TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI

KHOA CÔNG NGHỆ THÔNG TIN

------oOo------

BÁO CÁO BÀI TẬP LỚN

Môn học: Công nghệ Java

Chủ đề: Quản lý tài khoản ngân hàng

Giảng Viên hướng dẫn: TS. Vũ Huấn

Nhóm: 24

Nguyễn Trung Dũng - 211212181

Tạ Quốc Việt - 211213440

Lớp: CNTT5 – K62 – N07

Hà Nội, tháng 4 năm 2023

Mục lục

Lời Nói Đầu

Hiện nay, ứng dụng quản lý tài khoản ngân hàng đã trở thành một công cụ quan trọng giúp cho các giao dịch viên trong hoạt động kinh doanh hàng ngày. Với sự phát triển của công nghệ thông tin, các ứng dụng này ngày càng được cải tiến và nâng cao tính năng để đáp ứng tốt hơn nhu cầu của người dùng.

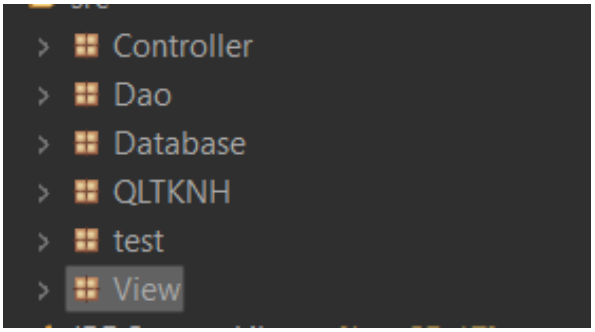
Đặc biệt, trong bối cảnh chuyển đổi số đang diễn ra mạnh mẽ, ứng dụng quản lý tài khoản ngân hàng là một trong những sản phẩm đóng vai trò quan trọng trong việc kết nối giữa ngân hàng và khách hàng. Các giao dịch viên có thể sử dụng ứng dụng này để quản lý tài khoản của khách hàng một cách nhanh chóng và tiện lợi, giúp cho việc đàm phán và thực hiện các giao dịch trở nên thuận tiện hơn bao giờ hết.

Tuy nhiên, để sử dụng ứng dụng quản lý tài khoản ngân hàng một cách hiệu quả, các giao dịch viên cần phải được đào tạo về cách sử dụng và cập nhật những thông tin mới nhất về sản phẩm này. Việc hợp tác giữa các trường đào tạo ngành công nghệ thông tin và các doanh nghiệp công nghệ sẽ giúp cho các giao dịch viên được tiếp cận với các công nghệ mới nhất và nâng cao kỹ năng của mình trong việc sử dụng ứng dụng quản lý tài khoản ngân hàng.

TỔNG QUAN ĐỀ BÀI

1. Đề tài bài tập lớn như sau:

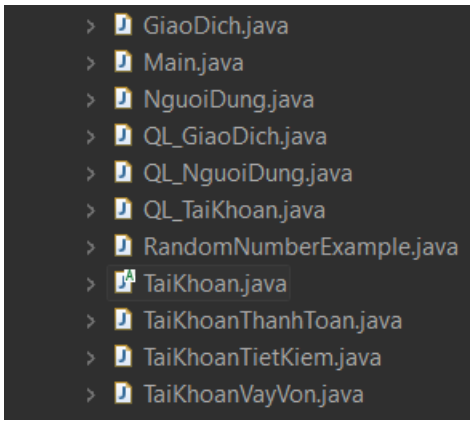
Xây dựng một chương trình quản lý tài khoản ngân hàng, chương trình được sử dụng bởi nhân viên ngân hàng.
Chương trình được xây dựng theo sơ đồ MVC (model – view – controller) và 1 package test để chạy thử chương trình:



Hình 1: mô tả các package trong đề tài

2. Package QLTKNH (model):

Gồm 11 class:



Hình 2.1: mô tả các class trong package QLTKNH(model)

Class TaiKhoan là một abstract class gồm 3 thuộc tính sotk (số tài khoản), tentk (tên tài khoản), sodutk (số dư tài khoản) và các hàm tạo, các phương thức cơ bản như nạp, chuyển, rút và tính lãi suất.

```java
public abstract class TaiKhoan {
    private String sotk;
    private String tentk;
    private double  sodutk;

    public TaiKhoan(String sotk, String tentk, double  sodutk) {
        this.sotk = sotk;
        this.tentk = tentk;
        this.sodutk = sodutk;
    }


    public TaiKhoan(String sotk, double sodutk) {
        super();
        this.sotk = sotk;
        this.sodutk = sodutk;
    }


    public TaiKhoan(String sotk) {
        super();
        this.sotk = sotk;
    }
```

Hình 2.2: mô tả các thuộc tính trong class Taikhoan

```java
    /*các phương thức cơ bản của một tài khoản ngân hàng */

    public abstract void napTien(double  soTien);

    public abstract void rutTien(double  soTien);

    public abstract void chuyenKhoan(TaiKhoan nguonNhan, double  soTien);

    protected abstract void congLai();
}
```

Hình 2.3: mô tả các Phương thức trong class Taikhoan

- **Class TaiKhoanThanhToan, TaiKhoanTietKiem, TaiKhoanVayVon** là 3 class kế thừa từ class TaiKhoan.

- **Class TaiKhoanThanhToan** có thuộc tính PHI_DICH_VU (phí dịch vụ) được cho trước bằng 1000 và loaitaikhoan (loại tài khoản) và 3 phương thức chuyển nạp rút (sau đó lưu lại lịch sử giao dịch).

- Nạp tiền:

```
@Override
public void napTien(double soTien) {

    if(soTien <= 0) {
        System.out.println("Số tiền nhập phải lớn hơn 0!");
    } else {
        setSodutk(getSodutk() + soTien);
        System.out.println("Nạp tiền thành công!");
        System.out.println("- Số dư tài khoản: " + getSodutk());
        GiaoDich gd = new GiaoDich("napien" + RandomNumberExample.RandomNumber(), soTien, LocalDateTime.now(), getSotk(),"Nạp Tiền");

        GiaoDich_Dao.getInstance().insert(gd);

    }
```

- *Hình 2.4: mô tả Phương thức nạp tiền trong class TaikhoanThanhToan*

- Rút tiền:

```
@Override
public void rutTien(double  soTien) {
    if(soTien <= 0) {
        System.out.println("Số tiền nhập phải lớn hơn 0!");
    } else if(soTien <= (getSodutk() - PHI_DICH_VU)) {
        setSodutk(getSodutk() - soTien - PHI_DICH_VU);
        System.out.println("Rút tiền thành công!");
        System.out.println("- Phí dịch vụ: " + PHI_DICH_VU);
        System.out.println("- Số dư tài khoản: " + getSodutk());
        GiaoDich gd = new GiaoDich("rutien" + RandomNumberExample.RandomNumber(), soTien, LocalDateTime.now(), getSotk(),"Rút tiền");

        GiaoDich_Dao.getInstance().insert(gd);
    } else {
        System.out.println("Số dư không đủ để thực hiện rút tiền!");
    }
}
```

- *Hình 2.5: mô tả Phương thức rút tiền trong class TaikhoanThanhToan*

- Chuyển khoản:

```
@Override
public void chuyenKhoan(TaiKhoan nguoiNhan, double  soTien) {
    if(soTien <= 0) {
        System.out.println("Số tiền nhập phải lớn hơn 0!");
    } else if(soTien <= (getSodutk() - PHI_DICH_VU)) {
        setSodutk(getSodutk() - soTien - PHI_DICH_VU);
        nguoiNhan.napTien(soTien);
        System.out.println("Rút tiền thành công!");
        System.out.println("- Phí dịch vụ: " + PHI_DICH_VU);
        System.out.println("- Số dư tài khoản: " + getSodutk());

        GiaoDich gd = new GiaoDich("Chuyenkhoan" + RandomNumberExample.RandomNumber(), soTien, LocalDateTime.now(), "Chuyển khoản", getSotk(),nguoiNhan.getSotk());
        GiaoDich_Dao.getInstance().insert(gd);
    } else {
        System.out.println("Số dư không đủ để thực hiện chuyển tiền!");
    }
}
```

Hình 2.6: mô tả Phương thức chuyển khoản trong class TaikhoanThanhToan

- **Class TaiKhoanTietKiem** có thêm 2 thuộc tính là LAI_TIET_KIEM (lãi suất tiết kiệm) được cho trước bằng 6% và ngayGui (ngày gửi) và phương thức cộng lãi tiết kiệm.

