TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO THÍ NGHIỆM/THỰC NGHIỆM

LẬP TRÌNH JAVA

XÂY DỰNG PHẦN MỀM QUẢN LÝ HOẠT ĐỘNG CLB IT SUPPORTER - HAUI

GVHD: Ths.Vũ Thị Dương

Sinh viên: Nguyễn Hoàng Giang

Hồ Nam Tú

Cù Đức Xuân

Nhóm: 12

Lớp: 20231IT6019001 **Khóa:** 16

Hà Nội - Năm 2023

MỤC LỤC

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: Hiển thị, thêm, sửa, xóa danh sách các máy của khách hàng.
- Quản lý Technician: Hiển thị, thêm, sửa, xóa danh sách thành viên của đội kĩ 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 kĩ 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 kĩ 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 kĩ 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 kĩ thuật CLB và quản lý các hoạt động sự kiện. Cho phép thành viên đội kĩ 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 kĩ 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:

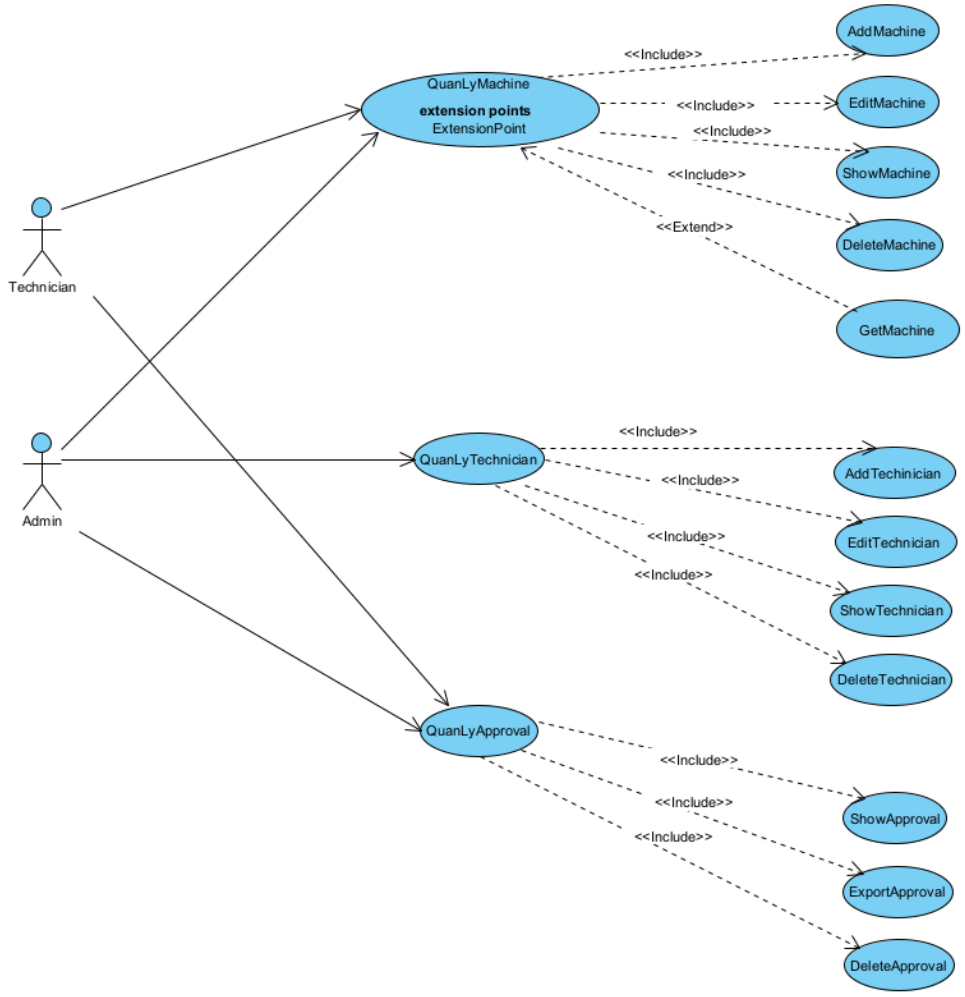
- Technician: Có nhiệm vụ thêm máy cho khách hang, chỉnh sửa thông tin máy, xóa máy và có thể nhận máy nếu muốn.
- Admin: Có nhiệm vụ quản lý technician, bao gồm thêm, sửa, xóa. Ngoài ra admin cũng có thể thực hiện tất cả chức năng của technician.

Các use case và nhiệm vụ:

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

- QuanLyTechnician: 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.
- QuanLyApproval: 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 xem, thêm, sửa, xóa. Cho phép 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 kĩ thuật viên
phone	String	Số điện thoại của kĩ thuật viên
studentCode	String	Mã sinh viên
classLearn	String	Lớp học và khóa hiện tại của kĩ thuật viên
accountName	String	Tên tài khoản kĩ thuật viên khi đăng nhập
password	String	Mật khẩu của kĩ 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 kĩ thuật viên
phoneTechnician	String	Số điện thoại của kĩ thuật viên
account	String	Tài khoản mà kĩ 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 – Hồ Nam Tú

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 – Hồ Nam Tú

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 tại vị trí xảy ra lỗi
  + Bắt lỗi, gom rác chưa điền thông tin đăng nhập

```
try {
    if (account.isEmpty() || password.isEmpty()) {
        JOptionPane.showMessageDialog(parentComponent: this,
                message: "Bạn cần nhập đầy đủ ID và Password", title: "Message",
                messageType: JOptionPane.WARNING_MESSAGE);
        throw new Exception(message: "Loi bo trong thong tin");
```

+ Bắt lỗi, gom rác 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.writeAccoutToFile(accout: 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);
        throw new Exception(message: "Loi dang nhap");
    }
}
```

- 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 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.writeAccoutToFile(accout: 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 – Hồ Nam Tú

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 tại vị trí xảy ra lỗi
  + Bắt lỗi, gom rác chưa chọn máy để thực hiện thao tác

```
int selectedRow = tblMachine.getSelectedRow();
try {
    if (selectedRow == -1) {
        JOptionPane.showMessageDialog(parentComponent: this,
                message: "Hãy chọn vào máy muốn xem", title: "Lỗi",
                messageType: JOptionPane.ERROR_MESSAGE);
        throw new Exception(message: "Loi chon dong");

    } else {
```

```
int selectedRow = tblMachine.getSelectedRow();
try {
    if (selectedRow == -1) {
        JOptionPane.showMessageDialog(parentComponent: this
                , message: "Hãy chọn máy muốn sửa thông tin",
                title: "Lỗi", messageType: JOptionPane.ERROR_MESSAGE);
        throw new Exception(message: "Loi chon dong");
    } else {
```

```
    } else if (indexDelete == -1) {
        JOptionPane.showMessageDialog(parentComponent: this,
                message: "Chưa chọn máy cần xóa", title: "Lỗi",
                messageType: JOptionPane.ERROR_MESSAGE);
        throw new Exception(message: "Loi chon dong");
```

- Bắt lỗi tại đối tượng Machine

```
public void setRequirements(String requirements) {
    try{
        if(requirements.isEmpty()){
            throw new Exception(message:"Loi trong yeu cau");
        }
        this.requirements = requirements;
    }
    catch(Exception ex){
        ex.printStackTrace();
    }

}
```

```
public void setSerial(String serial) {
    try{
        if(serial.isEmpty()){
            throw new Exception(message:"Loi bo trong serial");
        }
        this.serial = serial;
    }
    catch(Exception ex){
        ex.printStackTrace();
    }
}
```

