

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**  
**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO THÍ NGHIỆM/THỰC NGHIỆM**  
**LẬP TRÌNH JAVA**

**ĐỀ TÀI: Xây dựng phần mềm quản lý hoạt động**  
**CLB IT Supporter - HaUI**

**Giảng viên:** Ths.Vũ Thị Dương

**Nhóm:** 12

**Thành viên:** Nguyễn Hoàng Giang

Hồ Nam Tú

Cù Đức Xuân

**Lớp:** 20231IT6019001 **Khóa:** 16

**Hà Nội - Năm 2023**

## LỜI NÓI ĐẦU

Hiện nay, vai trò của Câu lạc bộ (CLB) IT Supporter không chỉ giới hạn trong việc hỗ trợ kỹ thuật mà còn mở rộng đến việc quản lý hoạt động nội bộ. Để đáp ứng nhu cầu này, chúng em đã quyết định thực hiện dự án xây dựng phần mềm quản lý hoạt động CLB IT Supporter - HaUI.

Bài tập lớn này không chỉ là một cơ hội để chúng em áp dụng những kiến thức đã học trong môn học Java mà còn là dịp để trải nghiệm quá trình phát triển phần mềm từ ý tưởng ban đầu đến sản phẩm hoàn chỉnh. Chúng em hy vọng rằng phần mềm sẽ đóng góp tích cực vào quá trình quản lý và tương tác trong cộng đồng của CLB IT Supporter.

Báo cáo này sẽ trình bày các chức năng, chi tiết hệ thống của phần mềm quản lý hoạt động CLB IT Supporter. Chúng em tin rằng những thông tin này không chỉ là kết quả của sự nỗ lực của chúng em mà còn là nguồn tham khảo hữu ích cho những người quan tâm đến phát triển phần mềm và quản lý hoạt động CLB. Trong quá trình tìm hiểu và thực hiện báo cáo nhóm chúng em không thể tránh khỏi những thiếu hụt và sai sót, vì vậy nhóm rất mong được nghe sự nhận xét và ý kiến đánh giá của cô dành cho nhóm.

Nhóm chúng em xin chân thành cảm ơn!

*Nhóm sinh viên thực hiện đề tài*

## MỤC LỤC

LỜI NÓI ĐẦU .....	1
I. MỞ ĐẦU .....	5
1. Lý do chọn đề tài .....	5
2. Mục tiêu và ý nghĩa .....	5
2.1. Mục tiêu .....	5
2.2. Ý nghĩa .....	5
3. Nội dung học tập và các kỹ năng, kiến thức then chốt .....	6
4. Bố cục chính .....	7
II. KẾT QUẢ NGHIÊN CỨU .....	8
1. Giới thiệu .....	8
2. Khảo sát hệ thống .....	9
2.2.1. Khảo sát sơ bộ .....	9
2.2.2. Tài liệu đặc tả yêu cầu .....	10
3. Phân tích hệ thống .....	11
3.1. Mô hình hóa chức năng hệ thống .....	11
3.2. Mô hình hóa dữ liệu .....	12
4. Thực hiện bài toán .....	14
4.1. Quản lý đăng nhập .....	14
4.1.1. Màn hình giới thiệu phần mềm .....	14
4.1.2. Quản lý thông tin đăng nhập .....	14
4.2. Quản lý chung .....	17
4.3. Quản lý Machine – Hồ Nam Tú .....	18
4.3.1. Quản lý chung Machine .....	18
4.3.2. Quản lý Machine_Add .....	20
4.3.3. Quản lý MachineShowAdd .....	22
4.3.4. Quản lý Machine_Show .....	23
4.3.5. Quản lý Machine_Edit .....	24
4.4. Quản lý Technician – Cù Đức Xuân .....	26
4.4.1. Quản lý chung Technician .....	26
4.4.2. Quản lý Technician_Add .....	29
4.4.3. Quản lý Technician_Show .....	31
4.4.4. Quản lý Technician_Edit .....	32
4.5. Quản lý Approval – Nguyễn Hoàng Giang .....	33
4.5.1. Quản lý chung Approval .....	33
4.5.2. Quản lý Approval_Show .....	36

4.5.3.	Quản lý Approval_Edit .....	37
4.5.4.	Quản lý Approval_Export .....	39
III.	KẾT LUẬN VÀ BÀI HỌC KINH NGHIỆM.....	41
	TÀI LIỆU THAM KHẢO .....	42

## **I. MỞ ĐẦU**

### **1. Lý do chọn đề tài**

Hiện nay, sự phát triển của công nghệ thông tin là một trong những lĩnh vực được phát triển hàng đầu với việc triển khai rộng rãi các ứng dụng cho các ngành nghề, tổ chức và xã hội. Cùng với đó là sự phát triển của phần mềm, phần mềm giúp tăng năng suất công việc, dễ dàng quản lý và lưu trữ thông tin.

Xây dựng phần mềm quản lý hoạt động CLB IT Supporter - HaUI là một đề tài hữu ích và thiết thực trong lĩnh vực quản lý. Có nhiều lý do để chọn đề tài này:

- Nhu cầu quản lý hiệu quả: Với số lượng đông đảo các bạn sinh viên tham dự các sự kiện Tech Support và sự phát triển của CLB Hỗ trợ kỹ thuật IT Supporter làm sự quản lý máy móc và nhân sự trở nên phức tạp. Một phần mềm quản lý CLB sẽ giúp đơn giản hóa quá trình này và tăng cường tính chính xác, hiệu quả của việc quản lý và tổ chức sự kiện cho CLB.
- Tiết kiệm thời gian và công sức: Việc sử dụng phần mềm quản lý hoạt động CLB IT Supporter - HaUI sẽ giảm bớt công việc thủ công và giấy tờ, điều này sẽ giúp cải thiện quá trình quản lý sự kiện và nhân sự CLB.

### **2. Mục tiêu và ý nghĩa**

#### **2.1. Mục tiêu**

- Mục tiêu chính của đề tài là ứng dụng ngôn ngữ lập trình Java vào vào việc phần mềm quản lý hoạt động CLB IT Supporter – HaUI.
- Tích hợp các giải pháp vào một phần mềm quản lý sự kiện và nhân sự một cách hiệu quả
- Nghiên cứu và cải thiện phần mềm quản lý CLB.

#### **2.2. Ý nghĩa**

- Đóng góp về mặt phương pháp luận và thực nghiệm vào lĩnh vực quản lý nhân sự và sự kiện của CLB Hỗ trợ kỹ thuật IT Supporter – HaUI một cách thực tế và hiệu quả.

- Cải tiến chất lượng phần mềm quản lý nhân sự và sự kiện để có thể nâng cao trình độ và kỹ năng áp dụng vào thực tế của CLB Hỗ trợ kỹ thuật IT Supporter – HaUI.

### **3. Nội dung học tập và các kỹ năng, kiến thức then chốt**

#### **3.1. Nội dung học tập**

- Cấu trúc của một chương trình Java
- Các kiểu dữ liệu và chuyển kiểu dữ liệu
- Các toán tử
- Các cấu trúc điều khiển
- Mảng và xử lý mảng
- Lớp và đối tượng trong Java
- Các hàm khởi tạo
- Phương thức tĩnh static
- Mảng đối tượng
- Kế thừa, kết tập
- Tính trừu tượng, đa hình và interface
- Ghi đè phương thức
- Xử lý ngoại lệ
- I/O theo luồng và thao tác với tệp
- Collection
- Giao diện Java Swing

#### **3.2. Kiến thức đã trang bị**

- Lập trình Java cơ sở
- Lập trình Java hướng đối tượng
- Xử lý ngoại lệ và thao tác với tệp
- Lập trình với cấu trúc Collection
- Lập trình giao diện Java Swing

### **3.3. Kỹ năng then chốt**

- Kỹ năng làm việc nhóm
- Kỹ năng phân tích và xử lý tình huống
- Kỹ năng thu thập và chuẩn hóa thông tin
- Kỹ năng xây dựng ý tưởng đề tài

## **4. Bố cục chính**

Bản báo cáo gồm 3 phần chính:

**Phần I: MỞ ĐẦU**

**Phần II: KẾT QUẢ NGHIÊN CỨU**

**Phần III: KẾT LUẬN VÀ BÀI HỌC KINH NGHIỆM**  
**TÀI LIỆU THAM KHẢO**

## II. KẾT QUẢ NGHIÊN CỨU

### 1. Giới thiệu

Đề tài nghiên cứu: “Xây dựng phần mềm quản lý hoạt động CLB IT Supporter – HaUI ”.

Để xây dựng bài toán trên, nhóm đã tìm hiểu, thảo luận và quyết định lựa chọn mô hình thác nước để xác định các giai đoạn xây dựng sản phẩm của nhóm:

- Phân tích yêu cầu: Thu thập và phân tích yêu cầu của hệ thống quản lý và tổ chức kỳ thi.
- Thiết kế hệ thống: Phân tích thiết kế hệ thống phần mềm quản lý và tổ chức kỳ thi, xác định các kiến trúc tổng thể của phần mềm.
- Xây dựng chương trình: theo thiết kế tạo ra các chương trình (units), tích hợp units cho giai đoạn tiếp theo. Mỗi unit được phát triển và kiểm thử gọi là Unit test.
- Kiểm thử: Cài đặt phần mềm và kiểm thử phần mềm nhằm kiểm tra và sửa những lỗi tìm được sao cho phần mềm hoạt động chính xác và đúng theo tài liệu đặc tả yêu cầu.
- Triển khai hệ thống: Triển khai hệ thống được sử dụng trong các hệ điều hành.

