquals("Хэрэглэгчийн нэр дутуу байна.", e.getMessage());
```

```
    }
```

```
@Test
public void testInvalidPhone() {
    Exception e = assertThrows(IllegalArgumentException.class, () ->
        SubscriptionValidator.validate("Бат Энх", "99111", "09-25-2025", "$35.00"));
    assertEquals("Утасны дугаарыг зөв оруулна уу (8 оронтой).", e.getMessage());
}
```

```
@Test
public void testInvalidRecurringFormat() {
    Exception e = assertThrows(IllegalArgumentException.class, () ->
        SubscriptionValidator.validate("Бат Энх", "99119911", "09-25-2025", "35.00"));
    assertEquals("Дахин төлбөрийг зөв оруулна уу (жишээ: $35.00).", e.getMessage());
}
```

```
@Test
public void testRecurringTooSmall() {
    Exception e = assertThrows(IllegalArgumentException.class, () ->
        SubscriptionValidator.validate("Бат Энх", "99119911", "09-25-2025", "$0.00"));
    assertEquals("Дахин төлбөрийн дүн $0.01-аас бага байж болохгүй.", e.getMessage());
}
```

```
@Test
public void testInvalidDateFormat() {
    Exception e = assertThrows(DateTimeParseException.class, () ->
        SubscriptionValidator.validate("Бат Энх", "99119911", "2025-09-25", "$35.00"));
    assertTrue(e.getMessage().contains("Дараагийн огноог зөв оруулна уу"));
}
}
```

Үр дүн

```
C:\Users\tonto\lib
λ java -jar junit-platform-console-standalone-1.10.0.jar --class-path . --scan-class-path

? Thanks for using JUnit! Support its development at https://junit.org/sponsoring

BKR Desktop
+-- JUnit Jupiter [OK]
| '-- SubscriptionValidatorTest [OK]
|   +-- testInvalidCustomer() [OK]
|   +-- testValidInput() [OK]
|   +-- testInvalidDateFormat() [OK]
|   +-- testInvalidPhone() [OK]
|   +-- testInvalidRecurringFormat() [OK]
|   '-- testRecurringTooSmall() [OK]
+-- JUnit Vintage [OK]
'-- JUnit Platform Suite [OK]

Test run finished after 351 ms
[Browser 4 containers found ]
[ 0 containers skipped ]
[ 4 containers started ]
[ 0 containers aborted ]
[ 4 containers successful ]
[Git Bash 0 containers failed ]
[ 6 tests found ]
[ 0 tests skipped ]
[ 6 tests started ]
[ 0 tests aborted ]
[ 6 tests successful ]
[G AntiVirus 0 tests failed ]
Free

WARNING: Delegated to the 'execute' command.
This behaviour has been deprecated and will be removed in a future release.
Please use the 'execute' command directly.
```

<https://github.com/Tontoosh/day6>