```
private void errorInitMachine(String requirements, String serial){
    try{
        if(requirements.isEmpty()){
            throw new Exception(message:"Loi bo trong yeu cau");
        }
        if(serial.isEmpty()){
            throw new Exception(message:"Loi bo trong serial");
        }
    }
    catch(Exception ex){
        ex.printStackTrace();
    }
}
```

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

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

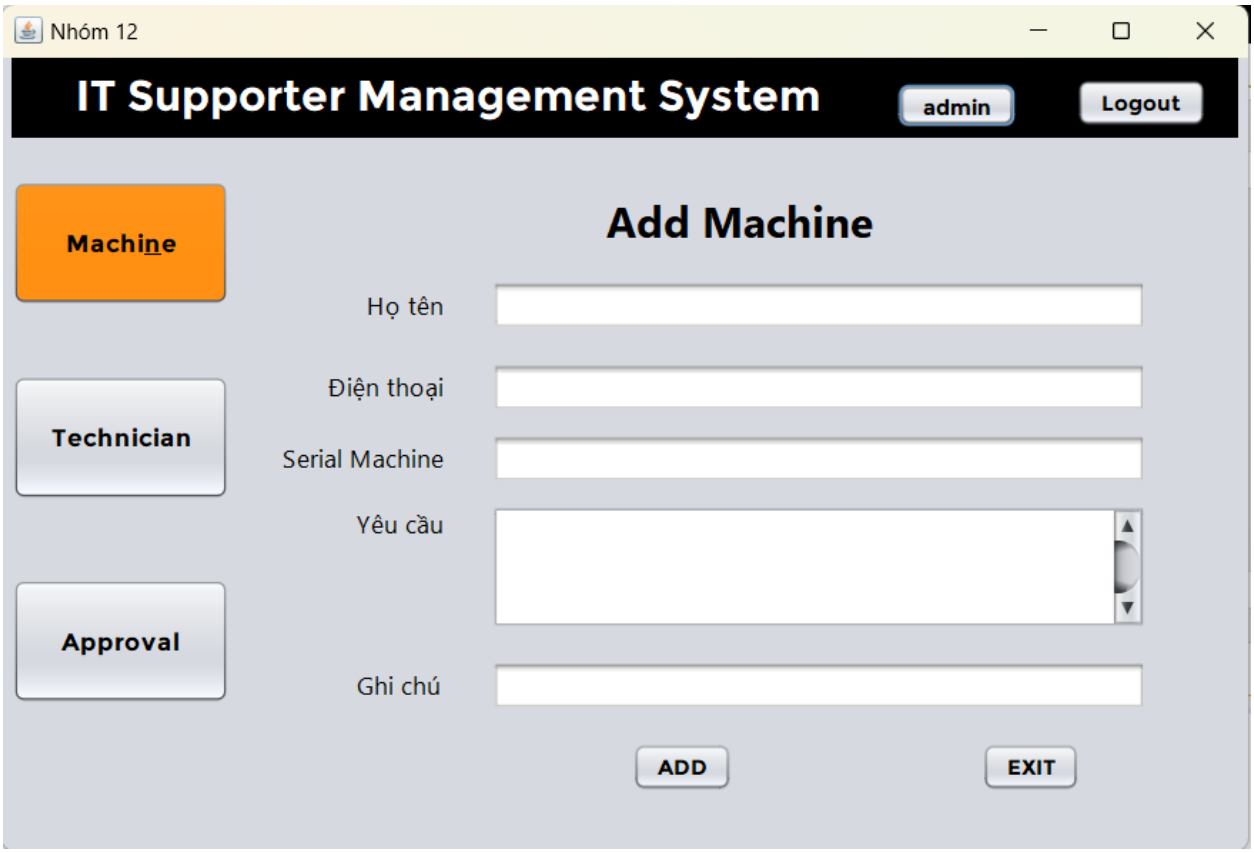
- Thao tác với file

  + 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



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 Techinician hoặc Approval. Nếu muốn thoát ấn EXIT sẽ trở về màn hình quản lý Machine.

- Hướng đối tượng

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

- Bắt lỗi và gom rác

```
try {
    if (hoTen.isEmpty()) {
        JOptionPane.showMessageDialog(parentComponent: this,
                message: "Không được để trống họ tên", title: "Lỗi",
                messageType: JOptionPane.OK_OPTION);
        throw new Exception(message: "Loi trong ten");
    } else if (SDT.isEmpty()) {
        JOptionPane.showMessageDialog(parentComponent: this,
                message: "Không được để trống số điện thoại", title: "Lỗi",
                messageType: JOptionPane.OK_OPTION);
        throw new Exception(message: "Loi trong so dien thoai");
    } else if (yeuCau.isEmpty()) {
        JOptionPane.showMessageDialog(parentComponent: this,
                message: "Không được để trống yêu cầu",
                title: "Lỗi", messageType: JOptionPane.OK_OPTION);
        throw new Exception(message: "Loi trong yeu cau");
    } else if (serial.isEmpty()) {
        JOptionPane.showMessageDialog(parentComponent: this,
                message: "Không được để trống số sê - ri", title: "Lỗi",
                messageType: JOptionPane.OK_OPTION);
        throw new Exception(message: "Loi trong serial");
    } 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);
        throw new Exception(message: "Loi trung serial");
    } else {
```

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.

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.writeToFile(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.writeToFile(list: machineList, fileName: "src/file/Machine.txt");
}
```

4.3.4. Quản lý Machine_Show

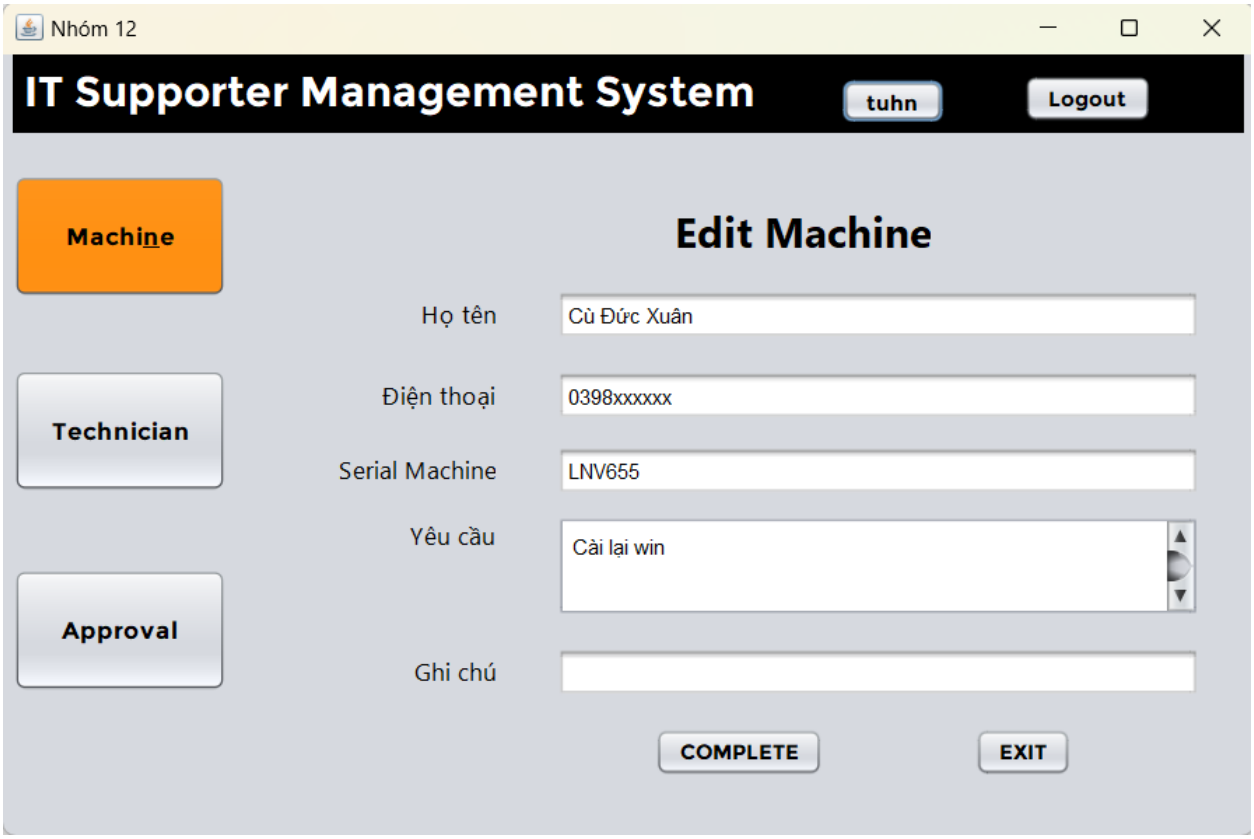
Hình 2.4: Màn hình Machine_Show

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



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 Technian có thể Logout khi cần thiết.