**Kết quả đạt được:** Phần mềm quản lý hoạt động CLB IT Supporter – HaUI sử dụng ngôn ngữ lập trình Java.

#### **Mô tả sản phẩm nghiên cứu:**

- Tên sản phẩm: Phần mềm quản lý hoạt động CLB IT Supporter – HaUI.
- Hình thức sản phẩm: Phần mềm ứng dụng chạy trên desktop
- Cấu trúc: Ngôn ngữ java
- Nội dung sản phẩm:



+ Mô tả: Ứng dụng giúp CLB IT Supporter quản lý các hoạt động của sự kiện: quản lý Machine , quản lý Technician , quản lý Approval.

+ Chức năng:

- Quản lý Machine: Hiện thị, thêm, sửa, xóa danh sách các máy của khách hàng.
- Quản lý Technician: Hiện thị, thêm, sửa, xóa danh sách thành viên của đội kỹ thuật.
- Quản lý Approval: Hiện thị, sửa, xuất hóa đơn cho khách hàng.

## **2. Khảo sát hệ thống**

### **2.2.1. Khảo sát sơ bộ**

Câu lạc bộ Hỗ trợ kỹ thuật IT Supporter – HaUI được thành lập vào ngày 29/08/2014, qua 9 năm hình thành và phát triển CLB Hỗ trợ kỹ thuật IT Supporter đã có những bước tiến lớn trong quá trình trưởng thành của mình. CLB được thành lập với mục đích hỗ trợ những vấn đề liên quan đến kỹ thuật khoa CNTT. Tạo môi trường cho các sinh viên trao đổi học tập về những vấn đề kỹ thuật và thỏa mãn được đam mê của mình về giải quyết các vấn đề liên quan đến máy tính. Ứng dụng những kinh nghiệm thực chiến về kỹ thuật để xây dựng được những kỹ năng cần thiết cho định hướng tương lai.

CLB Hỗ trợ kỹ thuật IT Supporter là địa chỉ tin tưởng của tất cả các bạn sinh viên trong và ngoài trường Đại Học Công Nghiệp Hà Nội với nhiệm vụ hỗ trợ, tư vấn các bạn các vấn đề về máy tính một cách chuyên sâu. Tiếp đó, CLB còn là nơi BGH khoa CNTT tin tưởng giao phó bảo trì, cập nhật phần cứng, phần mềm cho các phòng máy thực hành.

CLB Hỗ trợ kỹ thuật IT Supporter có nhiều hoạt động nhằm thúc đẩy sự phát triển kỹ năng của các thành viên và đem đến cho tất cả các bạn sinh viên một địa chỉ tin cậy để giao phó sự tin tưởng của mình. Có thể kể đến một số hoạt động

như sau:

Tech Support thường niên với nhiều hạng mục nhằm hỗ trợ những vấn đề về máy tính cho giảng viên sinh viên trong toàn trường

- + Vệ sinh máy tính
- + Tra keo tản nhiệt
- + Cài đặt phần mềm, windows.
- + Nâng cấp phần cứng với mức giá sinh viên chất lượng giảng viên
- + Tư vấn laptop phù hợp với bản thân

Bảo trì, cài đặt phòng máy ở khoa CNTT, giúp cho những phòng máy cập nhật kịp thời những phần mềm mới nhất, hoạt động mượt mà nhất cho những giờ thực hành trên lớp.

Để dễ dàng quản lý các hoạt động sự kiện của CLB thì việc sử dụng phần mềm này là vô cùng cần thiết. Qua quá trình khảo sát và phân tích các dữ liệu thu được, phần mềm quản lý hoạt động đặt ra các vấn đề cơ bản sau:

- Quản lý Machine: Đúng thông tin máy, yêu cầu của từng khách hàng.
- Quản lý Technician: Đúng thông tin thành viên, nhiệm vụ của thành viên.
- Quản lý Approval: Cập nhật tiến trình hoạt động của công việc.

### **2.2.2. Tài liệu đặc tả yêu cầu**

#### **Mô tả hệ thống:**

- Hệ thống quản lý hoạt động CLB IT Supporter – HaUI cho phép người quản trị quản lý thành viên của đội kỹ thuật CLB và quản lý các hoạt động sự kiện. Cho phép thành viên đội kỹ thuật CLB quản lý các hoạt động sự kiện: thêm, sửa, xóa máy khách hàng, cập nhật tiến trình, xuất hóa đơn.

#### **Giao diện và chức năng của hệ thống:**

- Hệ thống gồm 5 giao diện chính:
  - + Giao diện Đăng nhập
  - + Giao diện Trang chủ

- + Giao diện Quản lý Machine
- + Giao diện Quản lý Technician
- + Giao diện Quản lý Approval
- Chức năng:
  - + Giao diện đăng nhập: cho phép người quản trị, thành viên có quyền đăng nhập
  - + Giao diện trang chủ: cho phép người quản trị lựa chọn các mục quản lý của CLB
  - + Quản lý Machine: Hiển thị danh sách máy của khách hàng, thêm, sửa, xóa, xem chi tiết máy.
  - + Quản lý Technician: Hiển thị danh sách thành viên đội kỹ thuật, thêm, sửa, xóa, xem chi tiết thành viên.
  - + Quản lý Approval: Hiển thị danh sách máy đang được nhận bởi Technician nào và tiến trình hoạt động sửa chữa, sửa, xem chi tiết và xuất hóa đơn.

### **3. Phân tích hệ thống**

#### **3.1. Mô hình hóa chức năng hệ thống**

##### **Mô tả sơ bộ các Actor và nhiệm vụ chính:**

- Technician: Có nhiệm vụ thêm máy cho khách hàng, chỉnh sửa thông tin máy, xóa máy và có thể nhận máy nếu muốn.
- Admin: Có nhiệm vụ quản lý technician, bao gồm thêm, sửa, xóa. Ngoài ra admin cũng có thể thực hiện tất cả chức năng của technician.

##### **Các use case và nhiệm vụ:**

- Machine: cho phép admin và technician quản lý thông tin các máy tính nhận vào bao gồm thêm, sửa, xóa, xem máy.

- Technician: Cho phép admin quản lý thông tin của các technician hiện đang hoạt động trong câu lạc bộ bao gồm thêm, sửa, xóa. Ngoài ra admin và technician cũng có thể xem thông tin của các technician khác.
- Approval: Cho phép technician và admin thực hiện xem, in hóa đơn, chỉnh sửa thông tin trạng thái của technician trong quá trình thực hiện theo yêu cầu khách hàng.

### Biểu đồ use case

ID	Tên use case	Mô tả ngắn gọn	Chức năng	Ghi chú
UC_01	Machine	Quản lý machine	Cho phép admin và technician xem, thêm, sửa, xóa và nhận máy	
UC_02	Technician	Quản lý technician	Cho phép admin xem thông tin, thêm, sửa, xóa technician. Technician cũng có thể tự xem được thông tin của technician khác	
UC_03	Approval	Quản lý approval	Cho phép admin và technician có thể xem, sửa và in thông tin hóa đơn	

## 3.2. Mô hình hóa dữ liệu

### Machine

Tên	Kiểu dữ liệu	Mô tả
serial	String	Số serial trên máy

name	String	Tên khách hàng
phone	String	Số điện thoại khách hàng
requirements	String	Yêu cầu của khách hàng
notes	String	Ghi chú của khách hàng
statements	String	Trạng thái của máy

### Technician

Tên	Kiểu dữ liệu	Mô tả
name	String	Tên của kỹ thuật viên
phone	String	Số điện thoại của kỹ thuật viên
studentCode	String	Mã sinh viên
classLearn	String	Lớp học và khóa hiện tại của kỹ thuật viên
accountName	String	Tên tài khoản kỹ thuật viên khi đăng nhập
password	String	Mật khẩu của kỹ thuật viên khi đăng nhập

### Bill

Tên	Kiểu dữ liệu	Mô tả
billID	String	Mã hóa đơn
nameClient	String	Tên của khách hàng
phoneClient	String	Số điện thoại của khách hàng
serialMachine	String	Số serial trên máy của khách hàng
requirements	String	Yêu cầu của khách hàng
statements	String	Trạng thái của máy
nameTechnician	String	Tên của kỹ thuật viên
phoneTechnician	String	Số điện thoại của kỹ thuật viên

account	String	Tài khoản mà kỹ thuật viên sử dụng
note	String	Ghi chú trên hóa đơn

## 4. Thực hiện bài toán

### 4.1. Quản lý đăng nhập

#### 4.1.1. Màn hình giới thiệu phần mềm



Hình 2.1: Giao diện màn hình Login

Giao diện trang chủ được thực hiện bằng cơ chế kéo thả các nút và set icon cho Label. Người quản lý hoặc Technician có thể đăng nhập bằng nút Login.

#### 4.1.2. Quản lý thông tin đăng nhập



Hình 2.2: Giao diện màn hình Login1 (Điền thông tin đăng nhập)

Giao diện màn hình Login1 được thực hiện bằng cơ chế:

- Kéo thả các Button: Login, Exit; Radio Button: Show Password.
- Kéo thả các TextField: ID, Password.

Người quản lý và Technician có thể điền ID và Password được cấp để đăng nhập, có thể nhấn nút Show Password để xem ký tự của mật khẩu. Sau khi nhập ID và Password thì nhấn nút Login để đăng nhập và nhấn nút Exit để quay về màn hình giới thiệu.