- Tính tiền lãi tiết kiệm:

```
/*phương thức tính tiền lãi tiết kiệm */


public void congLai() {
    double tienLai = getSodutk() * LAI_TIET_KIEM / 100 * 180 / 365;
    long laiLamTron = Math.round(tienLai);
    setSodutk(getSodutk() + laiLamTron);
    System.out.println("Cộng lãi thành công!");
    System.out.println("- Tiền lãi: " + laiLamTron);
    System.out.println("- Số dư tài khoản: " + getSodutk());
    GiaoDich gd = new GiaoDich("congLai" + RandomNumberExample.RandomNumber(), laiLamTron, LocalDateTime.now(), getSotk(),"Cộng Lãi");
    GiaoDich_Dao.getInstance().insert(gd);

}
```

Hình 2.7: mô tả Phương thức tính tiền lãi tiết kiệm trong class TaikhoanTietKiem

- Nạp tiền tài khoản tiết kiệm:

```
@Override
public void napTien(double soTien) {
    if(soTien <= 0) {
        System.out.println("Số tiền nhập phải lớn hơn 0!");
    } else {
        setSodutk(getSodutk() + soTien);
        System.out.println("Nạp tiền thành công!");
        System.out.println("- Số dư tài khoản: " + getSodutk());
        GiaoDich gd = new GiaoDich("napTien" + RandomNumberExample.RandomNumber(), soTien, LocalDateTime.now(), getSotk(),"Nạp Tiền Tiết Kiệm");
         GiaoDich_Dao.getInstance().insert(gd);
    }
}
```

Hình 2.8: mô tả Phương thức nạp tiền trong class TaikhoanTietKiem

- Rút tiền tài khoản tiết kiệm:

```
@Override
public void rutTien(double soTien) {
    if(soTien <= 0) {
        System.out.println("Số tiền nhập phải lớn hơn 0!");
    } else if(soTien <= (getSodutk() - PHI_DICH_VU)) {
        setSodutk(getSodutk() - soTien - PHI_DICH_VU);
        System.out.println("Rút tiền thành công!");
        System.out.println("- Phí dịch vụ: " + PHI_DICH_VU);
        System.out.println("- Số dư tài khoản: " + getSodutk());
        GiaoDich gd = new GiaoDich("rutTien" + RandomNumberExample.RandomNumber(), soTien, LocalDateTime.now(), getSotk(),"Rút Tiền Tiết Kiệm");
        GiaoDich_Dao.getInstance().insert(gd);
    } else {
        System.out.println("Số dư không đủ để thực hiện rút tiền!");
    }
}
```

Hình 2.9: mô tả Phương thức rút tiền trong class TaikhoanTietKiem

- **Class TaiKhoanVayVon** có thêm 4 thuộc tính soTienVay (số tiền vay), soTienLai (số tiền lãi) , LAI_SUAT_VAY (lãi suất vay vốn) được cho trước bằng 0,1 và ngayVay(ngày vay) , phương thức tính tiền lãi vay vốn phải trả (sau đó lưu lại lịch sử giao dịch).

- Phương thức tính tiền lãi phải trả và kiểm tra xem có trả nợ muộn không:

```
    /* phương thức tính tiền lãi phải trả */

    public void tinhTienLai(LocalDate ngayTraNo) {
        long soNgayVay = ngayTraNo.toEpochDay() - ngayVay.toEpochDay();
        LocalDate ngayHanMuc = ngayVay.plusMonths(6);
        long soNgayHanChoPhep = ngayHanMuc.toEpochDay() - ngayVay.toEpochDay();
        double laiPhaiTra = soTienVay * LAI_SUAT_VAY * soNgayVay / 365;
        double laiabs =Math.abs(laiPhaiTra);
        long lailamtron = Math.round(laiabs);
        soTienLai += lailamtron;
        setSodutk(getSodutk() + soTienVay);
        if(soNgayVay <= soNgayHanChoPhep) {
            if(getSodutk() < (lailamtron + soTienVay)) {
                System.out.println("Tổng số tiền còn lại phải trả là: " + Math.abs(getSodutk() - (lailamtron + soTienVay)));
                System.out.println("số dư tài khoản là: 0 ");
                setSodutk(0);
            }
            else {
                setSodutk(getSodutk() - (lailamtron + soTienVay));
                System.out.println("Số dư tài khoản còn lại là: " + getSodutk());
            }
        }
        else {
            long soNgayTraNoMuon = soNgayVay - soNgayHanChoPhep;
            double phatQuaHan = soTienVay * 0.05 * soNgayTraNoMuon / 365;
            long phatLamTron = Math.round(phatQuaHan);
            System.out.println("Tài khoản vay vốn đã quá hạn " + soNgayTraNoMuon + " ngày.");
            if(getSodutk() < (phatLamTron + soTienVay + lailamtron)) {
                System.out.println("Tổng số tiền còn lại phải trả là: " + Math.abs(getSodutk() - lailamtron - soTienVay - phatLamTron));
                System.out.println("số dư tài khoản là: 0 ");
                setSodutk(0);
            }
            else {
                setSodutk(getSodutk() - (lailamtron + soTienVay + phatLamTron));
                System.out.println("Số dư tài khoản còn lại là: " + getSodutk());
            }
        }
        GiaoDich gd = new GiaoDich("tinhtienlai" + RandomNumberExample.RandomNumber(), lailamtron, LocalDateTime.now(), getSotk(),"Tính tiền lãi");
       GiaoDich_Dao.getInstance().insert(gd);
    }
```

Hình 2.10: mô tả Phương thức tính tiền lãi phải trả trong class TaikhoanVayVon

Class NguoiDung và Class GiaoDich là class gồm các hàm tạo, các getter, setter để tạo và set và lấy thông tin người dùng và giao dịch.

- Class NguoiDung:

```java
public class NguoiDung {
private String CCCD;
private String sdt;
private TaiKhoan tk_nguoidung;
private String tenNguoidung;
private LocalDate NgaySinh;
private String sotaikhoan;

public NguoiDung(String CCCD, String tenNguoidung , String sdt, LocalDate NgaySinh, TaiKhoan tk_nguoidung) {
    this.CCCD = CCCD;
    this.sdt = sdt;
    this.tk_nguoidung = tk_nguoidung;
    this.tenNguoidung = tenNguoidung;
    this.NgaySinh = NgaySinh;
}

public NguoiDung(String CCCD,String tenNguoidung, String sdt , LocalDate NgaySinh, String sotk_nguoidung) {
    this.CCCD = CCCD;
    this.sdt = sdt;
    sotaikhoan = sotk_nguoidung;
    this.tenNguoidung = tenNguoidung;
    this.NgaySinh = NgaySinh;
    }
```

Hình 2.11: mô tả code trong class NguoiDung

- Class GiaoDich:

```java
public class GiaoDich {
    private String magiaodich;

    private double sotien;
    private LocalDateTime thoigian_giaodich;
    private String loai_giaodich;
    private String taikhoan_giaodich;
    private String taikhoan_nhan;

    public GiaoDich(String magiaodich, double sotien, LocalDateTime thoigian_giaodich,String taikhoan_giaodich, String loai_giaodich) {
        this.magiaodich = magiaodich;
        this.sotien = sotien;
        this.thoigian_giaodich = thoigian_giaodich;
        this.taikhoan_giaodich = taikhoan_giaodich;
        this.loai_giaodich = loai_giaodich;

    }

    public GiaoDich(String magiaodich, double sotien, LocalDateTime thoigian_giaodich, String loai_giaodich, String taikhoan_giaodich,String taikhoan
        this.magiaodich = magiaodich;
        this.sotien = sotien;
        this.thoigian_giaodich = thoigian_giaodich;
        this.loai_giaodich = loai_giaodich;
        this.taikhoan_giaodich = taikhoan_giaodich;
        this.taikhoan_nhan = taikhoan_nhan;
    }
```

Hình 2.12: mô tả code trong class NguoiDung

Class QL_GiaoDich, QL_TaiKhoan, QL_NguoiDung có các tính năng như tìm kiếm, thêm, sửa, xóa, kiểm tra tồn tài các giao dịch, tài khoản và người dùng.