- Bắt lỗi và gom rác tại vị trí bị lỗi

```
try {
    if(hoTen.isEmpty() || dienThoai.isEmpty() || serial.isEmpty() || yeuCau.isEmpty()){
        JOptionPane.showMessageDialog(parentComponent: this,
                message: "Vui lòng không bỏ trống các thông tin cần thiết ! ",
                title:"Lỗi", messageType:JOptionPane.ERROR_MESSAGE);
        throw new Exception(message: "Loi bo trong thong tin");
    }
    if (machine.equals(obj: newMachine)) {
        JOptionPane.showMessageDialog(parentComponent: this,
                "Bạn chưa chỉnh sửa thông tin, vui lòng nhấn chỉnh sửa"
                        + " thông tin bạn mong muốn ! ",
                title:"Lỗi", messageType:JOptionPane.ERROR_MESSAGE);
        throw new Exception(message: "Loi chua chinh sua thong tin");
    } else {
```

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.

- Thao tác với file

  + Đọc từ file

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

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

  + Ghi vào file

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

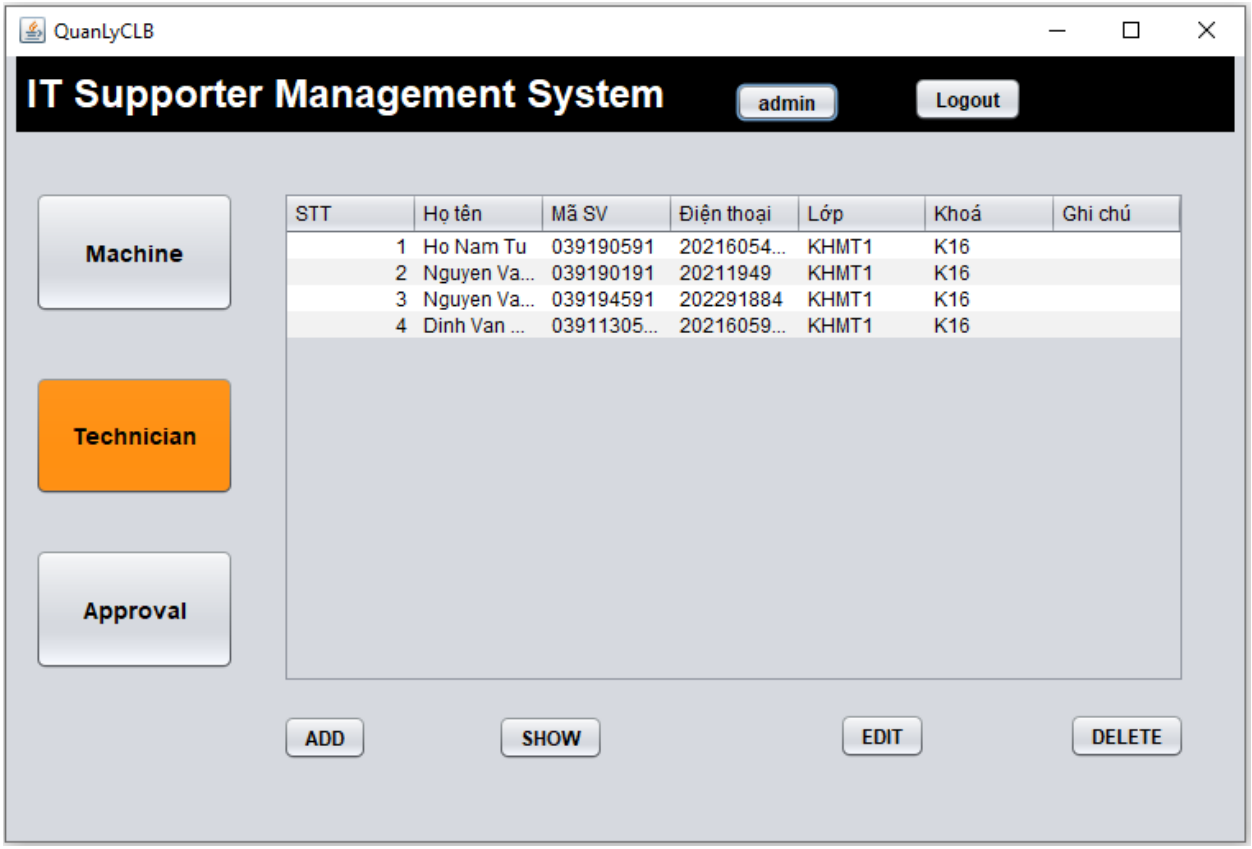
- Collection

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

```
if (option == JOptionPane.OK_OPTION) {
    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:

+ Bắt lỗi và thu gom rác tại vị trí xảy ra lỗi

```
// TODO add your handling code here:
String account = controller.readAccountFromFile(fileName: "src/file/account.txt");
if ("admin".equals(anObject: 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: "Loi chon dong");
        } 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(aValue: 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 {
                } else {
                    JOptionPane.showMessageDialog(parentComponent: this,
                            message: "Dòng bạn chọn đang trống",
                            title: "Lỗi", messageType: JOptionPane.ERROR_MESSAGE);
                    throw new Exception(message: "Dong chon loi");
                }
            }
        }
    } catch (Exception ex) {
        ex.printStackTrace();
    }
} else {
    JOptionPane.showMessageDialog(parentComponent: null,
            "Tài khoản của bạn không có quyền thêm technician,"
                    + " vui lòng liên hệ admin để thêm ! ",
            title: "Lỗi", messageType: JOptionPane.OK_OPTION);
}
```

+ Bắt lỗi tại đối tượng People

```
public void errorInit(String name, String phone){
    try{
        if(name.isEmpty()){
            throw new Exception(message:"Loi bo trong ten");
        }
        if(phone.isEmpty()){
            throw new Exception(message:"Loi bo trong so dien thoai");
        }
        if(!isNumeric(str:phone)){
            throw new Exception(message:"Loi ki tu trong so dien thoai");
        }
    }
    catch(Exception ex){
        ex.printStackTrace();
    }
}
```

```
public void errorName(String name){
    try{
        if(name.isEmpty()){
            throw new Exception(message: "Loi bo trong ten");
        }
    }
    catch(Exception ex){
        ex.printStackTrace();
    }
}

public void errorPhone(String phone){
    try{
        if(phone.isEmpty()){
            throw new Exception(message: "Loi bo trong so dien thoai");
        }
        if(!isNumeric(str: phone)){
            throw new Exception(message: "Loi ki tu trong so dien thoai");
        }
    }
    catch(Exception ex){
        ex.printStackTrace();
    }
}
```

+ Bắt lỗi tại đối tượng Technician

```
public void setPassWord(String passWord) {
    try{
        if(passWord.isEmpty()){
            throw new Exception(message: "Loi bo trong password");
        }
        this.passWord = passWord;
    }
    catch(Exception ex){
        ex.printStackTrace();
    }
}
```

```
public void setClassLearn(String classLearn) {
    try{
        if(classLearn.isEmpty()){
            throw new Exception(message: "Loi bo trong lop - khoa");
        }
        this.classLearn = classLearn;
    }
    catch(Exception ex){
        ex.printStackTrace();
    }
}
```

```
public void setStudentCode(String studentCode) {
    try{
        if(studentCode.isEmpty()){
            throw new Exception(message: "Loi bo trong password");
        }
        if(!super.isNumeric(str: studentCode)){
            throw new Exception(message: "Loi ki tu trong ma sinh vien");
        }
        this.studentCode = studentCode;
    }
    catch(Exception ex){
        ex.printStackTrace();
    }

}


public void setAccountName(String accountName) {

    try{
        if(accountName.isEmpty()){
            throw new Exception(message: "Loi bo trong ten account");
        }

        this.accountName = accountName;
    }
    catch(Exception ex){
        ex.printStackTrace();
    }

}
```

```
private void errorInitTech(String studentCode, String classLearn,
        String accountName, String passWord){
    try{
        if(studentCode.isEmpty()){
            throw new Exception(message: "Loi bo trong ma sinh vien");
        }
        if(classLearn.isEmpty()){
            throw new Exception(message: "Loi bo trong thong tin lop - khoa");
        }
        if(accountName.isEmpty()){
            throw new Exception(message: "Loi bo trong ten account");
        }
        if(passWord.isEmpty()){
            throw new Exception(message: "Loi bo trong password");
        }
        if(!super.isNumeric(str: studentCode)){
            throw new Exception(message: "Loi ki tu trong ma sinh vien");
        }
    }
    catch(Exception ex){
        ex.printStackTrace();
    }
}
```

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 kì.