- Hướng đối tượng:

- + Lớp Login\_1 được kết thừa từ javax.swing.JFrame

```
public class Login_1 extends javax.swing.JFrame {
```

- Bắt lỗi

- + Bắt lỗi chưa điền thông tin đăng nhập

```
if (account.isEmpty() || password.isEmpty()) {
    JOptionPane.showMessageDialog(parentComponent: this,
        message: "Bạn cần nhập đầy đủ ID và Password", title: "Message", messageType: JOptionPane.WARNING_MESSAGE);
}
```

### + Bắt lỗi điền sai thông tin đăng nhập

```
} else {
    List<Technician> techList = controller.readDataFromFile(fileName: "src/file/technician.txt");
    boolean check = false;
    for (Technician i : techList) {
        if (i.getAccountName().equals(anObject: account) && i.getPassword().equals(anObject: password)) {
            controller.writeAccountToFile(account: i.getAccountName(), fileName: "src/file/account.txt");
            check = true;
        }
    }

    if (check) {
        new MainMenu().setVisible(b: true);
        this.dispose();
        JOptionPane.showMessageDialog(parentComponent: this, message: "Đăng nhập thành công",
            title: "Message", messageType: JOptionPane.INFORMATION_MESSAGE);
    } else {
        JOptionPane.showMessageDialog(parentComponent: this, message: "Đăng nhập không thành công, vui lòng đăng nhập lại ! ",
            title: "Message", messageType: JOptionPane.ERROR_MESSAGE);
    }
}
```

### - Tập hợp

Hệ thống sử dụng ArrayList, một cấu trúc dữ liệu được sử dụng để lưu trữ và quản lý một tập hợp các phần tử động. Cho phép thêm phần tử vào cuối danh sách, truy cập vào phần tử tại một vị trí bất kì...

```
List<Technician> techList =
    controller.readDataFromFile(fileName: "src/file/technician.txt");
```

### + Get()

```
for (Technician i : techList) {
    if (i.getAccountName().equals(anObject: account) && i.getPassword().equals(anObject: password)) {
        controller.writeAccountToFile(account: i.getAccountName(), fileName: "src/file/account.txt");
        check = true;
    }
}
```

### - Thao tác với file

#### + readDataFromFile



```

@Override
public <T> List<T> readDataFromFile(String fileName) {
    List<T> list = new ArrayList<>();
    File file = new File(pathname: fileName);
    if (file.length() > 0) {
        try {
            file.createNewFile();
            FileInputStream fos = new FileInputStream(file);
            ObjectInputStream oos = new ObjectInputStream(in: fos);
            Object o = oos.readObject();
            list = (List<T>) o;
            oos.close();
            fos.close();
        } catch (IOException ex) {
            ex.printStackTrace();
        } catch (ClassNotFoundException ex) {
            ex.printStackTrace();
        }
    }
    return list;
}

```

```

List<Technician> techList =
    controller.readDataFromFile(fileName: "src/file/technician.txt");

```

+ writeAccountToFile

```

@Override
public void writeAccoutToFile(String accout, String fileName){
    try {
        FileWriter fileWriter = new FileWriter(fileName);
        fileWriter.write(str: accout);
        fileWriter.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
}

controller.writeAccoutToFile(accout: i.getAccountName(), fileName: "src/file/account.txt");

```

Hệ thống sử dụng thao tác với tệp theo 2 hàm readDataFromFile và writeAccountToFile giúp nhập và xuất đối tượng vào file. Đối tượng được đọc tại file *technician.txt* và được xuất ra file *account.txt*.

## 4.2. Quản lý chung



Hình 2.3: Màn hình quản lý chung

Giao diện trang quản lý chung được thực hiện bằng cơ chế:

- Kéo thả các Button
- Set icon cho Label

Người quản lý hoặc Technician có thể lựa chọn các đối tượng để bắt đầu thao tác quản lý như: Machine, Technician, Approval. Nếu người quản trị hoặc Technician không muốn tiếp tục thao tác thì có thể ấn Logout để thoát ra màn hình giới thiệu.

- Hướng đối tượng
- + Lớp MainMenu được kế thừa từ javax.swing.JFrame

```
public class MainMenu extends javax.swing.JFrame {
```

### 4.3. Quản lý Machine – Hồ Nam Tú

#### 4.3.1. Quản lý chung Machine



Hình 2.3: Màn hình Machine

Giao diện màn hình Machine được thực hiện bằng cơ chế:

- Kéo thả các Button: Machine, Technician, Approval, ADD, SHOW, EDIT, DELETE, Logout.
- Kéo thả JTable để hiển thị danh sách máy được nhận vào.

Người quản lý hoặc Technician có thể thao tác với các máy nhận vào qua các Button ADD, SHOW, EDIT, DELETE. Nút Logout để đưa người quản lý hoặc Technician về màn hình giới thiệu.

- Hướng đối tượng
  - + Lớp Machine\_Screen kế thừa từ lớp javax.swing.JFrame

```
public class Machine_Screen extends javax.swing.JFrame {
```

- Bắt lỗi
  - + Bắt lỗi danh sách máy chưa có máy

```
if (machineList.size() == 0) {
    showMessage(str: "Danh sách máy trống");
```

+ Bắt lỗi chưa chọn máy để th

```
} else if (indexDelete == -1) {
    showMessage(str: "Chưa chọn máy cần xóa");
```

#### - Tập hợp

```
List<Machine> machineList = controller.readDataFromFile
(fileName: "src/file/machine.txt");
```

#### - Thao tác với file

+ readDataFromFile

```
List<Machine> machineList = controller.readDataFromFile
(fileName: "src/file/machine.txt");
```

@Override

```
public <T> List<T> readDataFromFile(String fileName) {
    List<T> list = new ArrayList<>();
    File file = new File(pathname: fileName);
    if (file.length() > 0) {
        try {
            file.createNewFile();
            FileInputStream fos = new FileInputStream(file);
            ObjectInputStream oos = new ObjectInputStream(in: fos);
            Object o = oos.readObject();
            list = (List<T>) o;
            oos.close();
            fos.close();
        } catch (IOException ex) {
            ex.printStackTrace();
        } catch (ClassNotFoundException ex) {
            ex.printStackTrace();
        }
    }
    return list;
}
```

### 4.3.2. Quản lý Machine\_Add

The screenshot shows a web application window titled "Nhóm 12". The main header is "IT Supporter Management System" with "admin" and "Logout" buttons. The left sidebar has three buttons: "Machine" (active), "Technician", and "Approval". The "Add Machine" form includes fields for "Họ tên", "Điện thoại", "Serial Machine", "Yêu cầu", and "Ghi chú", along with "ADD" and "EXIT" buttons.

*Hình 2.4: Màn hình Machine\_Add*

Giao diện Add Machine được thực hiện bằng cơ chế:

- Kéo thả các Button: Machine, Technician, Approval, ADD, EXIT, Logout
- Kéo thả các JTextField: Họ tên, Điện thoại, Serial Machine, Yêu cầu, Ghi chú.

Người quản lý hoặc Technician sẽ điền các thông tin của máy nhận vào các TextField và sau khi điền đầy đủ nhấn ADD thì máy sẽ được thêm vào. Nếu muốn chuyển sang các chức năng khác thì nhấn vào nút Technician hoặc Approval. Nếu muốn thoát ấn EXIT sẽ trở về màn hình quản lý Machine.

- Bắt lỗi và gom rác

```

if (hoTen.isEmpty()) {
    JOptionPane.showMessageDialog(parentComponent: this, message: "Không được để trống họ tên", title: "Lỗi", messageType: JOptionPane.OK_OPTION);
} else if (SDT.isEmpty()) {
    JOptionPane.showMessageDialog(parentComponent: this, message: "Không được để trống số điện thoại", title: "Lỗi", messageType: JOptionPane.OK_OPTION);
} else if (yêuCau.isEmpty()) {
    JOptionPane.showMessageDialog(parentComponent: this, message: "Không được để trống yêu cầu", title: "Lỗi", messageType: JOptionPane.OK_OPTION);
} else if (serial.isEmpty()) {
    JOptionPane.showMessageDialog(parentComponent: this, message: "Không được để trống số sê - ri", title: "Lỗi", messageType: JOptionPane.OK_OPTION);
} else if (!checkSerial(serial)) {
    JOptionPane.showMessageDialog(parentComponent: this, message: "Serial đã tồn tại, hãy kiểm tra lại Serial của máy", title: "Lỗi", messageType: JOptionPane.ERROR_MESSAGE);
}
else {
    int option = JOptionPane.showConfirmDialog(parentComponent: this, message: "Bạn chắc chắn muốn thêm máy này chứ ?", title: "Xác nhận", optionType: JOptionPane.OK_CANCEL_OPTION);
    if (option == JOptionPane.OK_OPTION) {
        Machine machine = new Machine(name: hoTen, phone: SDT, serial, requirements: yêuCau, statements: "Chưa nhận", notes: ghiChu);
        machineList.add(e: machine);
        controller.writeToFile(list: machineList, fileName: "src/file/machine.txt");
        new Machine_Screen().setVisible(b: true);
        this.dispose();
    }
}
}

```

### 4.3.3. Quản lý MachineShowAdd

Hình 2.5: Màn hình MachineShowAdd

Giao diện MachineShowAdd được thực hiện bằng cơ chế:

- Kéo thả các Button Machine, Technician, Approval, Get machine, EXIT, Logout.
- Kéo thả các JTextField Họ tên, Điện thoại, Serial Machine, Yêu cầu, Ghi chú.

Người quản lý hoặc Technician sẽ được hiển thị màn hình MachineShowAdd khi chưa nhận hoặc hoàn thành máy để nhận máy để làm tiếp theo. Người quản lý hoặc Technician không muốn nhận máy thì nữa thì thoát ra bằng nút EXIT. Người quản lý hoặc Technician có thể Logout khi cần.