- Class QL_GiaoDich:

```java
public class QL_GiaoDich {
private static List<GiaoDich>  ds_giaodich;

public QL_GiaoDich() {
    this.ds_giaodich = new ArrayList<>();
}
public static void themGiaoDich(GiaoDich gd) {
    try {
        ds_giaodich.add(gd);
    } catch (NullPointerException e) {
        System.out.println("Danh sách giao dịch không được để trống");
    } catch (Exception e) {
        System.out.println("Có lỗi xảy ra khi thêm giao dịch");
    }
}

public void xoaGiaoDich( String maGiaoDich) {
    try {
        ds_giaodich.removeIf(gd -> gd.getMagiaodich().equals(maGiaoDich));
    } catch (NullPointerException e) {
        System.out.println("Danh sách giao dịch không được để trống");
    } catch (Exception e) {
        System.out.println("Có lỗi xảy ra khi xóa giao dịch");
    }
}
public String timKiemGiaoDich(String maGiaoDich) {
    try {
        for (GiaoDich gd : ds_giaodich) {
            if (gd.getMagiaodich().equals(maGiaoDich)) {
                return gd.toString();
            }
        }
        System.out.println("Không tìm thấy giao dịch có mã " + maGiaoDich);
    } catch (NullPointerException e) {
        System.out.println("Danh sách giao dịch không được để trống");
    } catch (Exception e) {
        System.out.println("Có lỗi xảy ra khi tìm kiếm giao dịch");
    }
    return null;
}
```

Hình 2.13: mô tả code các phương thức trong class QL_giaodich

```java
public void duyetGiaoDich() {
    try {
        for (GiaoDich gd : ds_giaodich) {
            System.out.println(gd.toString());
        }
    } catch (NullPointerException e) {
        System.out.println("Danh sách giao dịch không được để trống");
    } catch (Exception e) {
        System.out.println("Có lỗi xảy ra khi duyệt giao dịch");
    }
}
```

Hình 2.13: mô tả code các phương thức trong class QL_giaodich

- Class QL_TaiKhoan:

```java
public class QL_TaiKhoan {
private Map<String, TaiKhoan> ds_taikhoan;
public QL_TaiKhoan() {
    this.ds_taikhoan = new HashMap<>();
}
public TaiKhoan timKiemTaiKhoan(String sotaikhoan) {
    try {
        if (ds_taikhoan.containsKey(sotaikhoan)) {
            return ds_taikhoan.get(sotaikhoan);
        }
    } catch (NullPointerException e) {
        System.out.println("Danh sách tài khoản không được để trống");
    } catch (Exception e) {
        System.out.println("Có lỗi xảy ra khi tìm kiếm tài khoản");
    }
    return null;
}

public Boolean kiemtra_TaiKhoan(String sotaikhoan) {
    TaiKhoan tk = ds_taikhoan.get(sotaikhoan);
    if(tk != null && tk.getSotk().equals(sotaikhoan) ) {
        return true;
    }
    return false;
}

public void them_TaiKhoan(TaiKhoan tk) {
    if(ds_taikhoan.containsKey(tk.getSotk())) {
        throw new IllegalArgumentException("Tài Khoản Đã tồn tại");
    }
    ds_taikhoan.put(tk.getSotk(), tk);
}
public void xoa_TaiKhoan(String sotk) {
    if(!ds_taikhoan.containsKey(sotk)) {
        throw new IllegalArgumentException("Tài Khoản không tồn tại");
    }
    ds_taikhoan.remove(sotk);
}
public void capnhatThongTin_TaiKhoan(TaiKhoan tk,String sotk, String tentk) {

    if(this.timKiemTaiKhoan(tk.getSotk()).equals(null)) {
        System.out.println("Không tìm thấy tài khoản có số tk hợp lệ");
    }
    else {
       tk.getSotk();
       tk.setTentk(tentk);
       ds_taikhoan.put(sotk, tk);
    }
}
public void duyetDanhSachTaiKhoan() {
    for(Map.Entry<String, TaiKhoan> entry : ds_taikhoan.entrySet()) {
        System.out.println(entry.getValue());
    }
}
```

Hình 2.14: mô tả code các phương thức trong class QL_TaiKhoan

- Class QL_NguoiDung:

```java
public class QL_NguoiDung {
private Map<String, NguoiDung> ds_NguoiDung;

public QL_NguoiDung() {
    this.ds_NguoiDung = new HashMap<>();
}
public NguoiDung thongtin_NguoiDung(String CCCD) {
    if(ds_NguoiDung.containsKey(CCCD)) {
        return ds_NguoiDung.get(CCCD);
    }
    return null;
}

public boolean kiemTraNguoiDungTonTai(String CCCD) {
    return ds_NguoiDung.containsKey(CCCD);
}

public void them_NguoiDung(NguoiDung nguoidung) {
    if(ds_NguoiDung.containsKey(nguoidung.getCCCD())) {
        throw new IllegalArgumentException("Tài Khoản Đã tồn tại");
    }
    ds_NguoiDung.put(nguoidung.getCCCD(), nguoidung);
}
public Map<String, NguoiDung> getDs_NguoiDung() {
    return ds_NguoiDung;
}
public void setDs_NguoiDung(Map<String, NguoiDung> ds_NguoiDung) {
    this.ds_NguoiDung = ds_NguoiDung;
}
public void xoa_NguoiDung(String CCCD) {
    if(!ds_NguoiDung.containsKey(CCCD)) {
        throw new IllegalArgumentException("Tài Khoản không tồn tại");
    }
    ds_NguoiDung.remove(CCCD);
}
public void capnhatThongTin_NguoiDung(NguoiDung nguoidung,String CCCD, String sdt, String TenNguoiDung, LocalDate ngaysinh) {

   if(this.thongtin_NguoiDung(nguoidung.getCCCD()).equals(null)) {
      System.out.println("Không tìm thấy người dùng có tên tk hợp lệ");
   }
   else {
      nguoidung.setCCCD(CCCD);
      nguoidung.setSdt(sdt);
      nguoidung.setTenNguoidung(TenNguoiDung);
      nguoidung.setNgaySinh(ngaysinh);

      ds_NguoiDung.put(CCCD, nguoidung);
   }
}
```

Hình 2.14: mô tả code các phương thức trong class QL_NguoiDung

Class RandomNumberExample có tác dụng tạo ra số ngẫu nhiên.(dùng để sinh ra mã giao dịch ngẫu nhiên)

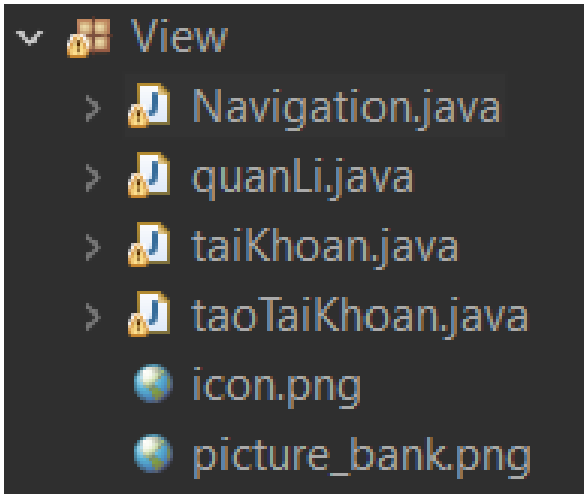
```java
public class RandomNumberExample {
    public static String RandomNumber() {
        Random random = new Random();
        int randomNumber = random.nextInt(1000000); // Số ngẫu nhiên trong khoảng từ 0 đến 999999
        String formattedNumber = String.format("%06d", randomNumber); // Định dạng số thành chuỗi
        return formattedNumber;
    }
}
```

Hình 2.15: mô tả code của class RandomNumberExample

3. Package View:

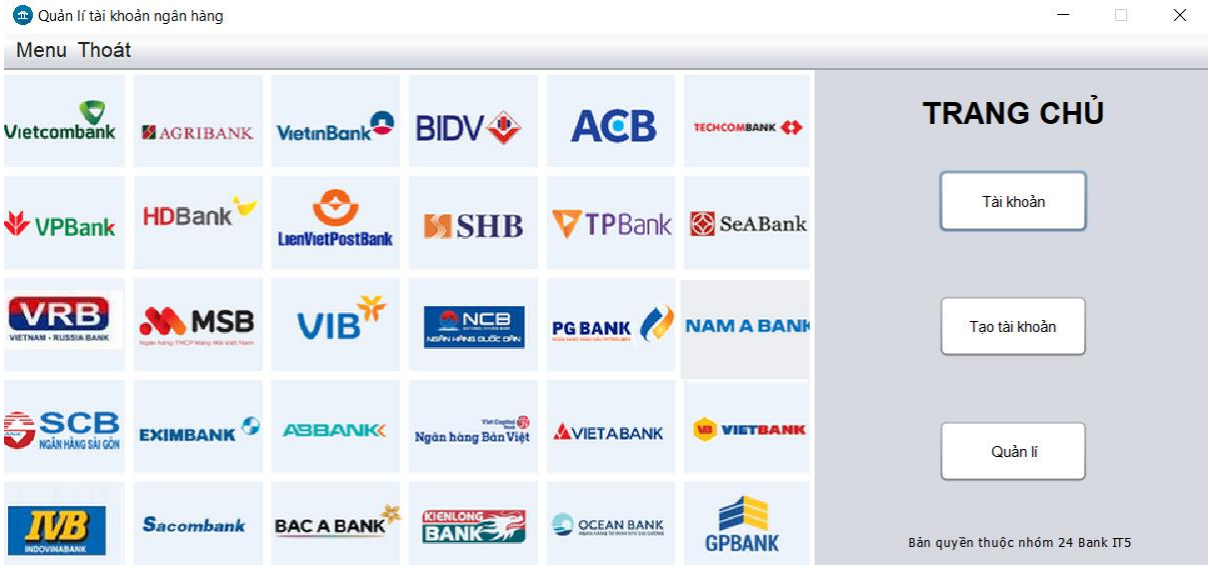
Ở phần này chúng em có sử dụng Jframe của thư viện javaSwing để thiết kiết phần giao diện chương trình,

Gồm 4 class và 2 tệp ảnh để làm logo ứng dụng:



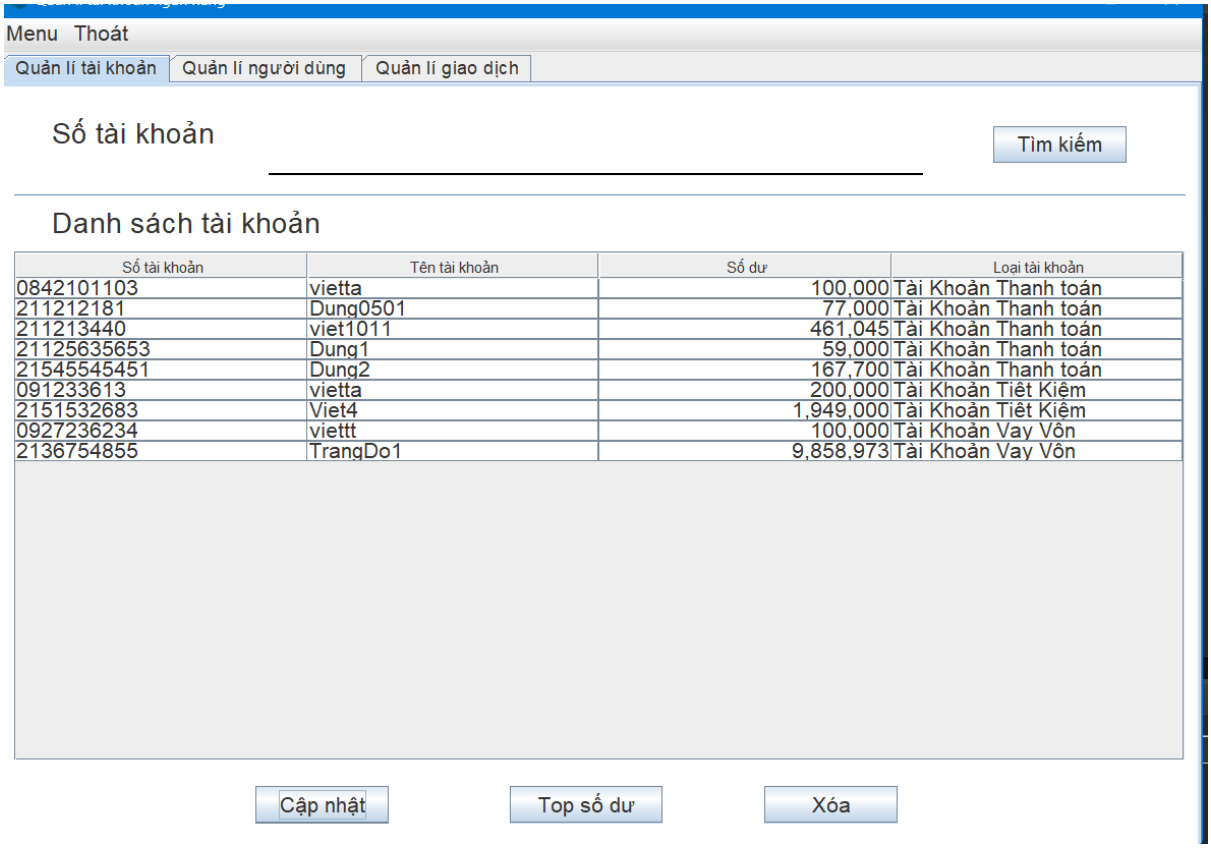
Hình 3.1: mô tả các class trong package view

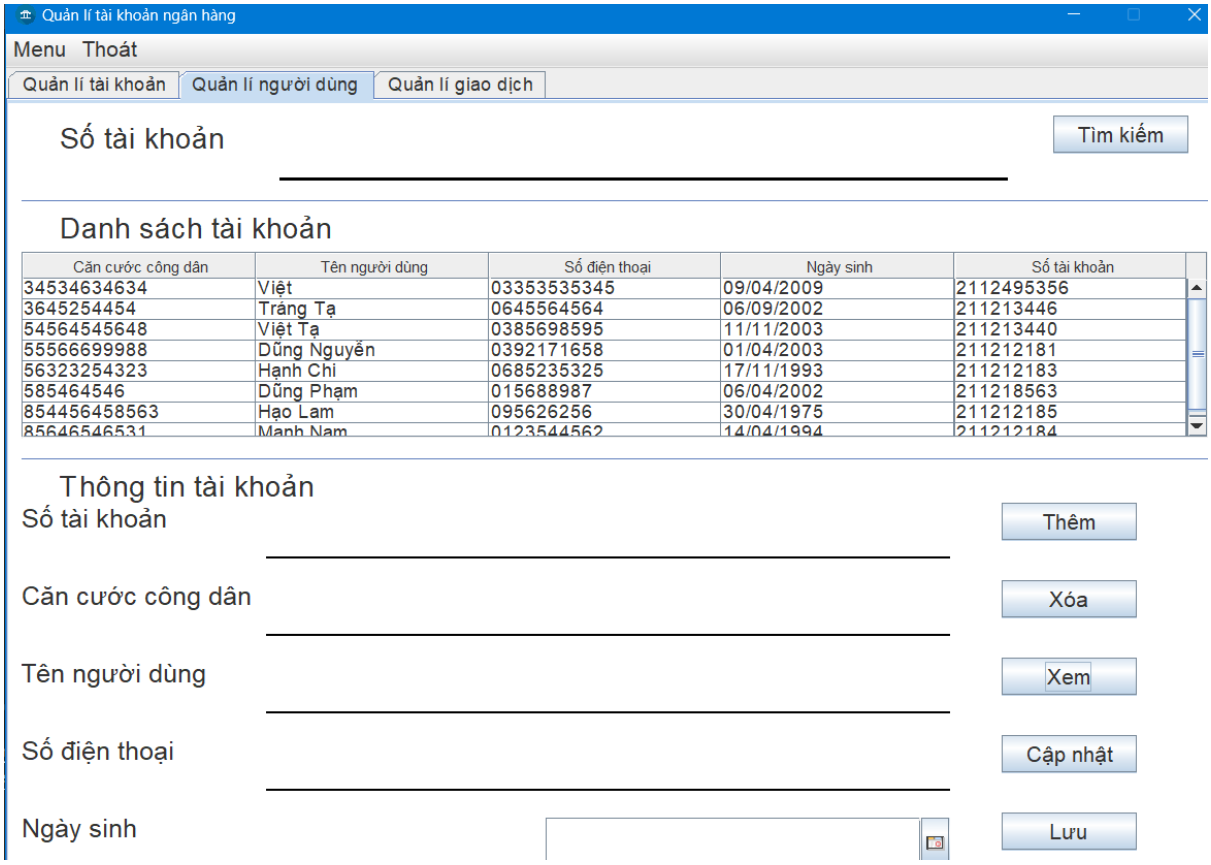
Class Navigation để làm trang điều hướng:

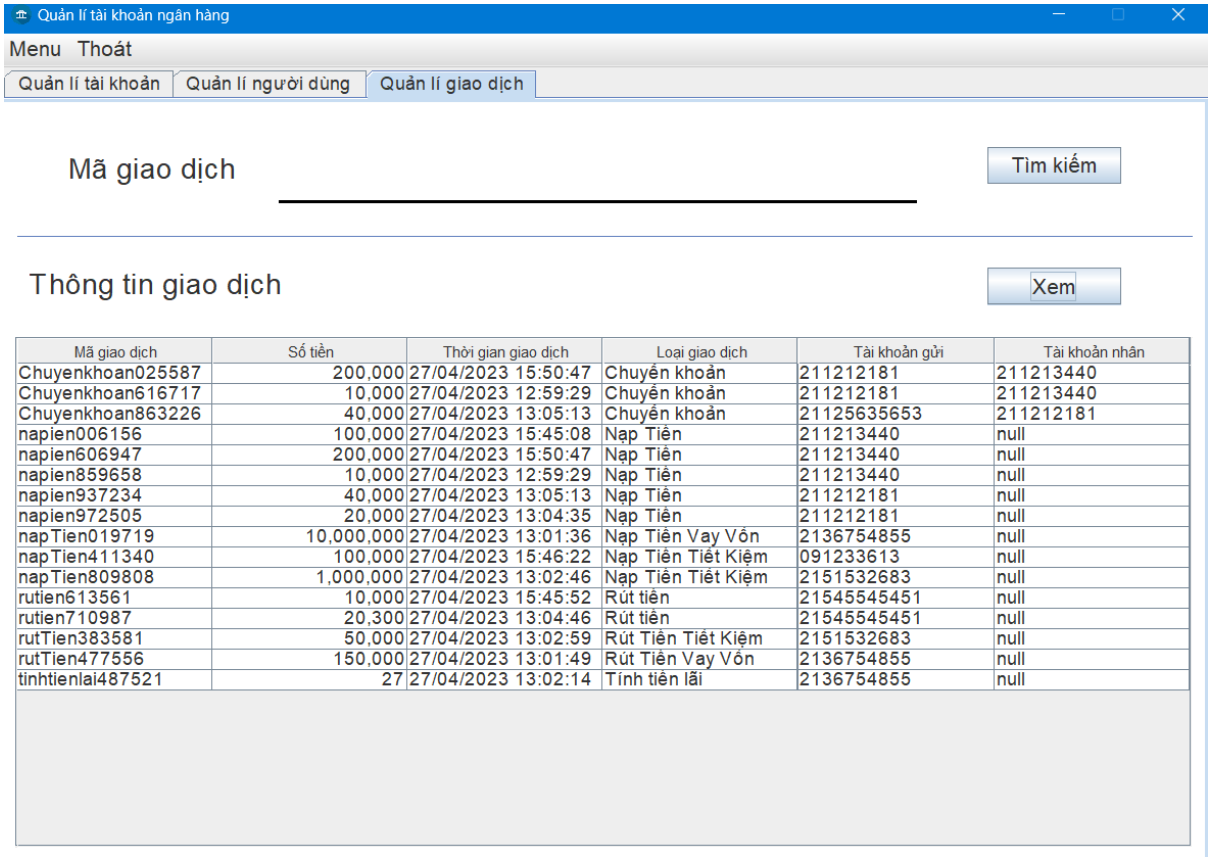


Hình 3.2: mô tả giao diện class Navigation

Class quanLi thực hiện chức năng quản lí các tài khoản, người dùng và giao dịch:

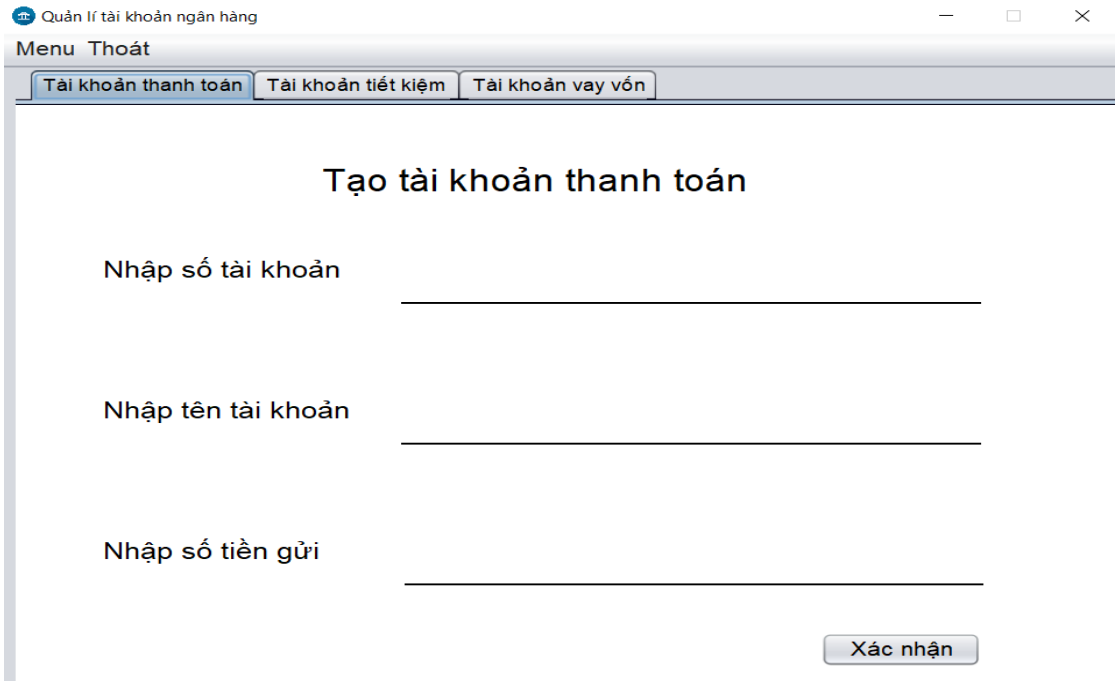

Menu Thoát
Quản lí tài khoản
Quản lí người dùng
Quản lí giao dịch
Số tài khoản
Tìm kiếm
Danh sách tài khoản
Số tài khoản | Tên tài khoản | Số dư | Loại tài khoản
0842101103 | vietta | 100,000 | Tài Khoản Thanh toán
211212181 | Dung0501 | 77,000 | Tài Khoản Thanh toán
211213440 | viet1011 | 461,045 | Tài Khoản Thanh toán
21125635653 | Dung1 | 59,000 | Tài Khoản Thanh toán
21545545451 | Dung2 | 167,700 | Tài Khoản Thanh toán
091233613 | vietta | 200,000 | Tài Khoản Tiết Kiệm
2151532683 | Viet4 | 1,949,000 | Tài Khoản Tiết Kiệm
0927236234 | viettt | 100,000 | Tài Khoản Vay Vốn
2136754855 | TrangDo1 | 9,858,973 | Tài Khoản Vay Vốn
Cập nhật
Top số dư
Xóa

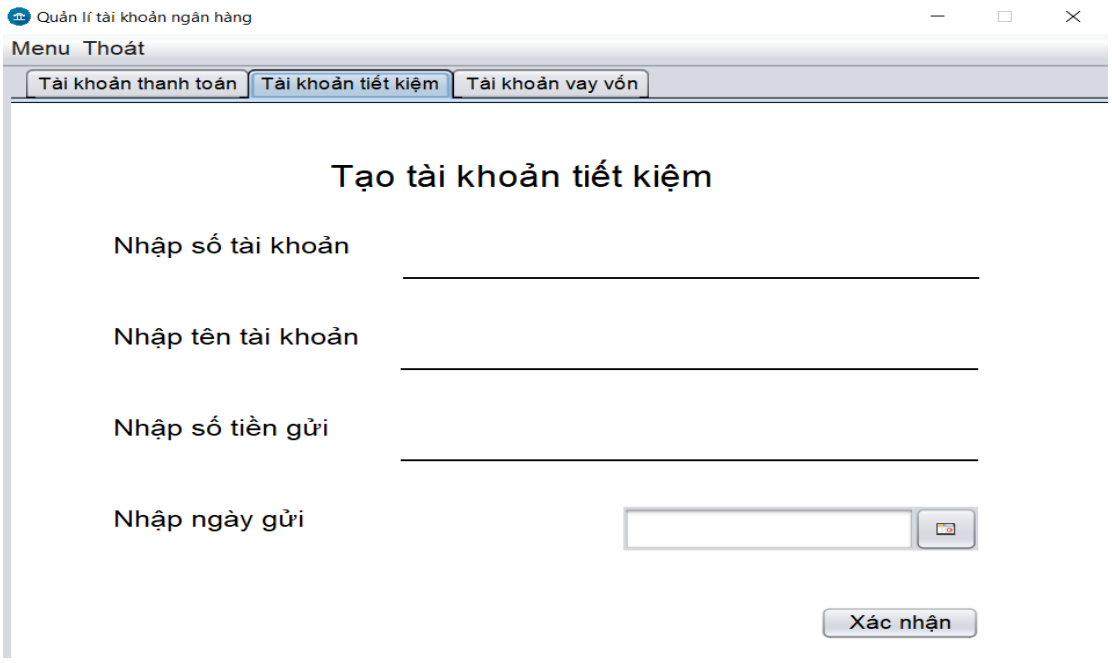

Quản lí tài khoản ngân hàng
Menu Thoát
Quản lí tài khoản
Quản lí người dùng
Quản lí giao dịch
Số tài khoản
Tìm kiếm
Danh sách tài khoản
Căn cước công dân | Tên người dùng | Số điện thoại | Ngày sinh | Số tài khoản
34534634634 | Việt | 03353535345 | 09/04/2009 | 2112495356
3645254454 | Tráng Tạ | 0645564564 | 06/09/2002 | 211213446
54564545648 | Việt Tạ | 0385698595 | 11/11/2003 | 211213440
55566699988 | Dũng Nguyễn | 0392171658 | 01/04/2003 | 211212181
56323254323 | Hạnh Chi | 0685235325 | 17/11/1993 | 211212183
585464546 | Dũng Phạm | 015688987 | 06/04/2002 | 211218563
854456458563 | Hạo Lam | 095626256 | 30/04/1975 | 211212185
85646546531 | Mạnh Nam | 0123544562 | 14/04/1994 | 211212184
Thông tin tài khoản
Số tài khoản
Thêm
Căn cước công dân
Xóa
Tên người dùng
Xem
Số điện thoại
Cập nhật
Ngày sinh
Lưu