```
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 hiện 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");
```

+ Lưu và chỉnh sửa file

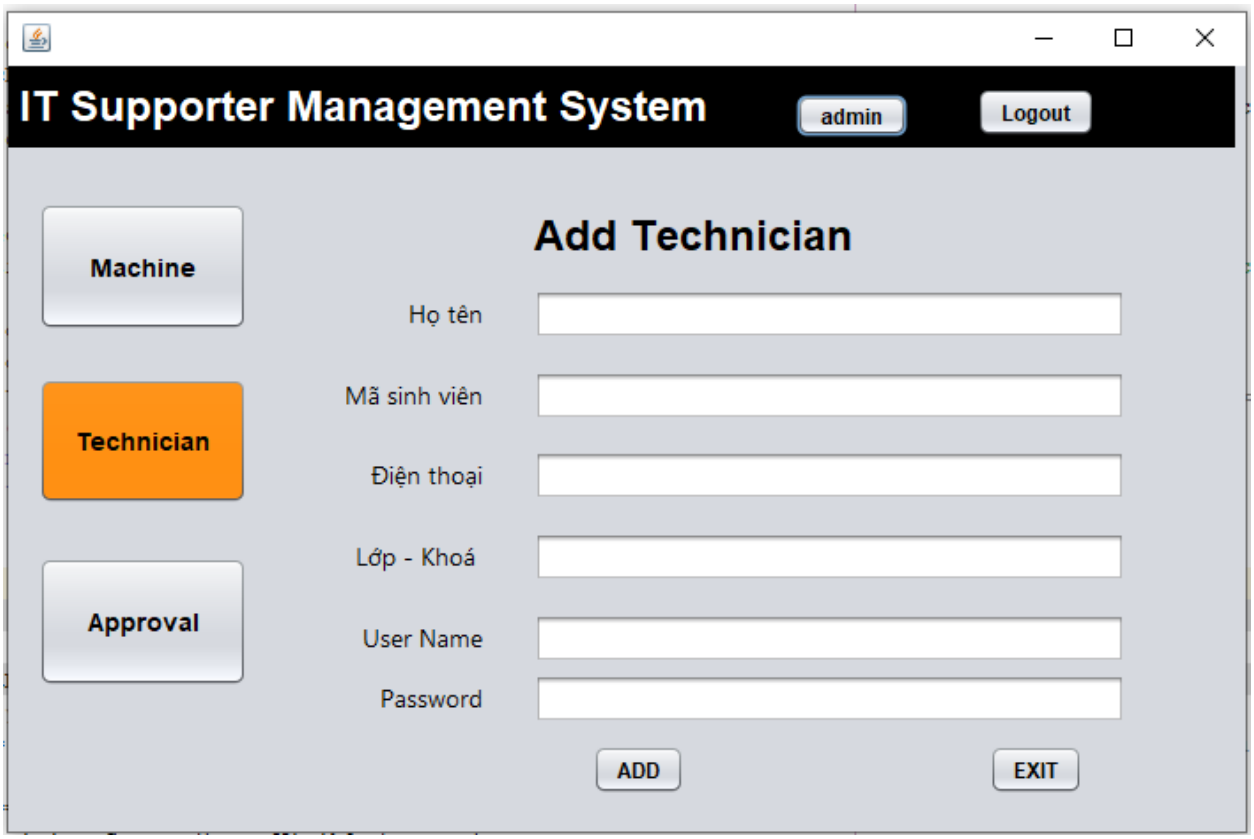
```
                ,
                technicianList.remove(index: selectedRow);
                controller.writeToFile(list: technicianList,
                        fileName: "src/file/technician.txt");
```

- 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(anObject: 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: "Loi chon dong");
                } else {

                } 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(aValue: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 {
```

```
                } else {
                    JOptionPane.showMessageDialog(parentComponent: this,
                            message: "Dòng bạn chọn đang trống",
                            title: "Lỗi", messageType: JOptionPane.ERROR_MESSAGE);
                    throw new Exception(message: "Dong chon loi");
                }
            }
        }
    } catch (Exception ex) {
        ex.printStackTrace();
    }
```

4.4.2. Quản lý Technician_Add



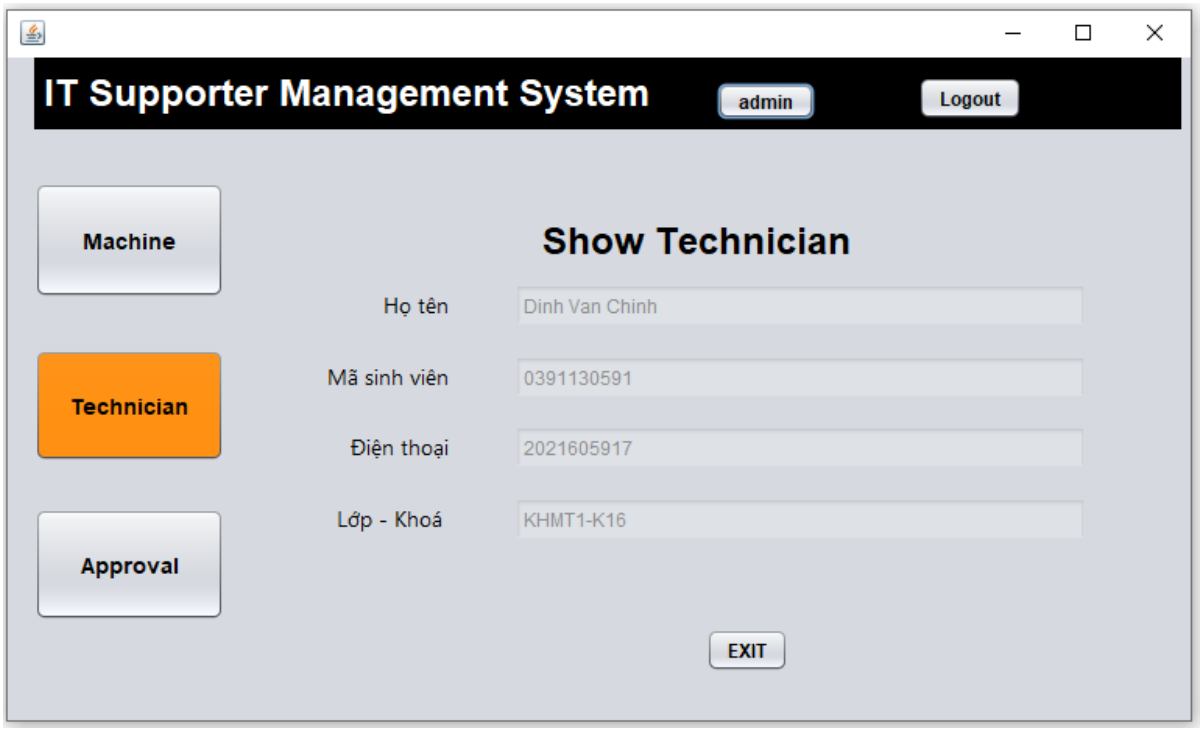
Hình 2.8: Giao diện Technician_Add

- Code thực hiện thao tác ADD

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

    String name = txtHoTen.getText();
    String studentCode = txtMSV.getText();
    String phone = txtSDT.getText();
    String classLearn = txtLopKhoa.getText();
    String account = txtAccout.getText();
    String password = txtPassword.getText();
    List<Technician> technicianList = controller.readDataFromFile
        (fileName: "src/file/technician.txt");
    try {
        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);
            throw new Exception(message: "Loi bo trong thong tin");
        } else {
            boolean check = true;
            for (Technician i : technicianList) {
                if (studentCode.equals(anObject: i.getStudentCode())
                        || phone.equals(anObject: i.getPhone())
                        || account.equals(anObject: i.getAccountName())) {
                    check = false;
                }
            }
```