- Thao tác với file:

- + Đọc từ file

```
List<Machine> machineList = controller.readDataFromFile(fileName: "src/file/machine.txt");
List<Bill> billList = controller.readDataFromFile(fileName: "src/file/bill.txt");
```

- + Ghi vào file

```

} else {
    deleteApproval(indexDelete);
    machineList.remove(index: indexDelete);
    showData(list: machineList, model: modelMachine);
    showMessage(str: "Máy đã được xóa");
    controller.writeFile(list: machineList, fileName: "src/file/Machine.txt");
}

```

- Collection

- + remove(): Xóa một bản ghi

```

List<Bill> billList = controller.readDataFromFile(fileName: "src/file/bill.txt");
for(Bill bill : billList){
    if((bill.getSerialMachine()).equals(anObject: (machineList.get(index: indexDelete)).getSerial())){
        billList.remove(o: bill);
        break;
    }
}
} else {
    deleteApproval(indexDelete);
    machineList.remove(index: indexDelete);
    showData(list: machineList, model: modelMachine);
    showMessage(str: "Máy đã được xóa");
    controller.writeFile(list: machineList, fileName: "src/file/Machine.txt");
}

```

#### 4.3.4. Quản lý Machine\_Show

The screenshot displays a web application window titled "IT Supporter Management System". The window has a header bar with a logo on the left and two buttons, "tuhn" and "Logout", on the right. The main content area is divided into a left sidebar and a main form area. The sidebar contains three buttons: "Machine" (highlighted in orange), "Technician", and "Approval". The main form area is titled "Show Machine" and contains five input fields: "Họ tên" (Last Name) with the value "Cù Đức Xuân", "Điện thoại" (Phone Number) with the value "0398xxxxxx", "Serial Machine" with the value "LNV655", "Yêu cầu" (Request) with the value "Cài lại win", and "Ghi chú" (Note). At the bottom right of the form is an "EXIT" button.

*Hình 2.4: Màn hình Machine\_Show*

Màn hình Machine\_Show được thực hiện bằng cơ chế:

- Kéo thả các Button Machine, Technician, Approval, EXIT, Logout
- Kéo thả các JTextField Họ tên, Điện thoại, Serial Machine, Yêu cầu, Ghi chú.

Người quản lý hoặc Technician có thể xem chi tiết thông tin về máy tính ở màn hình này, nếu không muốn xem nữa người quản lý hoặc Technician có thể ấn nút EXIT để trở lại màn hình quản lý máy. Người quản lý hoặc Technician có thể Logout nếu cần thiết.

#### 4.3.5. Quản lý Machine\_Edit



The screenshot shows a web application window titled "Nhóm 12". The main header is black with the text "IT Supporter Management System" in white. To the right of the header are two buttons: "tuhn" and "Logout". On the left side of the main content area, there are three buttons: "Machine" (highlighted in orange), "Technician", and "Approval". The main content area is titled "Edit Machine". It contains five text input fields with the following labels and values: "Họ tên" (Cù Đức Xuân), "Điện thoại" (0398xxxxxx), "Serial Machine" (LNV655), "Yêu cầu" (Cài lại win), and "Ghi chú". At the bottom right of the form are two buttons: "COMPLETE" and "EXIT".

*Hình 2.6: Màn hình Machine\_Edit*

Màn hình Machine\_Edit được thực hiện theo cơ chế:

- Kéo thả các Button Machine, Technician, Approval, COMPLETE, EXIT, Logout
- Kéo thả các JTextField Họ tên, Điện thoại, Serial Machine, Yêu cầu, Ghi chú.

Người quản lý hoặc Technician có thể chỉnh sửa thông tin máy nhận bằng cách điền lại các trường thông tin ở màn hình Machine\_Edit. Lúc hoàn thành người quản lý hoặc Technician sẽ nhấn nút Complete để hoàn tất sửa đổi thông tin máy. Người quản lý hoặc Technician có thể thoát sửa đổi bằng cách nhấn nút Exit. Người quản lý hoặc Technician có thể Logout khi cần thiết.

- Bắt lỗi và gom rác

```

if (machine.equals(obj.newMachine)) {
    JOptionPane.showMessageDialog(parentComponent:this,
        message:"Bạn chưa chỉnh sửa thông tin, vui lòng nhấn chỉnh sửa thông tin bạn mong muốn ! ", title: "Lỗi", messageType:JOptionPane.OK_OPTION);
} else {
    int option = JOptionPane.showConfirmDialog(parentComponent:this,
        message:"Bạn chắc chắn về thực hiện chỉnh sửa thông tin Machine của mình chứ ? ", title: "Xác nhận", optionType: JOptionPane.OK_CANCEL_OPTION);
    if (option == JOptionPane.OK_OPTION) {
        machineList.set(index, element:newMachine);
        controller.writeFile(list:machineList, fileName:"src/file/machine.txt");
        JOptionPane.showMessageDialog(parentComponent:this,
            message:"Bạn đã cập nhật thành công ! ", title: "Thông báo", messageType:JOptionPane.OK_OPTION);
        new Machine_Screen().setVisible(b: true);
        this.dispose();
    }
}
}

```

- Thao tác với file

+ Đọc từ file

```
String account = controller.readAccountFromFile(fileName:"src/file/account.txt");
```

+ Ghi vào file

```
controller.writeFile(list:machineList, fileName:"src/file/machine.txt");
```

- Collection

+ set(): Đặt giá trị phần tử trong ArrayList

```

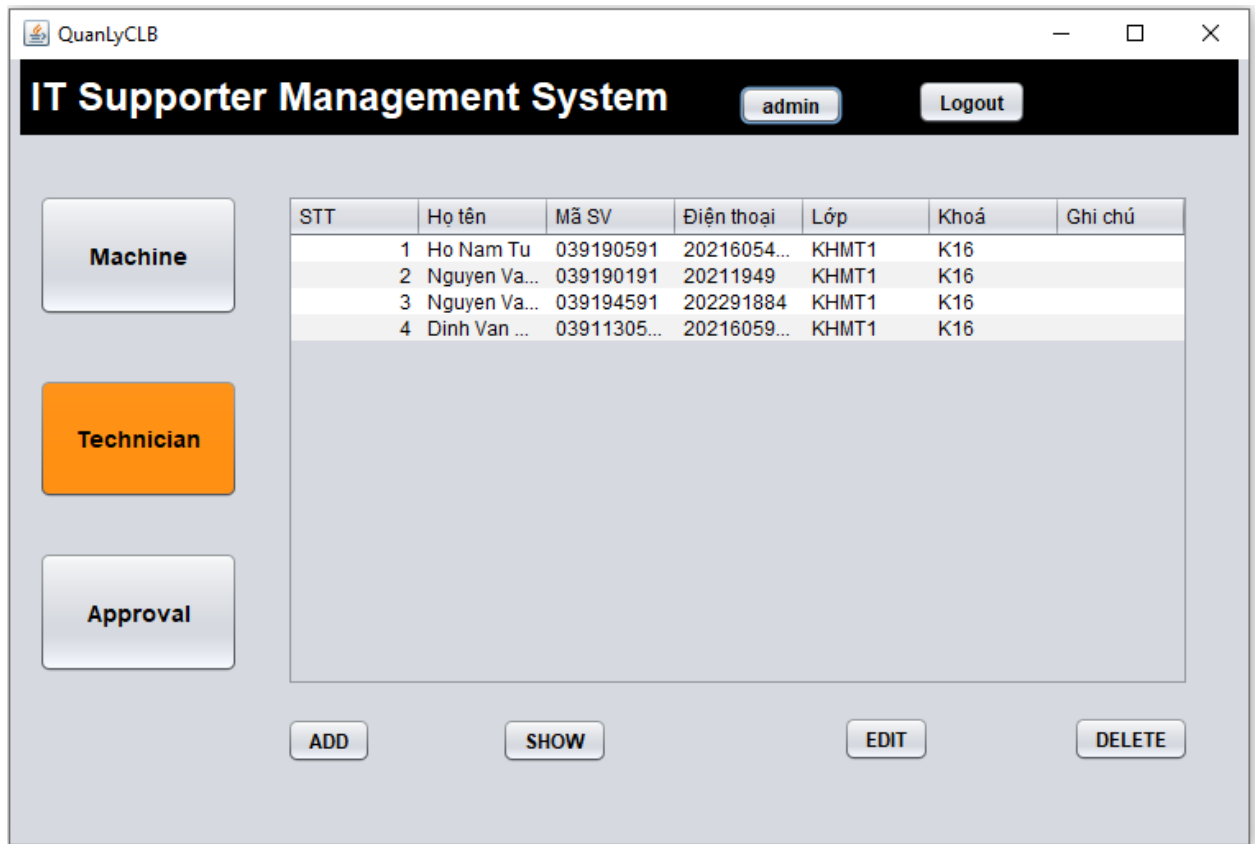
Machine newMachine = new Machine(name:hoTen, phone:dienThoai,
    serial, requirements:yeuCau, statements:machine.getStatements(), notes:ghiChu);

machineList.set(index, element:newMachine);

```

## 4.4. Quản lý Technician – Cù Đức Xuân

### 4.4.1. Quản lý chung Technician



Hình 2.7: Giao diện quản lý chung

Giao diện màn hình quản lý chung của Technician được thực hiện bằng cơ chế kéo thả các nút. Admin có thể lựa chọn các chức năng như: Add, Edit, Delete. Ngoài ra, Admin và Technician đều có thể xem thông tin của các technician được hiển thị trên bảng và logout khi cần thiết.