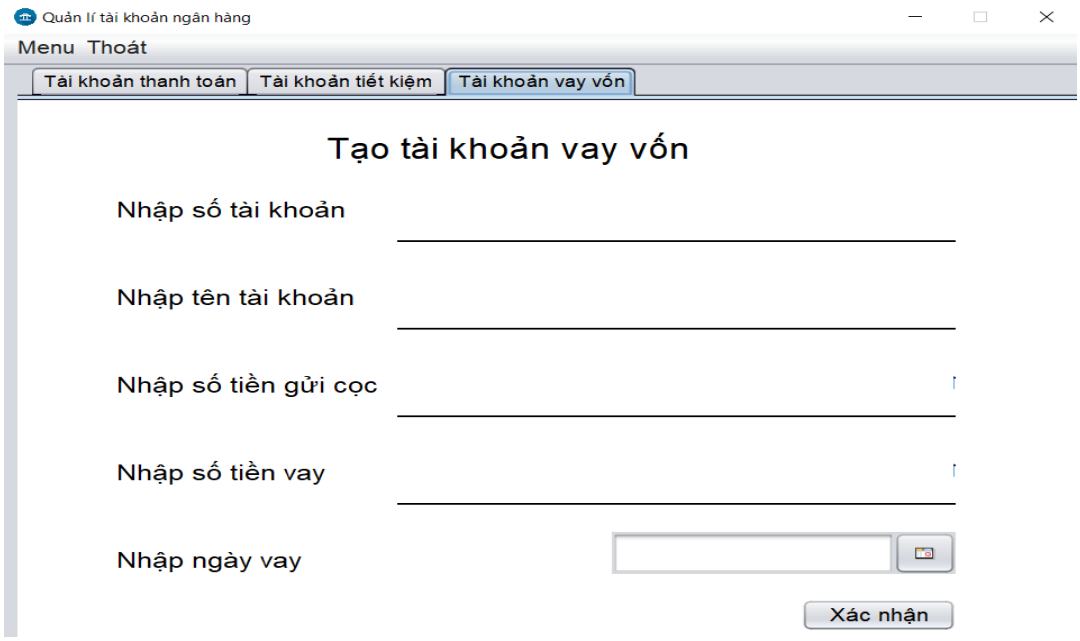


Hình 3.3: mô tả giao diện class QuanLy

Class taoTaiKhoan có chức năng tạo các tài khoản thanh toán, tài khoản tiết kiệm và tài khoản vay vốn:

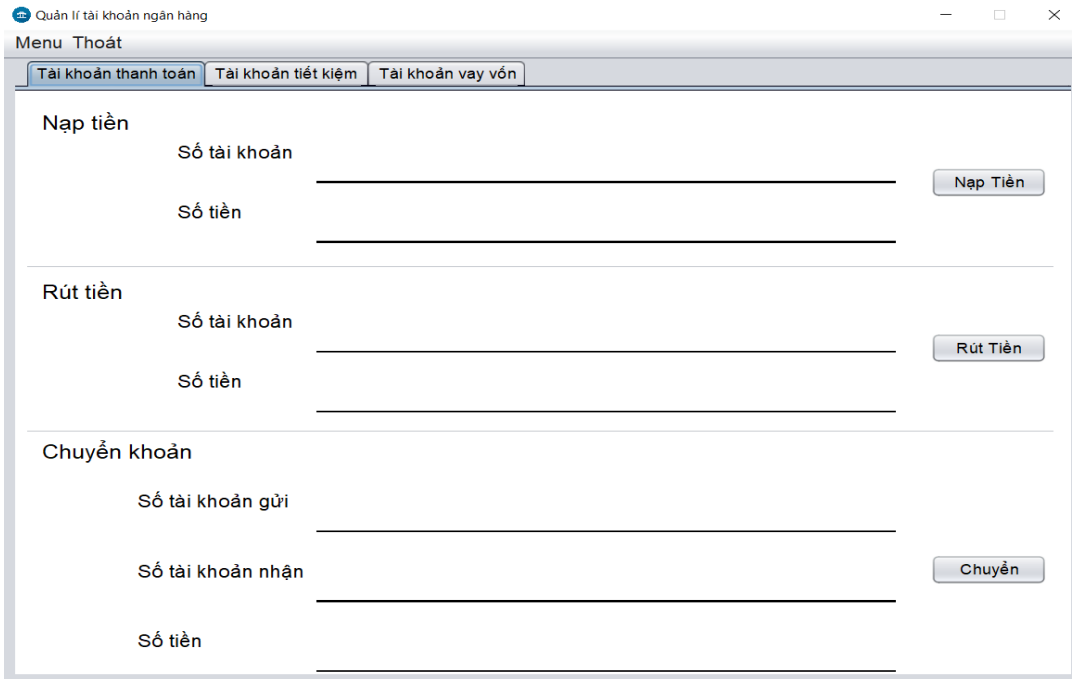

Quản lí tài khoản ngân hàng
Menu Thoát
Tài khoản thanh toán
Tài khoản tiết kiệm
Tài khoản vay vốn
Tạo tài khoản thanh toán
Nhập số tài khoản
Nhập tên tài khoản
Nhập số tiền gửi
Xác nhận

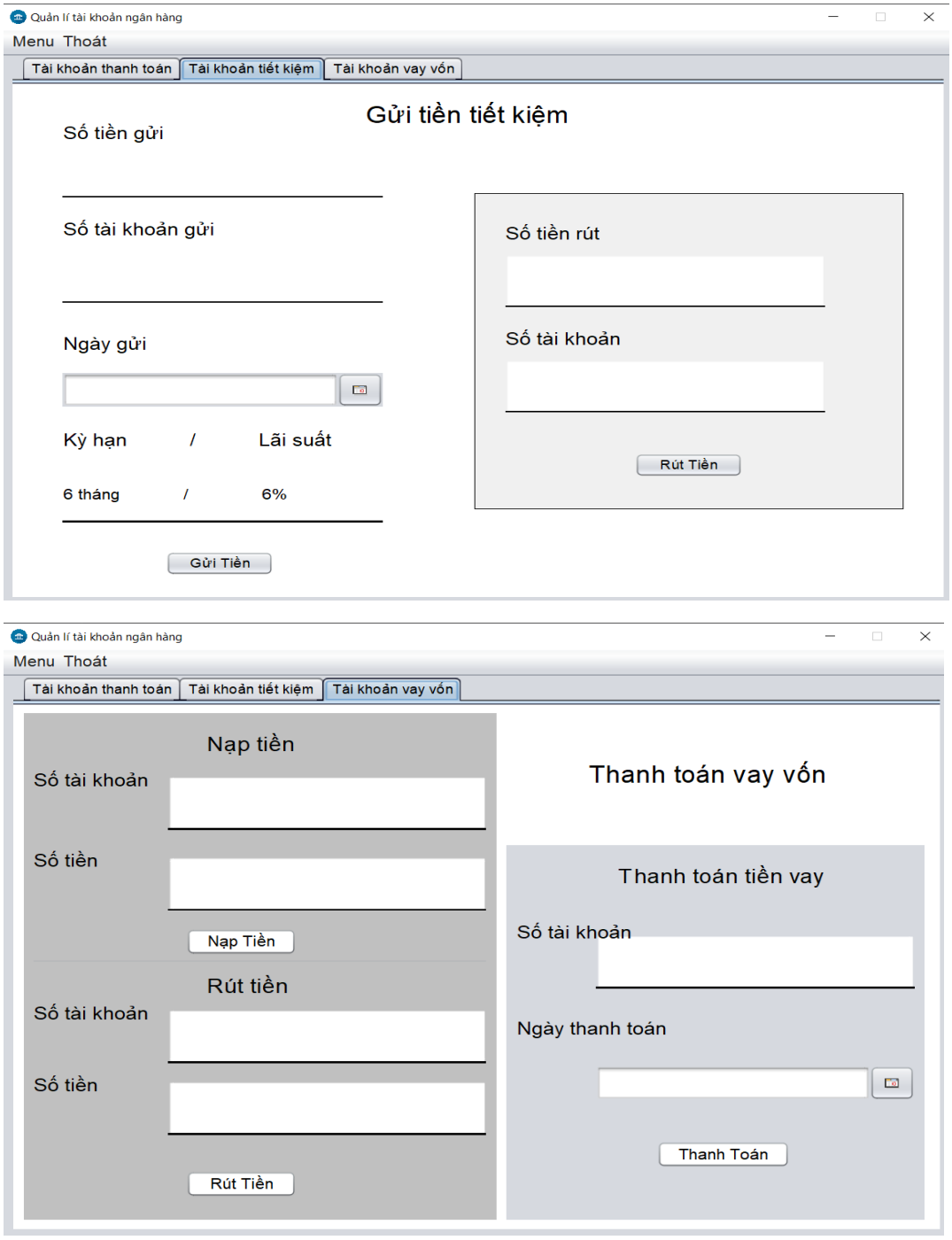

Quản lí tài khoản ngân hàng
Menu Thoát
Tài khoản thanh toán
Tài khoản tiết kiệm
Tài khoản vay vốn
Tạo tài khoản tiết kiệm
Nhập số tài khoản
Nhập tên tài khoản
Nhập số tiền gửi
Nhập ngày gửi
Xác nhận



Hình 3.4: mô tả giao diện class TaoTaiKhoan

Class taiKhoan có chức năng thực hiện các tính năng của tài khoản thanh toán, tài khoản tiết kiệm và tài khoản vay vốn như nạp, rút, chuyển khoản, tính lãi vay vốn, tiết kiệm:





Hình 3.4: mô tả giao diện class TaiKhoan

4. Package Controller:

Gồm 13 class để xử lí các nút bấm của từng lớp view:

Hình 4.1: mô tả các class trong package Controller

Một vài hình ảnh về code xử lý event của package controller:

```
/* controller chuyển cửa sổ  */
@Override
public void actionPerformed(ActionEvent e) {
    String src = e.getActionCommand();

    if(src.equals("Tài khoản")) {
        this.nav.setVisible(false);
        new taiKhoan();

    }

    else if(src.equals("Tạo tài khoản")) {
        this.nav.setVisible(false);
        new taoTaiKhoan();
    }

    else if(src.equals("Quản lí")) {
        this.nav.setVisible(false);

        new quanLi();


    }
    else if(src.equals("Đăng xuất")) {
        this.nav.setVisible(false);
        this.nav.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

}
```

```java
@Override
public void actionPerformed(ActionEvent e) {

  String cm = e.getActionCommand();

  JOptionPane.showMessageDialog(view, "bạn muốn " +cm );
  if(cm.equals("Xóa")) {
      this.view.xoaThongTinTaiKhoan();
  }
  else if(cm.equals("Cập nhật")) {
      this.view.ThemVaoBangTaikhoan();

  }
  else if(cm.equals("Tìm kiếm"))
  {
      this.view.timTaiKhoan();
  }
  else if(cm.equals("Top số dư")) {
      this.view.TopsoDu();
  }
}
```

```java
/* controller tab quản lý người dùng*/

@Override
public void actionPerformed(ActionEvent e) {
    String cm = e.getActionCommand();
    JOptionPane.showMessageDialog(ql_view, "bạn muốn " +cm );
    if(cm.equals("Thêm")) {

        this.ql_view.themNguoiDung();
        this.ql_view.removeForm();

    }
    else if(cm.equals("Cập nhật")) {
        this.ql_view.HienThiThongTinNguoiDung();
    }
    else if(cm.equals("Luu")) {
        this.ql_view.CapnhatNguoiDung();
    }
    else if(cm.equals("Xóa")) {
        this.ql_view.xoaThongTinNguoiDung();
    }
    else if(cm.equals("Tìm kiếm")) {
        this.ql_view.timNguoiDung();
    }
    else if(cm.equals("Xem")) {
        this.ql_view.removeForm();
        this.ql_view.themVaoBangND();
    }
}
```

```java
/*controller tab tao tài khoản tiết kiệm */

@Override
public void actionPerformed(ActionEvent e) {
    String cm = e.getActionCommand();
    JOptionPane.showMessageDialog(view2, "bạn muốn " +cm );
    if(cm.equals("Xác nhận")) {
        String stk = view2.textField_nhapSTKTK.getText();
        String ttk = view2.textField_nhapTTKTK.getText();
        double st = Double.parseDouble(view2.textField_STG.getText());
        Date date = view2.dateChooser_ngayGui.getDate();
        LocalDate localDate = date.toInstant().atZone(ZoneId.systemDefault()).toLocalDate();
        TaiKhoanTietKiem acc = new TaiKhoanTietKiem(stk, ttk, st, localDate);
        TaiKhoanTietKiem_Dao.getInstance().insert(acc);
        this.view2.removeForm();


    }

}
```

Và còn nhiều class khác trong package controller, thầy và các bạn có thể xem kỹ hơn ở phần source code.

5. Package Database và Dao:

Là 2 package có các class dành cho việc kết nối và làm việc với cơ sở dữ liệu.

- **Class JDBC_util** thuộc package Database và class thực hiện việc kết nối tơi cơ sở dữ liệu

```java
public class JDBC_util {
    public static Connection getConnection() {
        Connection c = null;
//      String link_SQL = "C:\\Users\\Dzung\\OneDrive - Đại học Giao thông vận tải\\Desktop\\link_sql.txt";
//      String link = "";
//      try {
//          link = new String(Files.readAllBytes(Paths.get(link_SQL)));
//          JOptionPane.showMessageDialog(null, link);
//      } catch (IOException e) {
//          // TODO Auto-generated catch block
//          e.printStackTrace();
//      }
        try {


            BufferedReader br = new BufferedReader(new FileReader("C:\\\\Users\\\\Dzung\\\\OneDrive - Đại học Giao thông vận tải\\\\Desktop\\\\link_sql

            //String url ="jdbc:mySQL://localhost:3306/qltknh";
            String url= br.readLine();
            String username = br.readLine();
            String password =br.readLine();
            br.close();
            DriverManager.registerDriver(new com.mysql.cj.jdbc.Driver());
            c = DriverManager.getConnection(url, username, password);
//          c = DriverManager.getConnection(link);
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
```

```
    } catch (Exception e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }

     return c;
 }
public static void closeConnection(Connection c) {
    try {
        if(c != null) {
            c.close();
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
 }
public static void prinnInfor(Connection c) {
    try {
        if(c != null) {
            java.sql.DatabaseMetaData mtdt = c.getMetaData();
            System.out.println(c.getMetaData().toString());
            }
    } catch (Exception e) {
        e.printStackTrace();
    }
 }

}
```

Hình 5.1: mô tả các đoạn code kết nối csdl trong class JDBC_uitil

- **Các class thuộc package** DAO là các class thực hiện việc truy vấn đến cơ sở dữ liệu với các phương thức được implements từ lớp interface Dao interface.

  Gồm các phương thức như:

  - Insert là phương thức thêm dữ liệu vào bảng trong csdl
  - Update là phương thức cập nhật lại dữ liệu đã được thêm vào csdl trc đó
  - Delete là xóa dữ liệu đã được thêm vào csdl trc đó
  - SelectAll là tất cả dữ liệu trong một bảng
  - SelectById là lấy dữ liệu của 1 hàng trong bảng theo ID
  - SelectByCondition là lấy dữ liệu của 1 bảng theo 1 điều kiện nào đó.

```java
/* các phương thức truy vấn csdl */

public interface Dao_interface<T> {
 public int insert(T t);

 public int update(T t);

 public int delete(T t);

 public ArrayList<T> selectAll();

 public T selectByID(T t);

 public ArrayList<T> selectByCondition(String condition);
}
```