```
            if (!check) {
                JOptionPane.showMessageDialog(parentComponent: this,
                        "Technician bị trùng mã sinh viên "
                                + "hoặc số điện thoại hoặc tên Account,"
                                + " vui lòng nhập lại ! ", title:"Lỗi",
                                messageType:JOptionPane.OK_OPTION);
                throw new Exception("Loi trung ma sinh vien "
                        + "hoac so dien thoai hoac ten account");
            } 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,
                            classLearn, accountName:account, passWord: password));
                    controller.writeToFile(list: technicianList,
                            fileName: "src/file/technician.txt");
                    new Technician_Screen().setVisible(b:true);
                    this.dispose();
                }
            }
        }
    } catch (Exception ex) {
        ex.printStackTrace();
    }
```

4.4.3. Quản lý Technician _Show



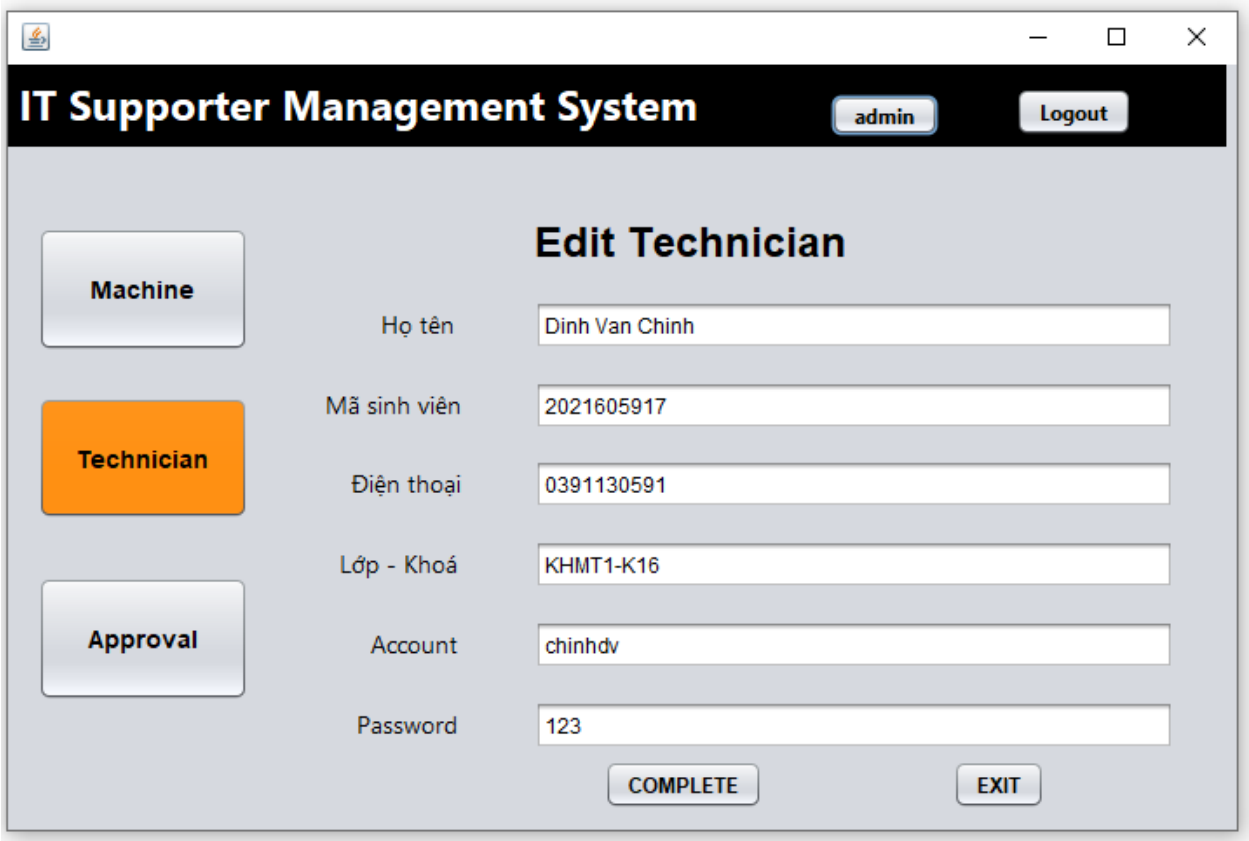
Hình 2.9: Giao diện Technician_Show

- 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: "Loi chon dong");
        } 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(nm: "#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



Hình 2.10: Giao diện Technician_Edit

- Code thực hiện chuyển màn hình từ Technician_Screen

```
private void bntEditActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String account = controller.readAccountFromFile(fileName: "src/file/account.txt");
    if ("admin".equals(anObject: 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: "Loi chon dong");
            } 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);
                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(anObject: i.getStudentCode())) {
                        accountTech = i.getAccountName();
                        password = i.getPassWord();
                        index = ps;
                    }
                    ps++;
                }
                if (index == -1) {
                ,
                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: "Loi index");
                } else {
                    Technician_Edit showDetailTechnician = new Technician_Edit();
                    showDetailTechnician.displayDetails(possition: index, hoTen, maSV,
                            dienThoai, lopKhoa, account: accountTech, password);
                    showDetailTechnician.setVisible(b: true);
                }
            }
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    } else {
        JOptionPane.showMessageDialog(parentComponent: this,
                "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);
    }
```

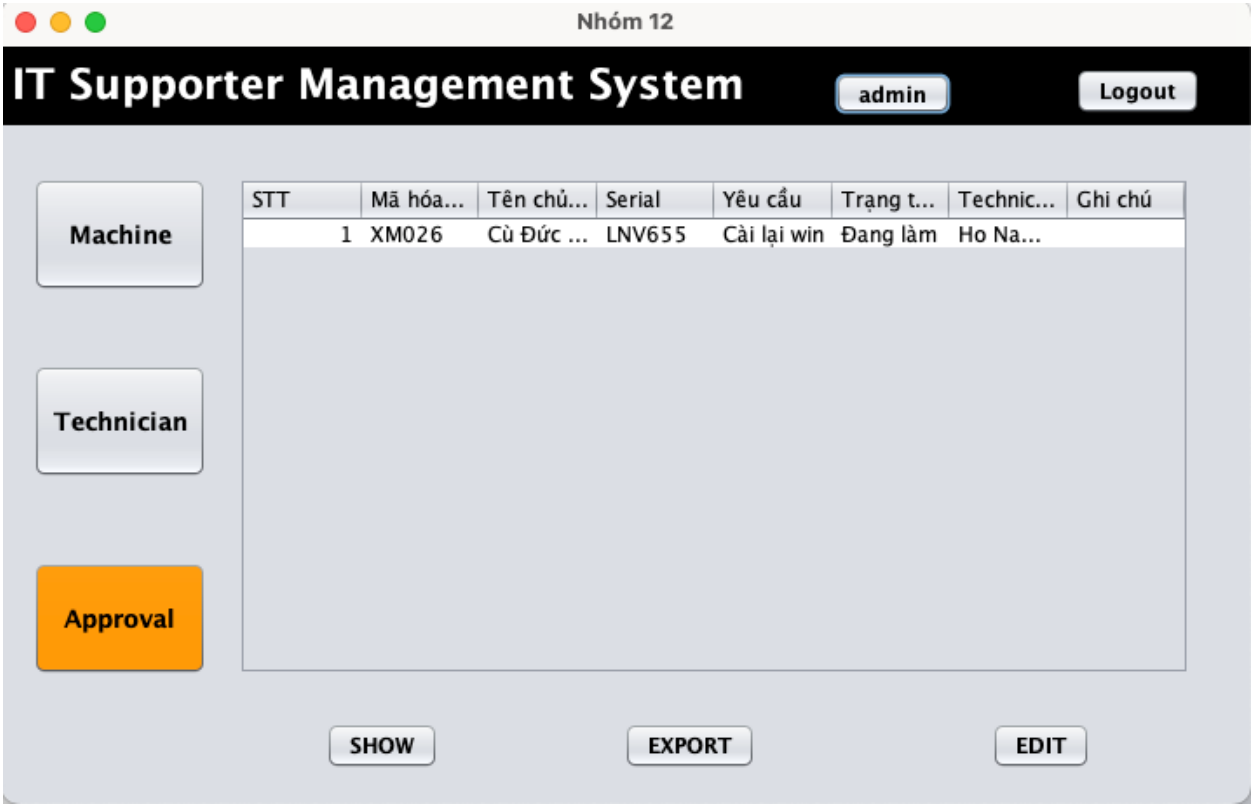
- Code thực hiện thao tác Edit

```
private void bntCompleteActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String hoTen = txtHoTen.getText();
    String maSV = txtMSV.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(x: index);
    System.out.println(x: "");
    for (Technician technician : technicianList) {
        if (technician.getStudentCode() != studentCode) {
            System.out.println(x: technician.getStudentCode());
            if (technician.getAccountName().equals(anObject: account)
                    || technician.getPhone().equals(anObject: dienThoai)
                    || technician.getStudentCode().equals(anObject: maSV)) {

                check = false;
            }
        }

    }
    }
    try {
        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);
            throw new Exception(message: "Loi thieu thong tin");
        } else if (check == false) {
            JOptionPane.showMessageDialog(parentComponent: this,
                    "Mã sinh viên hoặc số điện thoại "
                        + "hoặc tên accout đã bị trùng vui lòng nhập lại",
                    title: "Error", messageType: JOptionPane.ERROR_MESSAGE);
            throw new Exception(message: "Loi trung ma sinh vien, so dien thoai hoac account");