- Thực hiện bắt lỗi và thu gom rác:

```
private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String account = controller.readAccountFromFile(fileName: "src/file/account.txt");
    if ("admin".equals(account)) {
        int selectedRow = tblTechnician.getSelectedRow();
        try {
            if (selectedRow == -1) {
                JOptionPane.showMessageDialog(parentComponent, this, message: "Hãy chọn dòng để xóa", title: "Error", messageType: JOptionPane.ERROR_MESSAGE);
                throw new Exception(message: "Lỗi chọn dòng");
            } else {
                int option = JOptionPane.showConfirmDialog(parentComponent, this, message: "Bạn có chắc chắn muốn xóa Technician", title: "Confirmation", optionType: JOptionPane.YES_NO_OPTION);
                if (option == JOptionPane.YES_OPTION) {
                    DefaultTableModel model = (DefaultTableModel) tblTechnician.getModel();
                    Object value = model.getValueAt(row: selectedRow, column: 1);
                    List<Technician> technicianList = controller.readDataFromFile(fileName: "src/file/technician.txt");
                    if (value != null) {
                        technicianList.removeIf(technician: technicianList.get(0).getName().equals(value));
                        controller.writeDataToFile(technicianList, fileName: "src/file/technician.txt");
                        JOptionPane.showMessageDialog(parentComponent, this, message: "Đòng bạn chọn đang trống", title: "Lỗi", messageType: JOptionPane.ERROR_MESSAGE);
                        throw new Exception(message: "Dòng chọn lỗi");
                    }
                }
            }
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    } else {
        JOptionPane.showMessageDialog(parentComponent, null, message: "Tài khoản của bạn không có quyền thêm technician, vui lòng liên hệ admin để thêm !", title: "Lỗi", messageType: JOptionPane.OK_OPTION);
    }
}
```

```

private void btnEditActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String account = controller.readAccountFromFile(fileName: "src/file/account.txt");
    if ("admin".equals(account)) {
        int selectedRow = tblTechnician.getSelectedRow();
        try {
            if (selectedRow == -1) {
                JOptionPane.showMessageDialog(parentComponent: this, message: "Vui lòng chọn dòng cần chỉnh sửa", title: "Lỗi", messageType: JOptionPane.ERROR_MESSAGE);
                throw new Exception(message: "Lỗi chọn dòng");
            } else {
                DefaultTableModel model = (DefaultTableModel) tblTechnician.getModel();
                String hoTen = (String) model.getValueAt(row: selectedRow, column: 1);
                String maSV = (String) model.getValueAt(row: selectedRow, column: 3);
                String dienThoai = (String) model.getValueAt(row: selectedRow, column: 2);
                String lopKhoa = (String) model.getValueAt(row: selectedRow, column: 4) + "-" + (String) model.getValueAt(row: selectedRow, column: 5);
                String accountTech = "", password = "";
                List<Technician> technicianList = controller.readDataFromFile(fileName: "src/file/technician.txt");
                int index = -1;
                int ps = 0;
                for (Technician i : technicianList) {
                    if (maSV.equals(i.getStudentCode())) {
                        accountTech = i.getAccountName();
                        password = i.getPassword();
                    }
                    ps++;
                }
                if (index == -1) {
                    JOptionPane.showMessageDialog(parentComponent: this, message: "Không tìm thấy dòng trên trong cơ sở dữ liệu", title: "Lỗi", messageType: JOptionPane.ERROR_MESSAGE);
                    throw new Exception(message: "Lỗi index");
                } else {
                    ...5 lines
                }
            } catch (Exception ex) {
                ex.printStackTrace();
            }
        } else {
            ...3 lines
        }
    }
}

```

Ta sử dụng try – catch để xử lý ngoại lệ. Nếu ngoại lệ xảy ra, hệ thống sẽ hiển thị thông báo lên màn hình. Ngược lại nếu không có ngoại lệ, hệ thống sẽ hoạt động bình thường.

- Tập hợp:

Hệ thống sử dụng List, một cấu trúc dữ liệu được sử dụng để lưu trữ và quản lý một tập hợp các phần tử. Cho phép thêm phần tử vào cuối danh sách, truy cập vào phần tử tại một vị trí bất kỳ.

```
List<Technician> technicianList = controller.readDataFromFile(fileName: "src/file/technician.txt");
```

- Đọc và ghi file

Với hệ thống này, ta sử dụng 1 đối tượng MyController để thực thi interface controller trong package controller. Với bảng ta sẽ đọc từ file “technician.txt” và dữ liệu để hiển thị tên account ta sẽ đọc từ file “account.txt”.

- + Đọc file

```

public Technician_Screen() {
    String account = controller.readAccountFromFile(fileName: "src/file/account.txt");
    List<Technician> technicianList = controller.readDataFromFile(fileName: "src/file/technician.txt");
    initComponents();
    setLocationRelativeTo(c: null);
    bntAccout.setText(text: account);
    modelTechnician = (DefaultTableModel) tblTechnician.getModel();
    this.showData(list: technicianList, model: modelTechnician);
    bntTechnician.setBackground(bg: Color.decode(hex: "#fb8500"));
}

```

+ Lưu và chỉnh sửa file

```

controller.writeToFile(list: technicianList, fileName: "src/file/technician.txt");

```

- Thao tác delete một bản ghi trên file

```

private void bntDeleteActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String account = controller.readAccountFromFile(fileName: "src/file/account.txt");
    if ("admin".equals(account)) {
        int selectedRow = tblTechnician.getSelectedRow();
        try {
            if (selectedRow == -1) {
                JOptionPane.showMessageDialog(parentComponent: this, message: "Hãy chọn dòng để xóa", title: "Error", messageType: JOptionPane.ERROR_MESSAGE);
                throw new Exception(message: "Lỗi chọn dòng");
            } else {
                int option = JOptionPane.showConfirmDialog(parentComponent: this, message: "Bạn có chắc chắn muốn xóa Technician", title: "Confirmation", optionType: JOptionPane.YES_NO_OPTION);

                if (option == JOptionPane.YES_OPTION) {
                    DefaultTableModel model = (DefaultTableModel) tblTechnician.getModel();
                    Object value = model.getValueAt(row: selectedRow, column: 1);
                    List<Technician> technicianList = controller.readDataFromFile(fileName: "src/file/technician.txt");
                    if (value != null) {
                        model.removeRow(row: selectedRow);

                        // Cập nhật lại STT
                        for (int i = 0; i < model.getRowCount(); i++) {
                            model.setValueAt(value: i, row: i, column: 0);
                        }
                        technicianList.remove(index: selectedRow);
                        controller.writeToFile(list: technicianList, fileName: "src/file/technician.txt");

                        JOptionPane.showMessageDialog(parentComponent: this, message: "Đã xóa Technician này", title: "Message", messageType: JOptionPane.INFORMATION_MESSAGE);
                    } else {
                        JOptionPane.showMessageDialog(parentComponent: this, message: "Dòng bạn chọn đang trống", title: "Lỗi", messageType: JOptionPane.ERROR_MESSAGE);
                        throw new Exception(message: "Dòng chọn lỗi");
                    }
                }
            }
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    } else {
        JOptionPane.showMessageDialog(parentComponent: null, message: "Tài khoản của bạn không có quyền thêm technician, vui lòng liên hệ admin để thêm !", title: "Lỗi", messageType: JOptionPane.OK_OPTION);
    }
}

```

#### 4.4.2. Quản lý Technician\_Add

Hình 2.8: Giao diện Technician\_Add

- Code thực hiện chuyển màn hình từ Technician\_Screen

```
private void btnAddActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String account = controller.readAccountFromFile(fileName: "src/file/account.txt");
    if ("admin".equals(account)) {
        new Technician_Add().setVisible(true);
        this.dispose();
    } else {
        JOptionPane.showMessageDialog(parentComponent: null, message: "Tài khoản của bạn không có quyền thêm technician, vui lòng liên hệ admin để thêm !", title: "Lỗi", messageType: JOptionPane.OK_OPTION);
    }
}
```

- Code thực hiện thao tác ADD

```
private void btnAddActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    String name = txtHoTen.getText();
    String studentCode = txtMNV.getText();
    String phone = txtSDP.getText();
    String classLearn = txtLopHoc.getText();
    String account = txtAccount.getText();
    String password = txtPassword.getText();
    List<Technician> technicianList = controller.readDataFromFile(fileName: "src/file/technician.txt");
    if (name.isEmpty() || studentCode.isEmpty() || phone.isEmpty() || classLearn.isEmpty() || account.isEmpty() || password.isEmpty()) {
        JOptionPane.showMessageDialog(parentComponent: this, message: "Vui lòng không bỏ trống !", title: "Lỗi", messageType: JOptionPane.OK_OPTION);
    }
    else {
        boolean check = true;
        for (Technician i : technicianList) {
            if (studentCode.equals(i.getStudentCode()) || phone.equals(i.getPhone()) || account.equals(i.getAccountName())) {
                check = false;
            }
        }
        if (!check) {
            JOptionPane.showMessageDialog(parentComponent: this, message: "Technician bị trùng mã sinh viên hoặc số điện thoại hoặc tên Account, vui lòng nhập lại !", title: "Lỗi", messageType: JOptionPane.OK_OPTION);
        }
        else {
            int option = JOptionPane.showConfirmDialog(parentComponent: this, message: "Bạn chắc chắn muốn thêm technician ?", title: "Xác nhận", optionType: JOptionPane.OK_CANCEL_OPTION);
            if (option == JOptionPane.OK_OPTION) {
                technicianList.add(new Technician(name, phone: studentCode, studentCode: phone, classLearn, accountName: account, password: password));
                controller.writeToFile(list: technicianList, fileName: "src/file/technician.txt");
                new Technician_Screen().setVisible(true);
                this.dispose();
            }
        }
    }
}
```

#### 4.4.3. Quản lý Technician \_Show

Hình 2.9: Giao diện Technician\_Show

- Code thực hiện chuyển màn hình từ Technician\_Screen và thực hiện thao tác show

```
private void bntShowActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        // TODO add your handling code here:
        int selectedRow = tblTechnician.getSelectedRow();

        if (selectedRow == -1) {
            JOptionPane.showMessageDialog(parentComponent, this, message: "Vui lòng chọn dòng cần xem.", title: "Error", messageType: JOptionPane.ERROR_MESSAGE);
            throw new Exception(message: "Lỗi chọn dòng");
        } else {
            DefaultTableModel model = (DefaultTableModel) tblTechnician.getModel();

            String hoTen = (String) model.getValueAt(row:selectedRow, column: 1);
            String maSV = (String) model.getValueAt(row:selectedRow, column: 2);
            String dienThoai = (String) model.getValueAt(row:selectedRow, column: 3);
            String lopKhoa = (String) model.getValueAt(row:selectedRow, column: 4) + "-" + (String) model.getValueAt(row:selectedRow, column: 5);

            Technician_Show showTechnician = new Technician_Show();
            showTechnician.displayDetails(hoTen, maSV, dienThoai, lopKhoa);
            showTechnician.setVisible(b: true);
        }
    } catch (Exception ex) {
        ex.printStackTrace();
        // Handle the exception or show an error message to the user.
    }
}
```

```

private static MyController controller = new MyController();
public Technician_Show() {
    String account = controller.readAccountFromFile(fileName: "src/file/account.txt");
    initComponents();
    bntAccout.setText(text: account);
    bntTechnician.setBackground(bg: Color.decode(m: "#fb8500"));
}

public void displayDetails(String hoTen, String maSV, String dienThoai, String lopKhoa) {
    txtHoTen.setText(t: hoTen);
    txtMSV.setText(t: maSV);
    txtSDT.setText(t: dienThoai);
    txtLopKhoa.setText(t: lopKhoa);
}