Hình 5.2: mô tả các phương thức để truy vấn csdl

Một vài hình ảnh về các class thuộc package DAO

```java
public class NguoiDung_Dao implements Dao_interface<NguoiDung>{

    public static NguoiDung_Dao getInstance() {
        return new NguoiDung_Dao();
    }

    @Override
    public int insert(NguoiDung t) {
        int ketQua = 0;
        try {
            Connection cn = JDBC_util.getConnection();
            Statement st = cn.createStatement();

            String sql = "INSERT INTO nguoidung (cccd, tennguoidung, sdt, ngaysinh, taikhoan) " +
                         "VALUES('" +t.getCCCD() +"', '"+t.getTenNguoidung()+ "', '"+t.getSdt() +"', '"+t.getNgaySinh()+"', '"+t.getSoTk_nguoidung()+"')";

             ketQua = st.executeUpdate(sql);


            System.out.println("Bạn đã thực thi: "+ sql);
            System.out.println("Có "+ ketQua+" dòng bị thay đổi!");


            JDBC_util.closeConnection(cn);
        } catch (SQLException e) {
            e.printStackTrace();
        }

        return ketQua;
    }

    @Override
    public int update(NguoiDung t) {
        int ketQua = 1;
        try {
            Connection cn = JDBC_util.getConnection();
            Statement st = cn.createStatement();

            String sql = "UPDATE nguoidung " +
                         "SET "   +
```

```java
@Override
public int update(NguoiDung t) {
    int ketQua = 1;
    try {
        Connection cn = JDBC_util.getConnection();
        Statement st = cn.createStatement();

        String sql = "UPDATE nguoidung " +
                     "SET "   +
                     "cccd= '"+t.getCCCD()+"'" +
                     ", tennguoidung = '" +t.getTenNguoidung()+"'" +
                     ", sdt = '" +t.getSdt()+"'" +
                     ", ngaysinh = '" +t.getNgaySinh()+"'" +
                     ", taikhoan = '"+t.getSoTk_nguoidung()+"'" +
                     "WHERE cccd = '"+t.getCCCD()+"\'";
         ;
         ketQua = st.executeUpdate(sql);

        System.out.println("Bạn đã thực thi: "+ sql);
        System.out.println("Có "+ ketQua+" dòng bị thay đổi!");


        JDBC_util.closeConnection(cn);
    } catch (SQLException e) {
        e.printStackTrace();
    }

    return ketQua;
}
```

```java
@Override
public int delete(NguoiDung t) {

    int ketQua = 0;
    try {
        // Buớc 1: tạo kết nối đến CSDL
        Connection con = JDBC_util.getConnection();

        // Buớc 2: tạo ra đối tượng statement
        Statement st = con.createStatement();

        // Buớc 3: thực thi câu lệnh SQL

        String sql = "DELETE from nguoidung "+
                     " WHERE cccd='"+t.getCCCD()+"'";
        System.out.println(sql);
        ketQua = st.executeUpdate(sql);

        // Buớc 4:
        System.out.println("Bạn đã thực thi: "+ sql);
        System.out.println("Có "+ ketQua+" dòng bị thay đổi!");

        // Buớc 5:
        JDBC_util.closeConnection(con);
    } catch (SQLException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }

    return ketQua;
}
```

```java
@Override
public ArrayList<NguoiDung> selectAll() {
    ArrayList ketQua = new ArrayList();
    try {
        // Bước 1: tạo kết nối đến CSDL
        Connection con = JDBC_util.getConnection();

        // Bước 2: tạo ra đối tượng statement
        Statement st = con.createStatement();

        // Bước 3: thực thi câu lệnh SQL

        String sql = "SELECT * FROM nguoidung";
        System.out.println(sql);
        ResultSet rs = st.executeQuery(sql);

        // Bước 4:
        while(rs.next()) {
            String cccd = rs.getString("CCCD");
            String tennguoidung = rs.getString("tennguoidung");
            String sdt = rs.getString("sdt");
            LocalDate ngaysinh = rs.getDate("ngaySinh").toLocalDate();
            String taikhoan = rs.getString("taikhoan");


            NguoiDung user = new NguoiDung(cccd, tennguoidung, sdt, ngaysinh, taikhoan);
            ketQua.add(user);
        }

        // Bước 5:
        JDBC_util.closeConnection(con);
    } catch (SQLException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }

    return ketQua;
}
```

```java
@Override
public NguoiDung selectByID(NguoiDung t) {
    NguoiDung ketQua = null;
    try {
        // Bước 1: tạo kết nối đến CSDL
        Connection con = JDBC_util.getConnection();

        // Bước 2: tạo ra đối tượng statement
        Statement st = con.createStatement();

        // Bước 3: thực thi câu lệnh SQL

        String sql = "SELECT * FROM nguoidung WHERE cccd ='" +t.getCCCD()+"'";
        System.out.println(sql);
        ResultSet rs = st.executeQuery(sql);

        // Bước 4:
        while(rs.next()) {
            String cccd = rs.getString("CCCD");
            String tennguoidung = rs.getString("tennguoidung");
            String sdt = rs.getString("sdt");
            LocalDate ngaysinh = rs.getDate("ngaySinh").toLocalDate();
            String taikhoan = rs.getString("taikhoan");


            ketQua = new NguoiDung(cccd, tennguoidung, sdt, ngaysinh, taikhoan);

        }

        // Bước 5:
        JDBC_util.closeConnection(con);
    } catch (SQLException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }

    return ketQua;
}
```

```java
@Override
public ArrayList<NguoiDung> selectByCondition(String condition) {
    ArrayList ketQua = new ArrayList();
    try {
        // Buoc 1: tao ket noi den CSDL
        Connection con = JDBC_util.getConnection();

        // Buoc 2: tao ra doi tuong statement
        Statement st = con.createStatement();

        // Buoc 3: thuc thi cau lenh SQL

        String sql = "SELECT * FROM nguoidung WHERE " + condition;
        System.out.println(sql);
        ResultSet rs = st.executeQuery(sql);

        // Buoc 4:
        while(rs.next()) {
            String cccd = rs.getString("CCCD");
            String tennguoidung = rs.getString("tennguoidung");
            String sdt = rs.getString("sdt");
            LocalDate ngaysinh = rs.getDate("ngaySinh").toLocalDate();
            String taikhoan = rs.getString("taikhoan");


            NguoiDung user = new NguoiDung(cccd, tennguoidung, sdt, ngaysinh, taikhoan);
            ketQua.add(user);
        }

        // Buoc 5:
        JDBC_util.closeConnection(con);
    } catch (SQLException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }

    return ketQua;
}
```

6. *Phần Cơ sở Dữ liệu:*

Trong phần này nhóm chúng em sử dụng Xampp và Heidisql để làm việc với cơ sở dữ liệu.

Sau đây là các bảng trong csdl QuanLyTaiKhoanNganHang:

- Bảng nguoidung với thuộc tính **cccd** được chỉ định là khóa chính:

#	Name	Datatype	Length/Set	Unsigned	Allow N...	Zerofill	Default	Comment	Collation
1	**cccd**	**VARCHAR**	**50**	☐	☐	☐	**No default**		**utf8mb4_unicode_ci**
2	tennguoidung	VARCHAR	50	☐	☑	☐	NULL		utf8mb4_unicode_ci
3	sdt	VARCHAR	50	☐	☑	☐	NULL		utf8mb4_unicode_ci
4	ngaysinh	DATE		☐	☑	☐	NULL		
5	taikhoan	VARCHAR	500	☐	☑	☐	NULL		utf8mb4_unicode_ci

Hình 6.1: mô tả bảng nguoidung trong csdl

- Bảng giaodich với thuộc tính **magiaodich** được chỉ định làm khóa chính:

#	Name	Datatype	Length/Set	Unsigned	Allow N...	Zerofill	Default	Comment	Collation	Ex
1	**magiaodich**	**VARCHAR**	**50**	☐	☐	☐	**No default**		**utf8mb4_unicode_ci**	
2	sotien	DOUBLE		☐	☑	☐	'0'			
3	thoigiangiaod...	DATETIME		☐	☑	☐	NULL			
4	taikhoangiaod...	VARCHAR	50	☐	☑	☐	NULL		utf8mb4_unicode_ci	
5	taikhoannhan	VARCHAR	50	☐	☑	☐	NULL		utf8mb4_unicode_ci	
6	loaigiaodich	VARCHAR	50	☐	☑	☐	NULL		utf8mb4_unicode_ci	

Hình 6.2: mô tả bảng giaodich trong csdl

- Bảng taikhoanthanhtoan, bảng taikhoantietkiem, bảng taikhoanvayvon có thuộc tính **sotaikhoan** làm khóa chính:

#	Name	Datatype	Length/Set	Unsigned	Allow N...	Zerofill	Default	Comment	Collation	Expression	Virtuality
1	**sotaikhoan**	**VARCHAR**	**50**	☐	☐	☐	**No default**		**utf8mb4_unicode_ci**		
2	tentaikhoan	VARCHAR	50	☐	☐	☐	No default		utf8mb4_unicode_ci		
3	sodutaikhoan	DOUBLE		☐	☑	☐	NULL				

Hình 6.3: mô tả bảng taikhoanthanhtoan trong csdl

#	Name	Datatype	Length/Set	Unsigned	Allow N...	Zerofill	Default	Comment	Collation	Expression	Virtuality
1	**sotaikhoan**	**VARCHAR**	**50**	☐	☐	☐	**No default**		**utf8mb4_unicode_ci**		
2	tentaikhoan	VARCHAR	50	☐	☐	☐	No default		utf8mb4_unicode_ci		
3	sodutaikhoan	DOUBLE		☐	☑	☐	NULL				
4	ngaygui	DATE		☐	☑	☐	NULL				
5	laisuattietkiem	DOUBLE		☐	☑	☐	NULL				

Hình 6.4: mô tả bảng taikhoantietkiem trong csdl

#	Name	Datatype	Length/Set	Unsigned	Allow N...	Zerofill	Default	Comment	Collation	Expression	Virtuality
1	**sotaikhoan**	**VARCHAR**	**50**	☐	☐	☐	**No default**		**utf8mb4_unicode_ci**		
2	tentaikhoan	VARCHAR	50	☐	☐	☐	No default		utf8mb4_unicode_ci		
3	sodutaikhoan	DOUBLE		☐	☑	☐	'0'				
4	sotienvay	DOUBLE		☐	☑	☐	'0'				
5	sotienlai	DOUBLE		☐	☑	☐	'0'				
6	ngayvay	DATE		☐	☑	☐	NULL				

Hình 6.5: mô tả bảng taikhoanvayvon trong csdl

Lời cảm ơn

Cuối cùng, chúng em xin gửi lời tri ân sâu sắc đến thầy Vũ Huấn. Trong quá trình tìm hiểu và học tập bộ môn, chúng em đã nhận được sự giảng dạy và hướng dẫn rất tận tình, tâm huyết của thầy. Thầy đã giúp chúng em tích lũy thêm nhiều kiến thức hay và bổ ích. Từ những kiến thức mà thầy truyền đạt đã giúp chúng em trình bày được những gì mình đã tìm hiểu về bài báo cáo.

Tuy nhiên, kiến thức về ngôn ngữ Java của chúng em vẫn còn những hạn chế nhất định. Do đó, không thể tránh khỏi những thiếu sót trong quá trình hoàn thành bài báo cáo này. Mong thầy xem và góp ý để bài báo cáo của chúng em được hoàn thiện hơn.

Kính chúc thầy có nhiều sức khỏe, hạnh phúc và thành công hơn nữa trong sự nghiệp “trồng người” để tiếp tục dìu dắt nhiều thế hệ học trò đến những bến bờ tri thức.

Chúng em xin chân thành cảm ơn!