        } else {
            Technician newTechnician = new Technician(name: hoTen,
                    phone: dienThoai, studentCode, classLearn: 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);
                throw new Exception(message: "Loi chua thuc hien chinh sua");
            } else {
```

```
            } else {
                int option = JOptionPane.showConfirmDialog(parentComponent: this,
                        "Bạn chắc chắn về thực hiện chỉnh sửa thông tin "
                                + "Technician của mình chú ? ",
                        title: "Xác nhận", optionType: JOptionPane.OK_CANCEL_OPTION);
                if (option == JOptionPane.OK_OPTION) {
                    technicianList.set(index, element:newTechnician);
                    controller.writeToFile(list:technicianList,
                            fileName:"src/file/technician.txt");
                    new Technician_Screen().setVisible(b:true);
                    this.dispose();
                }
            }
        }
    }
    catch(Exception ex){
        ex.printStackTrace();
    }

}
```

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 bntShowActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    int selectedRow = tblBill.getSelectedRow();
    try {
        if (selectedRow == -1) {
            JOptionPane.showMessageDialog(parentComponent: this,
                    message: "Please select a record to show details.",
                    title:"Error", messageType:JOptionPane.ERROR_MESSAGE);
            throw new Exception(message: "Loi chon dong");
        } else {
            List<Bill> listBill = controller.readDataFromFile(fileName: "src/file/bill.txt");
            DefaultTableModel modelBill = (DefaultTableModel) tblBill.getModel();
            String billID = (String) modelBill.getValueAt(row: selectedRow, column: 1);
            for (Bill bill : listBill) {
                if (billID.equals(anObject: bill.getBillID())) {
                    Approval_Show approval = new Approval_Show();
                    approval.displayDetails(nameClient: bill.getNameClient(),
                            phoneClient:bill.getPhoneClient(), serial:bill.getSerialMachine(),
                            requirements:bill.getRequirements(), statements: bill.getStatements(),
                            nameTechnician: bill.getNameTechnician(),
                            phoneTechnician: bill.getPhoneTechnician(), notes:bill.getNote());
                    approval.setVisible(b: true);
                    this.dispose();
                }
            }
        }
    } catch (Exception ex) {
        ex.printStackTrace();
    }
}
```

```
    private void bntExportActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        int selectedRow = tblBill.getSelectedRow();
        try {
            if (selectedRow == -1) {
                JOptionPane.showMessageDialog(parentComponent: this,
                        message: "Please select a record to show details.",
                        title:"Error", messageType:JOptionPane.ERROR_MESSAGE);
                throw new Exception(message: "Loi chon dong");
            } else {
                List<Bill> listBill = controller.readDataFromFile(fileName: "src/file/bill.txt");
                DefaultTableModel modelBill = (DefaultTableModel) tblBill.getModel();
                String billID = (String) modelBill.getValueAt(row: selectedRow, column:1);
                for (Bill bill : listBill) {
                    if (billID.equals(anObject: bill.getBillID())) {
                        Approval_Print approval = new Approval_Print();
                        approval.displayDetails(nameClient: bill.getNameClient(),
                                phoneClient:bill.getPhoneClient(), requirements:bill.getRequirements(),
                                statements: bill.getStatements(), nameTechnician: bill.getNameTechnician(),
                                phoneTechnician: bill.getPhoneTechnician(), notes:bill.getNote());
                        approval.setVisible(b: true);
                        this.dispose();
                    }
                }
            }
        }
        catch(Exception ex){
            ex.printStackTrace();
        }

    }
    private void bntEditActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        int selectedRow = tblBill.getSelectedRow();
        try {
            if (selectedRow == -1) {
                JOptionPane.showMessageDialog(parentComponent: this,
                        message: "Bạn chưa chọn hóa đơn để xem", title:"Lỗi",
                        messageType:JOptionPane.ERROR_MESSAGE);
            } else {
                List<Bill> listBill = controller.readDataFromFile(fileName: "src/file/bill.txt");
                DefaultTableModel modelBill = (DefaultTableModel) tblBill.getModel();
                String billID = (String) modelBill.getValueAt(row: selectedRow, column:1);
                int index = -1;
                for (int i = 0; i < listBill.size(); i++) {
                    if (listBill.get(index:i).getBillID().equals(anObject: billID)) {
                        index = i;
                        System.out.println(x: index);
                        break;
                    }
                }
```

```
            if (index != -1) {
                for (Bill bill : listBill) {
                    if (billID.equals(anObject: bill.getBillID())) {
                        Approval_Edit approval = new Approval_Edit();
                        approval.displayDetails(possition: index, hoTen:bill.getNameClient(),
                                dienThoai: bill.getPhoneClient(), serial:bill.getSerialMachine(),
                                yeuCau:bill.getRequirements(), trangThai: bill.getStatements(),
                                technician: bill.getNameTechnician(),
                                dienThoaiTechnician: bill.getPhoneTechnician(), ghiChu:bill.getNote());
                        approval.setVisible(b: true);
                        this.dispose();
                    }
                }
            } else {
                JOptionPane.showMessageDialog(parentComponent: this,
                        message: "Dòng được chọn không có trong cơ sở dữ liệu",
                        title:"Lỗi", messageType:JOptionPane.ERROR_MESSAGE);
                throw new Exception(message: "Khong ton tai dong duoc chon trong co so du lieu");
            }
        }
    } catch (Exception ex) {
        ex.printStackTrace();

    }

}
```

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 kì.

```
List<Bill> listBill = controller.readDataFromFile(fileName: "src/file/bill.txt");
DefaultTableModel modelBill = (DefaultTableModel) tblBill.getModel();
```