```

#### 4.4.4. Quản lý Technician \_Edit

The screenshot displays the 'IT Supporter Management System' window. The title bar includes standard window controls. The header bar is black with white text and buttons. The left sidebar contains three buttons: 'Machine' (light blue), 'Technician' (orange), and 'Approval' (light blue). The main content area has a light gray background. At the top of this area is the title 'Edit Technician'. Below it is a form with six rows, each with a label on the left and a text input field on the right. The labels are 'Họ tên', 'Mã sinh viên', 'Điện thoại', 'Lớp - Khoá', 'Account', and 'Password'. The input fields contain the values 'Dinh Van Chinh', '2021605917', '0391130591', 'KHMT1-K16', 'chinhdv', and '123' respectively. At the bottom right of the form area are two buttons: 'COMPLETE' and 'EXIT'.

Hình 2.10: Giao diện Technician\_Edit

- Code thực hiện chuyển màn hình từ Technician\_Screen



```

private void btnEditActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String account = controller.readAccountFromFile(fileName: "src/file/account.txt");
    if ("admin".equals(account)) {
        int selectedRow = tblTechnician.getSelectedRow();
        try {
            if (selectedRow == -1) {
                JOptionPane.showMessageDialog(parentComponent, this, message: "Vui lòng chọn dòng cần chỉnh sửa", title: "Lỗi", messageType: JOptionPane.ERROR_MESSAGE);
                throw new Exception(message: "Lỗi chọn dòng");
            } else {
                DefaultTableModel model = (DefaultTableModel) tblTechnician.getModel();
                String hoTen = (String) model.getValueAt(selectedRow, column: 1);
                String maSV = (String) model.getValueAt(selectedRow, column: 3);
                String dienThoai = (String) model.getValueAt(selectedRow, column: 2);
                String lopKhoa = (String) model.getValueAt(selectedRow, column: 4) + "-" + (String) model.getValueAt(selectedRow, column: 5);
                String accountTech = "", password = "";
                List<Technician> technicianList = controller.readDataFromFile(fileName: "src/file/technician.txt");
                int index = -1;
                int ps = 0;
                for (Technician i : technicianList) {
                    if (maSV.equals(i.getStudentCode())) {
                        accountTech = i.getAccountName();
                        password = i.getPassword();
                        index = ps;
                    }
                    ps++;
                }
                if (index == -1) {
                    JOptionPane.showMessageDialog(parentComponent, this, message: "Không tìm thấy dòng trên trong cơ sở dữ liệu", title: "Lỗi", messageType: JOptionPane.ERROR_MESSAGE);
                    throw new Exception(message: "Lỗi index");
                } else {
                    Technician_Edit showDetailTechnician = new Technician_Edit();
                    showDetailTechnician.displayDetails(selectedRow, hoTen, maSV, dienThoai, lopKhoa, account: accountTech, password);
                    showDetailTechnician.setVisible(is: true);
                }
            }
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    } else {
        JOptionPane.showMessageDialog(parentComponent, this, message: "Bạn không phải admin nên không có quyền chỉnh sửa, vui lòng liên hệ admin để chỉnh sửa bảng technician !", title: "Lỗi", messageType: JOptionPane.OK_OPTION);
    }
}

```

## - Code thực hiện thao tác Edit

```

private void btnCompleteActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String hoTen = txtHoTen.getText();
    String maSV = txtMaSV.getText();
    String dienThoai = txtSDT.getText();
    String lopKhoa = txtLopKhoa.getText();
    String account = txtAccount.getText();
    String password = txtPassword.getText();

    List<Technician> technicianList = controller.readDataFromFile(fileName: "src/file/technician.txt");
    boolean check = true;
    String studentCode = technicianList.get(index).getStudentCode();
    System.out.println(s: index);
    System.out.println(s: "");
    for (Technician technician : technicianList) {
        if (technician.getStudentCode() != studentCode) {
            System.out.println(s: technician.getStudentCode());
            if (technician.getAccountName().equals(account) || technician.getPhone().equals(dienThoai) || technician.getStudentCode().equals(maSV)) {
                check = false;
            }
        }
    }

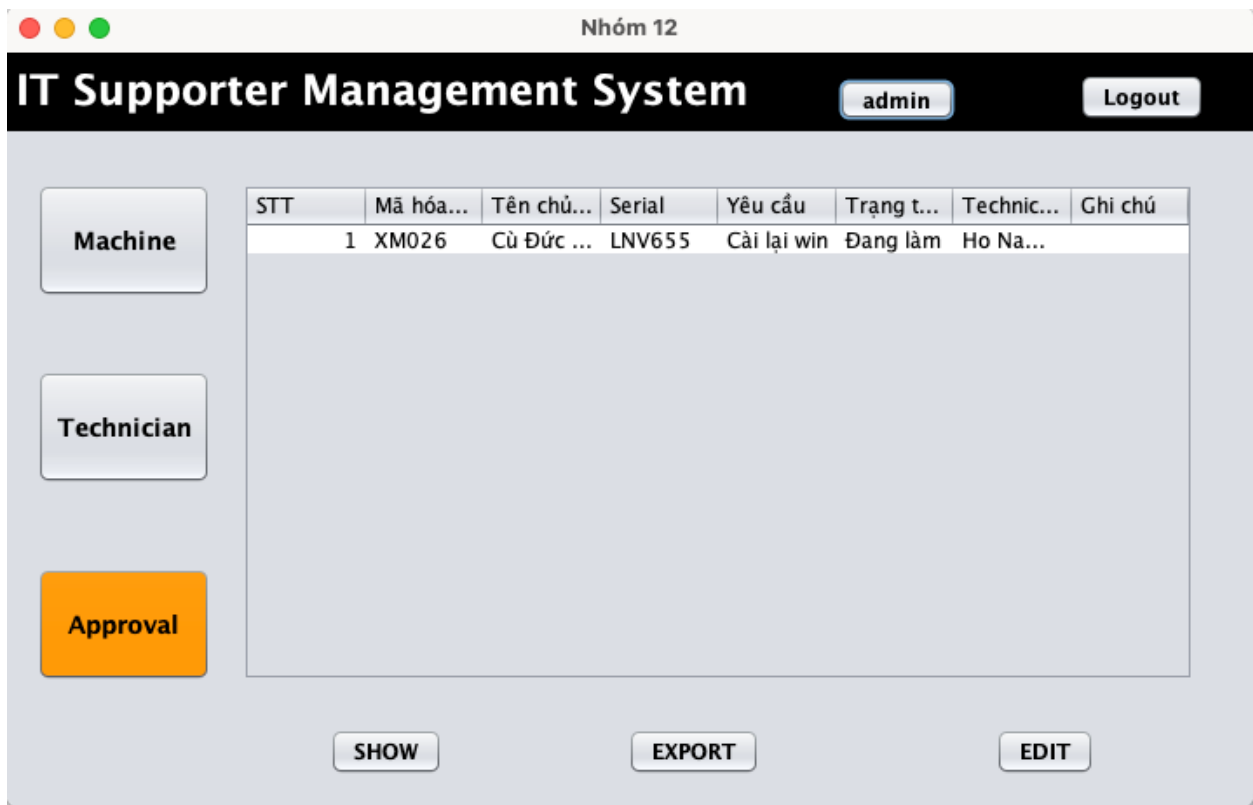
    if (hoTen.isEmpty() || maSV.isEmpty() || dienThoai.isEmpty() || lopKhoa.isEmpty() || account.isEmpty() || password.isEmpty()) {
        JOptionPane.showMessageDialog(parentComponent, this, message: "Hãy điền đủ các thông tin", title: "Error", messageType: JOptionPane.ERROR_MESSAGE);
        return;
    } else if (check == false) {
        JOptionPane.showMessageDialog(parentComponent, this, message: "Mã sinh viên hoặc số điện thoại hoặc tên account đã bị trùng vui lòng nhập lại", title: "Error", messageType: JOptionPane.ERROR_MESSAGE);
    } else {
        Technician newTechnician = new Technician(name: hoTen, phone: dienThoai, studentCode, classRoom: lopKhoa, accountName: account, password: password);

        if (technician.equals(obj: newTechnician)) {
            JOptionPane.showMessageDialog(parentComponent, this, message: "Bạn chưa chỉnh sửa, Vui lòng thực hiện chỉnh sửa !", title: "Lỗi", messageType: JOptionPane.OK_OPTION);
        } else {
            int option = JOptionPane.showConfirmDialog(parentComponent, this, message: "Bạn chắc chắn về thực hiện chỉnh sửa thông tin Technician của mình chứ ? ", title: "Xác nhận", optionType: JOptionPane.OK_CANCEL_OPTION);
            if (option == JOptionPane.OK_OPTION) {
                technicianList.set(index, obj: newTechnician);
                controller.writeToFile(fileName: "src/file/technician.txt");
                new Technician_Screen().setVisible(is: true);
                this.dispose();
            }
        }
    }
}

```

## 4.5. Quản lý Approval – Nguyễn Hoàng Giang

### 4.5.1. Quản lý chung Approval



Hình 2.11: Giao diện quản lý chung

Giao diện màn hình quản lý chung của Approval được thực hiện bằng cơ chế kéo thả các nút. Người dùng có thể lựa chọn các chức năng như: Show, Export, Edit. Ngoài ra, Admin và Technician đều có thể xem thông tin của các hóa đơn được hiển thị trên bảng và logout khi cần thiết.