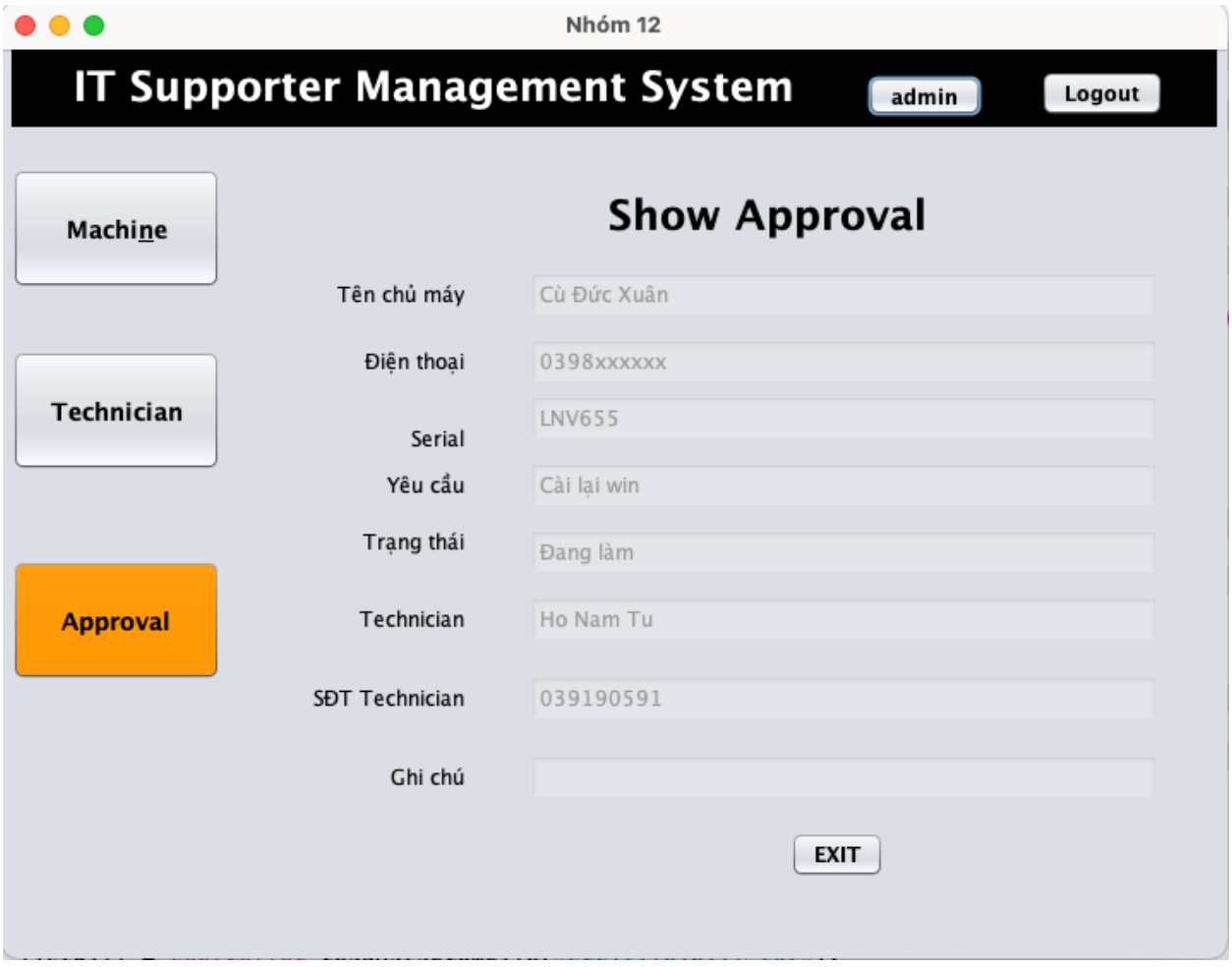
- Đọc và ghi file

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 hiện thị tên account ta sẽ đọc từ file “account.txt”.

+ Đọc file

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

4.5.2. Quản lý Approval_Show



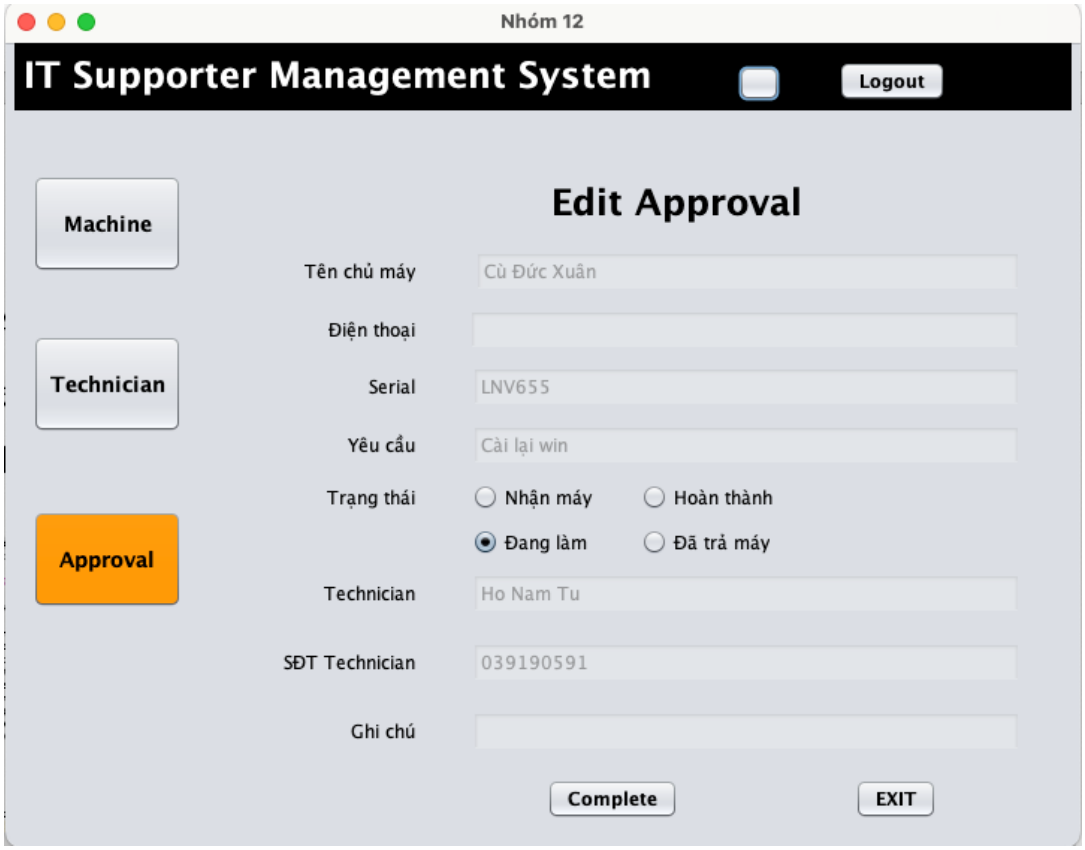
Hình 2.12: Giao diện Approval_Show

- Code thực hiện chuyển màn hình từ Approval_Screen và thực hiện thao tác show:

```
private void bntShowActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    int selectedRow = tblBill.getSelectedRow();
    try {
        if (selectedRow == -1) {
            JOptionPane.showMessageDialog(parentComponent: this,
                    message: "Please select a record to show details.",
                    title:"Error", messageType:JOptionPane.ERROR_MESSAGE);
            throw new Exception(message: "Loi chon dong");
        } else {
            List<Bill> listBill = controller.readDataFromFile(fileName: "src/file/bill.txt");
            DefaultTableModel modelBill = (DefaultTableModel) tblBill.getModel();
            String billID = (String) modelBill.getValueAt(row: selectedRow, column: 1);
            for (Bill bill : listBill) {
                if (billID.equals(anObject: bill.getBillID())) {
                    Approval_Show approval = new Approval_Show();
                    approval.displayDetails(nameClient: bill.getNameClient(),
                            phoneClient:bill.getPhoneClient(), serial:bill.getSerialMachine(),
                            requirements:bill.getRequirements(), statements: bill.getStatements(),
                            nameTechnician: bill.getNameTechnician(),
                            phoneTechnician: bill.getPhoneTechnician(), notes:bill.getNote());
                    approval.setVisible(b: true);
                    this.dispose();
                }
            }
        }
    } catch (Exception ex) {
        ex.printStackTrace();
    }
}
```

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

4.5.3. Quản lý Approval_Edit



Hình 2.13: Giao diện Approval_Edit

- Code chuyển màn hình từ Approval_Screen sang Approval_Edit và hiển thị thông tin lên màn hình Edit:

```
private void bntEditActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    int selectedRow = tblBill.getSelectedRow();
    try {
        if (selectedRow == -1) {
            JOptionPane.showMessageDialog(parentComponent: this,
                    message: "Bạn chưa chọn hóa đơn để xem", title:"Lỗi",
                    messageType:JOptionPane.ERROR_MESSAGE);
        } else {
            List<Bill> listBill = controller.readDataFromFile(fileName: "src/file/bill.txt");
            DefaultTableModel modelBill = (DefaultTableModel) tblBill.getModel();
            String billID = (String) modelBill.getValueAt(row: selectedRow, column:1);
            int index = -1;
            for (int i = 0; i < listBill.size(); i++) {
                if (listBill.get(index:i).getBillID().equals(anObject: billID)) {
                    index = i;
                    System.out.println(x: index);
                    break;
                }
            }
```