- Thực hiện bắt lỗi:

```
private void btnShowActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        int selectedRow = tblBill.getSelectedRow();
        if (selectedRow == -1) {
            JOptionPane.showMessageDialog(this, "Please select a record to show details.", "Error", JOptionPane.ERROR_MESSAGE);
        } else {
            List<Bill> listBill = controller.readDataFromFile("src/file/bill.txt");
            DefaultTableModel modelBill = (DefaultTableModel) tblBill.getModel();
            String billID = (String) modelBill.getValueAt(selectedRow, 1);
            for (Bill bill : listBill) {
                if (billID.equals(bill.getBillID())) {
                    Approval_Show approval = new Approval_Show();
                    approval.displayDetails(bill.getNameClient(), bill.getPhoneClient(), bill.getSerialMachine(), bill.getRequirements(), bill.getStatements(), bill.getNameTechnician(), bill.getPhoneTechnician(), bill.getNote());
                    approval.setVisible(true);
                    this.dispose();
                }
            }
        }
    } catch (Exception ex) {
        ex.printStackTrace();
        JOptionPane.showMessageDialog(this, "Đã xảy ra lỗi: " + ex.getMessage(), "Lỗi", JOptionPane.ERROR_MESSAGE);
    }
}
```

```

private void btnExportActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        int selectedRow = tblBill.getSelectedRow();
        if (selectedRow == -1) {
            JOptionPane.showMessageDialog(this, "Please select a record to show details.", "Error", JOptionPane.ERROR_MESSAGE);
        } else {
            List<Bill> listBill = controller.readDataFromFile("src/file/bill.txt");
            DefaultTableModel modelBill = (DefaultTableModel) tblBill.getModel();
            String billID = (String) modelBill.getValueAt(selectedRow, 1);
            for (Bill bill : listBill) {
                if (billID.equals(bill.getBillID())) {
                    ApprovalPrint approval = new ApprovalPrint();
                    approval.displayDetails(bill.getNameClient(), bill.getPhoneClient(), bill.getRequirements(), bill.getStatements(), bill.getNameTechnician(), bill.getPhoneTechnician(), bill.getNote());
                    approval.setVisible(true);
                    this.dispose();
                }
            }
        }
    } catch (Exception ex) {
        ex.printStackTrace();
        JOptionPane.showMessageDialog(this, "Đã xảy ra lỗi: " + ex.getMessage(), "Lỗi", JOptionPane.ERROR_MESSAGE);
    }
}

private void btnEditActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        int selectedRow = tblBill.getSelectedRow();
        if (selectedRow == -1) {
            JOptionPane.showMessageDialog(this, "Bạn chưa chọn hóa đơn để xem", "Lỗi", JOptionPane.ERROR_MESSAGE);
        } else {
            List<Bill> listBill = controller.readDataFromFile("src/file/bill.txt");
            DefaultTableModel modelBill = (DefaultTableModel) tblBill.getModel();
            String billID = (String) modelBill.getValueAt(selectedRow, 1);
            int index = -1;
            for (int i = 0; i < listBill.size(); i++) {
                if (listBill.get(i).getBillID().equals(billID)) {
                    index = i;
                    System.out.println(index);
                    break;
                }
            }
            for (Bill bill : listBill) {
                if (billID.equals(bill.getBillID())) {
                    ApprovalEdit approval = new ApprovalEdit();
                    approval.displayDetails(index, bill.getNameClient(), bill.getPhoneClient(), bill.getSerialMachine(), bill.getRequirements(), bill.getStatements(), bill.getNameTechnician(), bill.getPhoneTechnician(), bill.getNote());
                    approval.setVisible(true);
                    this.dispose();
                }
            }
        }
    } catch (Exception ex) {
        ex.printStackTrace();
        JOptionPane.showMessageDialog(this, "Đã xảy ra lỗi: " + ex.getMessage(), "Lỗi", JOptionPane.ERROR_MESSAGE);
    }
}

```

Ta sử dụng try – catch để xử lý ngoại lệ. Nếu ngoại lệ xảy ra, hệ thống sẽ hiển thị thông báo lên màn hình. Ngược lại nếu không có ngoại lệ, hệ thống sẽ hoạt động bình thường.

#### - Tập hợp:

Hệ thống sử dụng List, một cấu trúc dữ liệu được sử dụng để lưu trữ và quản lý một tập hợp các phần tử. Cho phép thêm phần tử vào cuối danh sách, truy cập vào phần tử tại một vị trí bất kỳ.

```
List<Bill> listBill = controller.readDataFromFile("src/file/bill.txt");
```

#### - Đọc và ghi file

Với hệ thống này, ta sử dụng 1 đối tượng MyController để thực thi interface controller trong package controller. Với bảng ta sẽ đọc từ file “bill.txt” và dữ liệu để hiển thị tên account ta sẽ đọc từ file “account.txt”.

#### + Đọc file

```

public Approval_Screen() {
    List<Bill> listBill = controller.readDataFromFile("src/file/bill.txt");
    String account = controller.readAccountFromFile("src/file/account.txt");
    initComponents();
    setLocationRelativeTo(null);
    setTitle("Nhóm 12");
    bntAccout.setText(account);
    DefaultTableModel modelBill = (DefaultTableModel) tblBill.getModel();
    this.showData(listBill, modelBill);
    bntApproval.setBackground(Color.decode("#fb8500"));
}

```

+ Lưu và chỉnh sửa file:

```

controller.writeToFile(billList, "src/file/bill.txt");

```

#### 4.5.2. Quản lý Approval\_Show

The screenshot displays the 'Approval\_Show' interface within the 'IT Supporter Management System'. The window has a title bar 'Nhóm 12'. The header bar includes the system name and user controls ('admin', 'Logout'). A sidebar on the left contains navigation buttons: 'Machine', 'Technician', and 'Approval' (which is the active screen, highlighted in orange). The main content area, titled 'Show Approval', contains a form with the following data:

Tên chủ máy	Cù Đức Xuân
Điện thoại	0398xxxxxx
Serial	LNV655
Yêu cầu	Cài lại win
Trạng thái	Đang làm
Technician	Ho Nam Tu
SĐT Technician	039190591
Ghi chú	

An 'EXIT' button is located at the bottom right of the form area.

Hình 2.12: Giao diện Approval\_Show

- Code thực hiện chuyển màn hình từ Approval\_Screen và thực hiện thao tác show:

```
private void btnExportActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        int selectedRow = tblBill.getSelectedRow();
        if (selectedRow == -1) {
            JOptionPane.showMessageDialog(this, "Please select a record to show details.", "Error", JOptionPane.ERROR_MESSAGE);
        } else {
            List<Bill> listBill = controller.readDataFromFile("src/file/bill.txt");
            DefaultTableModel modelBill = (DefaultTableModel) tblBill.getModel();
            String billID = (String) modelBill.getValueAt(selectedRow, 1);
            for (Bill bill : listBill) {
                if (billID.equals(bill.getBillID())) {
                    Approval_Print approval = new Approval_Print();
                    approval.displayDetails(bill.getNameClient(), bill.getPhoneClient(), bill.getRequirements(), bill.getStatements(), bill.getNameTechnician(), bill.getPhoneTechnician(), bill.getNote());
                    approval.setVisible(true);
                    this.dispose();
                }
            }
        }
    } catch (Exception ex) {
        ex.printStackTrace();
        JOptionPane.showMessageDialog(this, "Đã xảy ra lỗi: " + ex.getMessage(), "Lỗi", JOptionPane.ERROR_MESSAGE);
    }
}

public void displayDetails(String nameClient, String phoneClient, String serial, String requirements, String statements, String nameTechnician, String phoneTechnician, String notes) {
    txtChuMay.setText(nameClient);
    txtSDT.setText(phoneClient);
    txtSerial.setText(serial);
    txtYeuCau.setText(requirements);
    txtTrangThai.setText(statements);
    txtTech.setText(nameTechnician);
    txtSDTTech.setText(phoneTechnician);
    txtGhiChu.setText(notes);
}
}
```

### 4.5.3. Quản lý Approval\_Edit

The screenshot shows a Java Swing window titled "Nhóm 12" for an "IT Supporter Management System". The window has a dark header bar with the system name and a "Logout" button. On the left side, there is a vertical sidebar with three buttons: "Machine", "Technician", and "Approval" (which is highlighted in orange). The main content area is titled "Edit Approval" and contains a form with the following fields and controls:

- Tên chủ máy:** Text field containing "Củ Đức Xuân".
- Điện thoại:** Empty text field.
- Serial:** Text field containing "LNV655".
- Yêu cầu:** Text field containing "Cài lại win".
- Trạng thái:** Radio button group with four options: "Nhận máy", "Hoàn thành", "Đang làm" (selected), and "Đã trả máy".
- Technician:** Text field containing "Ho Nam Tu".
- SDT Technician:** Text field containing "039190591".
- Ghi chú:** Empty text field.

At the bottom of the form, there are two buttons: "Complete" and "EXIT".