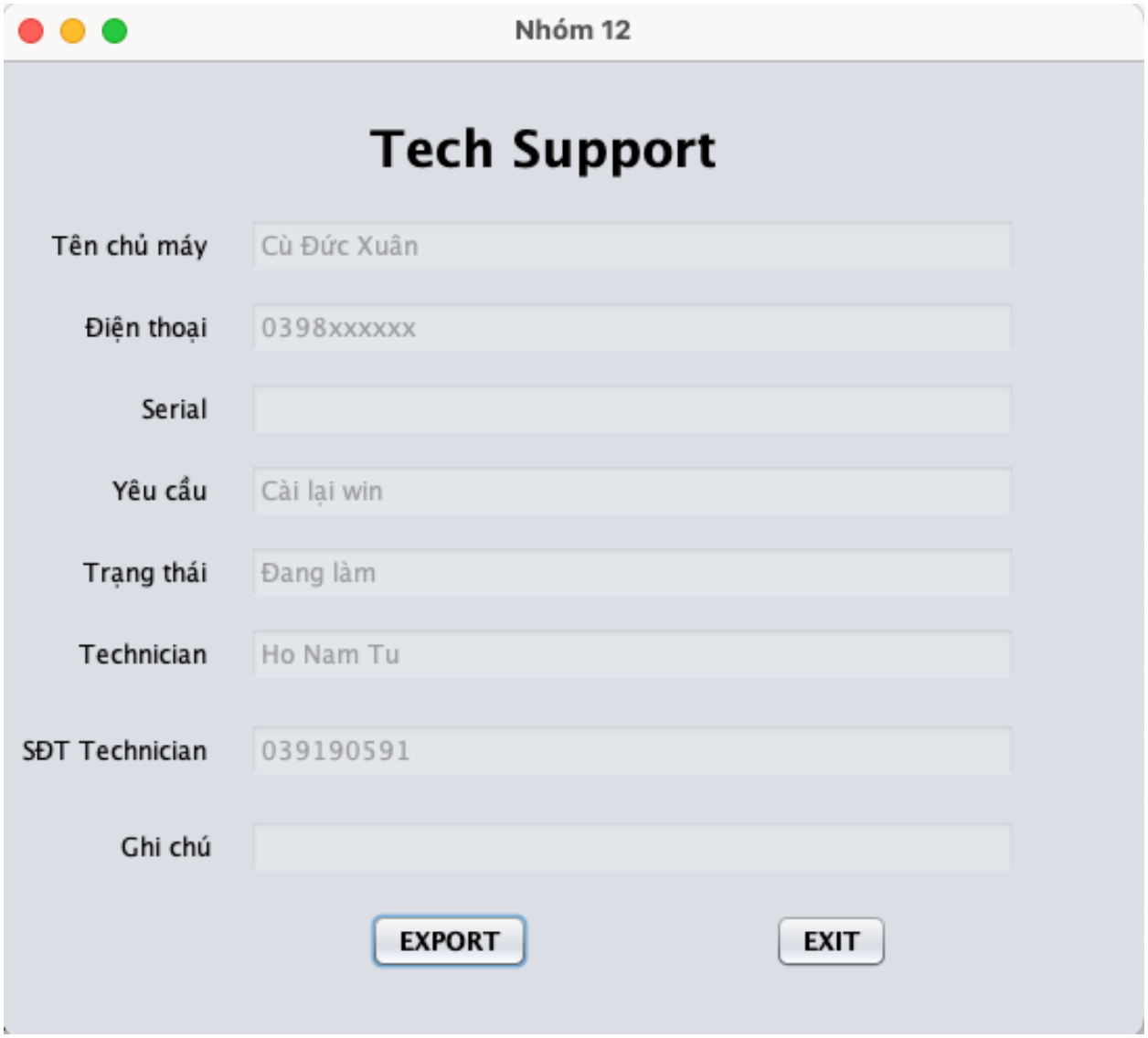
```
            if (index != -1) {
                for (Bill bill : listBill) {
                    if (billID.equals(anObject: bill.getBillID())) {
                        Approval_Edit approval = new Approval_Edit();
                        approval.displayDetails(possition: index, hoTen:bill.getNameClient(),
                                dienThoai: bill.getPhoneClient(), serial:bill.getSerialMachine(),
                                yeuCau:bill.getRequirements(), trangThai: bill.getStatements(),
                                technician: bill.getNameTechnician(),
                                dienThoaiTechnician: bill.getPhoneTechnician(), ghiChu:bill.getNote());
                        approval.setVisible(b: true);
                        this.dispose();
                    }
                }
            } else {
                JOptionPane.showMessageDialog(parentComponent: this,
                        message: "Dòng được chọn không có trong cơ sở dữ liệu",
                        title:"Lỗi", messageType:JOptionPane.ERROR_MESSAGE);
                throw new Exception(message: "Khong ton tai dong duoc chon trong co so du lieu");
            }
        }
    } catch (Exception ex) {
        ex.printStackTrace();

    }
```

```
public void displayDetails(int possition, 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(t: hoTen);
    txtSerial.setText(t: dienThoai);
    txtSerial.setText(t: serial);
    txtYeuCau.setText(t: yeuCau);
    if (trangThai.equalsIgnoreCase(anotherString: "Nhận máy")) {
        radNhanMay.setSelected(b: true);
    } else if (trangThai.equalsIgnoreCase(anotherString: "Hoàn thành")) {
        radHoanThanh.setSelected(b: true);
    } else if (trangThai.equalsIgnoreCase(anotherString: "Đang làm")) {
        radDangLam.setSelected(b: true);
    } else if (trangThai.equalsIgnoreCase(anotherString: "Đã trả máy")) {
        radDaTraMay.setSelected(b: true);
    }
    txtTech.setText(t: technician);
    txtSDTTech.setText(t: dienThoaiTechnician);
    txtGhiChu.setText(t: ghiChu);
    statements = trangThai;
    index = possition;
    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";
    }
    try {

    try {
        if (trangThai.equals(anObject: statements)) {
            JOptionPane.showMessageDialog(parentComponent: this,
                    message: "Vui lòng chọn trạng thái khác để thực hiện chỉnh sửa hóa đơn",
                    title: "Lỗi", messageType: JOptionPane.OK_OPTION);
            throw new Exception(message: "Loi chua chinh sua");
        } else {
            List<Bill> billList = controller.readDataFromFile(fileName: "src/file/bill.txt");
            billList.get(index).setStatements(statements: trangThai);
            controller.writeToFile(list: billList, fileName: "src/file/bill.txt");
            List<Machine> listMachine = controller.readDataFromFile(fileName: "src/file/machine.txt");
            for (Machine machine : listMachine) {
                if (machine.getSerial().equals(anObject: serialNum)) {
                    machine.setStatements(statements: trangThai);
                }
            }
            controller.writeToFile(list: listMachine, fileName: "src/file/machine.txt");
            JOptionPane.showMessageDialog(parentComponent: this,
                    message: "Chỉnh sửa hóa đơn thành công", title: "Thông báo",
                    messageType: JOptionPane.OK_OPTION);
            new Approval_Screen().setVisible(b: true);
            this.dispose();
        }
    }
    catch(Exception ex){
        ex.printStackTrace();
    }
```

4.5.4. Quản lý Approval_Print



Hình 2.14: Giao diện Approval_Print

- Code chuyển màn hình từ Approval_Screen sang Approval_Print và hiển thị thông tin lên màn hình Print:

```
private void bntExportActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    int selectedRow = tblBill.getSelectedRow();
    try {
        if (selectedRow == -1) {
            JOptionPane.showMessageDialog(parentComponent: this,
                    message: "Please select a record to show details.",
                    title:"Error", messageType:JOptionPane.ERROR_MESSAGE);
            throw new Exception(message: "Loi chon dong");
        } else {
            List<Bill> listBill = controller.readDataFromFile(fileName: "src/file/bill.txt");
            DefaultTableModel modelBill = (DefaultTableModel) tblBill.getModel();
            String billID = (String) modelBill.getValueAt(row: selectedRow, column: 1);
            for (Bill bill : listBill) {
                if (billID.equals(anObject: bill.getBillID())) {
                    Approval_Print approval = new Approval_Print();
                    approval.displayDetails(nameClient: bill.getNameClient(),
                            phoneClient:bill.getPhoneClient(), requirements:bill.getRequirements(),
                            statements: bill.getStatements(), nameTechnician: bill.getNameTechnician(),
                            phoneTechnician: bill.getPhoneTechnician(), notes:bill.getNote());
                    approval.setVisible(b: true);
                    this.dispose();
                }
            }
        }
    }
    catch(Exception ex){
        ex.printStackTrace();
    }

}
```

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

- Code thực hiện thao tác Export:

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

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

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

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

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

    FileNameExtensionFilter filter = new FileNameExtensionFilter
    (description:"Tệp văn bản (*.txt)", extensions: "txt");
    fileChooser.setFileFilter(filter);

    int userSelection = fileChooser.showSaveDialog(parent:this);

    if (userSelection == JFileChooser.APPROVE_OPTION) {
        try (PrintWriter writer = new PrintWriter(file: 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("SĐT Technician: " + txtSDTTech.getText());
            writer.println("Ghi chú: " + txtGhiChu.getText());
            JOptionPane.showMessageDialog(parentComponent: this, message: "Xuất file thành công");
            new Approval_Screen().setVisible(b:true);
            this.dispose();
        } catch (FileNotFoundException e) {
            JOptionPane.showMessageDialog(parentComponent: this,
                    "Lỗi không thể xuất file : " + e.getMessage(),
                    title:"Error", messageType:JOptionPane.ERROR_MESSAGE);
            e.printStackTrace();
        }
    }
}
```

III. KẾT LUẬN VÀ BÀI HỌC KINH NGHIỆM

Những kiến thức và kĩ 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