Hình 2.13: Giao diện Approval\_Edit

## - Code chuyển màn hình từ Approval\_Screen sang Approval\_Edit:

```
private void bntEditActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        int selectedRow = tblBill.getSelectedRow();
        if (selectedRow == -1) {
            JOptionPane.showMessageDialog(this, "Bạn chưa chọn hóa đơn để xem", "Lỗi", JOptionPane.ERROR_MESSAGE);
        } else {
            List<Bill> listBill = controller.readDataFromFile("src/file/bill.txt");
            DefaultTableModel modelBill = (DefaultTableModel) tblBill.getModel();
            String billID = (String) modelBill.getValueAt(selectedRow, 1);
            int index = -1;
            for (int i = 0; i < listBill.size(); i++) {
                if (listBill.get(i).getBillID().equals(billID)) {
                    index = i;
                    System.out.println(index);
                    break;
                }
            }
            for (Bill bill : listBill) {
                if (billID.equals(bill.getBillID())) {
                    Approval_Edit approval = new Approval_Edit();
                    approval.displayDetails(index, bill.getNameClient(), bill.getPhoneClient(), bill.getSerialMachine(), bill.getRequirements(), bill.getStatements(), bill.getNameTechnician(), bill.getPhoneTechnician(), bill.getNote());
                    approval.setVisible(true);
                    this.dispose();
                }
            }
        }
    } catch (Exception ex) {
        ex.printStackTrace();
        JOptionPane.showMessageDialog(this, "Đã xảy ra lỗi: " + ex.getMessage(), "Lỗi", JOptionPane.ERROR_MESSAGE);
    }
}
```

```
public Approval_Edit() {
    String account = controller.readAccountFromFile("src/file/account.txt");
    initComponents();
    setLocationRelativeTo(null);
    setTitle("Nhóm 12");
    bntApproval.setBackground(Color.decode("#fb8500"));
}

public void displayDetails(int position, String hoTen, String dienThoai, String serial, String yeuCau, String trangThai, String technician, String dienThoaiTechnician, String ghiChu) {
    // Set the text of your JLabels or JTextFields in this method
    txtTenChuMay.setText(hoTen);
    txtSerial.setText(dienThoai);
    txtSerial.setText(serial);
    txtYeuCau.setText(yeuCau);
    if (trangThai.equalsIgnoreCase("Nhận máy")) {
        radNhanMay.setSelected(true);
    } else if (trangThai.equalsIgnoreCase("Hoàn thành")) {
        radHoanThanh.setSelected(true);
    } else if (trangThai.equalsIgnoreCase("Đang làm")) {
        radDangLam.setSelected(true);
    } else if (trangThai.equalsIgnoreCase("Đã trả máy")) {
        radDaTraMay.setSelected(true);
    }
    txtTech.setText(technician);
    txtSDTTech.setText(dienThoaiTechnician);
    txtGhiChu.setText(ghiChu);
    statements = trangThai;
    index = position;
    serialNum = serial;
}
}
```

## - Code thực hiện thao tác Edit:

```
private void bntCompleteActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    String trangThai = "";
    if (radNhanMay.isSelected()) {
        trangThai = "Nhận máy";
    } else if (radHoanThanh.isSelected()) {
        trangThai = "Hoàn thành";
    } else if (radDangLam.isSelected()) {
        trangThai = "Đang làm";
    } else if (radDaTraMay.isSelected()) {
        trangThai = "Đã trả máy";
    }

    if (trangThai.equals(statements)) {
        JOptionPane.showMessageDialog(this, "Vui lòng chọn trạng thái khác để thực hiện chỉnh sửa hóa đơn", "Lỗi", JOptionPane.OK_OPTION);
    }
    else {
        List<Bill> billList = controller.readDataFromFile("src/file/bill.txt");
        billList.get(index).setStatements(trangThai);
        controller.writeToFile(billList, "src/file/bill.txt");
        JOptionPane.showMessageDialog(this, "Chỉnh sửa hóa đơn thành công", "Thông báo", JOptionPane.OK_OPTION);
        new Approval_Screen().setVisible(true);
        this.dispose();
    }
    List<Machine> listMachine = controller.readDataFromFile("src/file/machine.txt");
    for (Machine machine : listMachine) {
        if (machine.getSerial().equals(serialNum)) {
            machine.setStatements(trangThai);
        }
    }
    controller.writeToFile(listMachine, "src/file/machine.txt");
}
}
```

```
private void btnExitActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    new Approval_Screen().setVisible(true);
    this.dispose();
}
```

#### 4.5.4. Quản lý Approval\_Export

The screenshot shows a Java Swing window titled "Nhóm 12". Inside the window is a form titled "Tech Support". The form consists of several text input fields, each with a label to its left:

- Tên chủ máy: Củ Đức Xuân
- Điện thoại: 0398xxxxxx
- Serial: (empty)
- Yêu cầu: Cài lại win
- Trạng thái: Đang làm
- Technician: Ho Nam Tu
- SĐT Technician: 039190591
- Ghi chú: (empty)

At the bottom of the form, there are two buttons: "EXPORT" and "EXIT".

Hình 2.14: Giao diện Approval\_Export

- Code chuyển màn hình từ Approval\_Screen sang Approval\_Export:

```

private void btnExportActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        int selectedRow = tblBill.getSelectedRow();
        if (selectedRow == -1) {
            JOptionPane.showMessageDialog(this, "Please select a record to show details.", "Error", JOptionPane.ERROR_MESSAGE);
        } else {
            List<Bill> listBill = controller.readDataFromFile("src/file/bill.txt");
            DefaultTableModel modelBill = (DefaultTableModel) tblBill.getModel();
            String billID = (String) modelBill.getValueAt(selectedRow, 1);
            for (Bill bill : listBill) {
                if (billID.equals(bill.getBillID())) {
                    Approval_Export approval = new Approval_Export();
                    approval.displayDetails(bill.getNameClient(), bill.getPhoneClient(), bill.getRequirements(), bill.getStatements(), bill.getNameTechnician(), bill.getPhoneTechnician(), bill.getNote());
                    approval.setVisible(true);
                    this.dispose();
                }
            }
        }
    } catch (Exception ex) {
        ex.printStackTrace();
        JOptionPane.showMessageDialog(this, "Đã xảy ra lỗi: " + ex.getMessage(), "Lỗi", JOptionPane.ERROR_MESSAGE);
    }
}

public Approval_Export() {
    initComponents();
    setTitle("Nhóm 12");
    setLocationRelativeTo(null);
}

public void displayDetails(String nameClient, String phoneClient, String requirements, String statements, String nameTechnician, String phoneTechnician, String notes) {
    txtChuMay.setText(nameClient);
    txtSDT.setText(phoneClient);
    txtYeuCau.setText(requirements);
    txtTrangThai.setText(statements);
    txtTech.setText(nameTechnician);
    txtSDTTech.setText(phoneTechnician);
    txtGhiChu.setText(notes);
}

```

## - Code thực hiện thao tác Export:

```

private void btnExportActionPerformed(java.awt.event.ActionEvent evt) {
    String userName = System.getProperty("user.name");

    JFileChooser fileChooser = new JFileChooser();
    fileChooser.setCurrentDirectory(new java.io.File("C:/Users/" + userName + "/Desktop"));
    fileChooser.setDialogTitle("Chọn nơi lưu file");
    Date currentDate = new Date();

    // Định dạng thời gian thành chuỗi để sử dụng làm tên file
    SimpleDateFormat dateFormat = new SimpleDateFormat("yyyyMMdd_HHmms");

    String fileName = "file_" + dateFormat.format(currentDate) + ".txt";

    // Set tên file mặc định trong hộp thoại lưu
    fileChooser.setSelectedFile(new java.io.File(fileName));

    FileNameExtensionFilter filter = new FileNameExtensionFilter("Tập văn bản (*.txt)", "txt");
    fileChooser.setFileFilter(filter);

    int userSelection = fileChooser.showSaveDialog(this);

    if (userSelection == JFileChooser.APPROVE_OPTION) {
        try (PrintWriter writer = new PrintWriter(fileChooser.getSelectedFile())) {
            writer.println("Tên chủ máy: " + txtChuMay.getText());
            writer.println("Điện thoại: " + txtSDT.getText());
            writer.println("Yêu cầu: " + txtYeuCau.getText());
            writer.println("Trạng thái: " + txtTrangThai.getText());
            writer.println("Technician: " + txtTech.getText());
            writer.println("SBT Technician: " + txtSDTTech.getText());
            writer.println("Ghi chú: " + txtGhiChu.getText());
            JOptionPane.showMessageDialog(this, "Xuất file thành công");
            new Approval_Screen().setVisible(true);
            this.dispose();
        } catch (FileNotFoundException e) {
            JOptionPane.showMessageDialog(this, "Error exporting file: " + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
        }
    }
}

private void txtExitActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    new Approval_Screen().setVisible(true);
    this.dispose();
}

```



### **III. KẾT LUẬN VÀ BÀI HỌC KINH NGHIỆM**

Những kiến thức và kỹ năng học được thông qua Bài tập lớn:

- Cấu trúc của một chương trình java
- Các kiểu dữ liệu và chuyển kiểu dữ liệu
- Các toán tử
- Các cấu trúc điều khiển
- Mảng và xử lý mảng
- Lớp và đối tượng trong java
- Các hàm khởi tạo
- Phương thức tĩnh static
- Nạp chồng phương thức
- Mảng đối tượng
- Kết tập, kế thừa trong java
- Tính trừu tượng, đa hình và interface
- Ghi đè phương thức
- Xử lý ngoại lệ, I/O theo luồng và thao tác với tệp
- Cấu trúc Collection (ArrayList, TreeSet)
- Giao diện Java Swing

## **TÀI LIỆU THAM KHẢO**

- [1]. Giáo trình Lập trình HĐT với Java, Nguyễn Bá Nghiễn, Ngô Văn Bình, Vương Quốc Dũng, Đỗ Sinh Trường; NXB Thống kê, 2020.
- [2]. Bộ slide bài giảng lập trình java- Bộ môn CMPM- trường ĐHCN HN
- [3]. Lập trình hướng đối tượng với Java; Đoàn Văn Ban; NXB Khoa học và Kỹ thuật, Hà Nội 2006 (Tái bản).
- [4]. Lập trình Java nâng cao, Đoàn Văn Ban, NXB Khoa học và Kỹ thuật, Hà Nội 2006
- [5]. The Java Programming Language; Author: K. Arnold, J. Gosling; Published: Addison-Wesley, 1996, ISBN 0-201-63455-4