

Etap 5-7

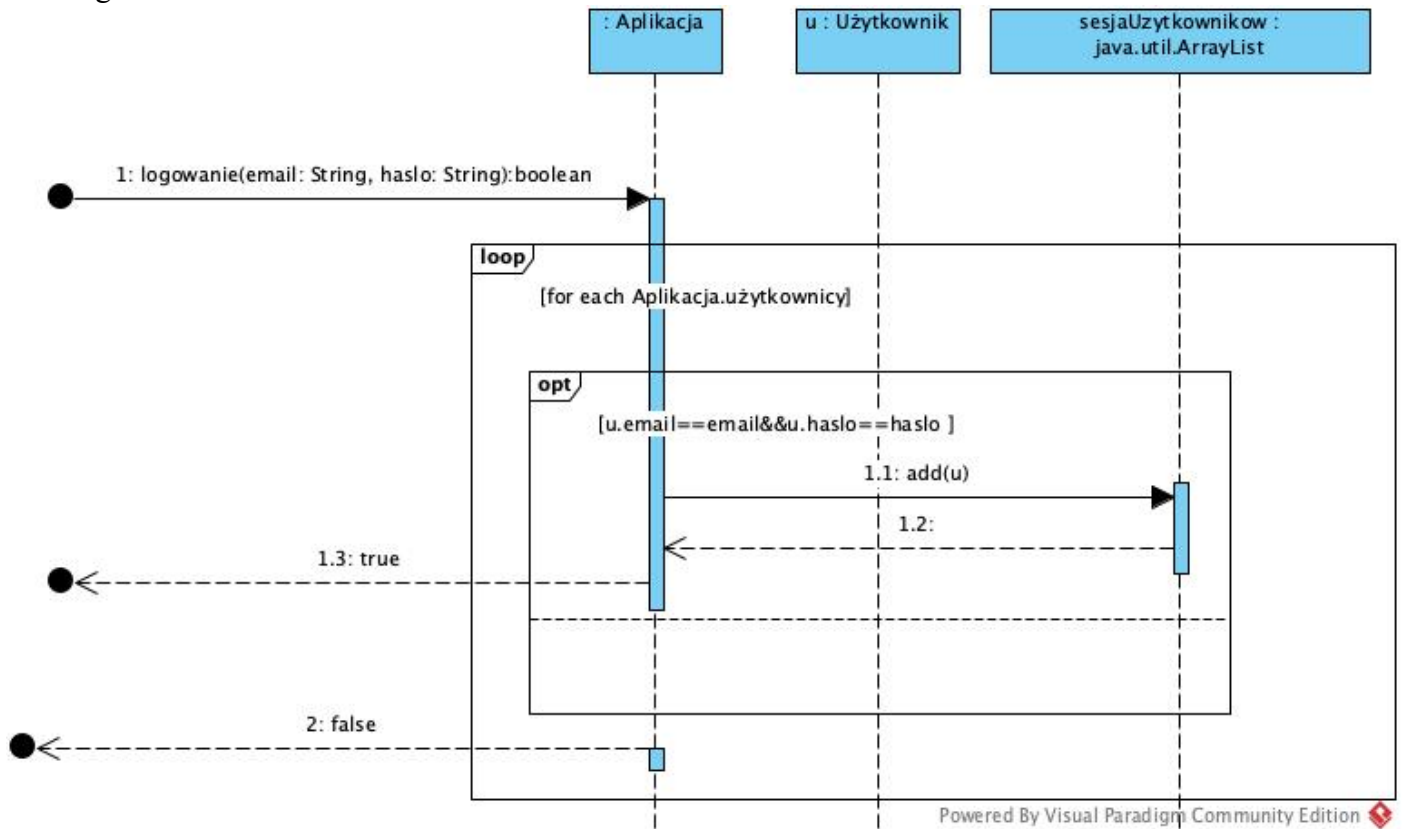
Diagram klas, diagramy sekwencji wraz z kodem, kod źródłowy klas.

1. Diagram klas



2. Diagramy sekwencji

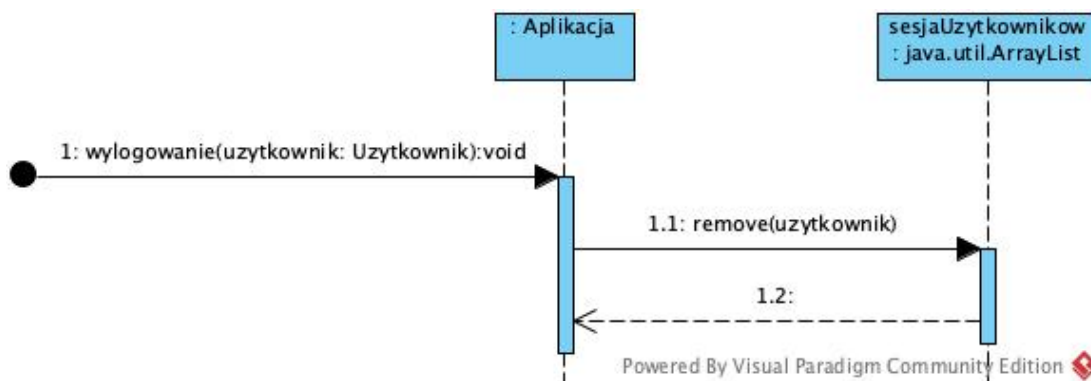
2.1. Logowanie



```

public boolean logowanie (String email, String haslo){
    for(Uzytkownik u : uzytkownicy)
        if(u.email==email&&u.haslo==haslo)
        {
            sesjaUzytkownikow.add(u);
            return true;
        }
    return false;
}
    
```

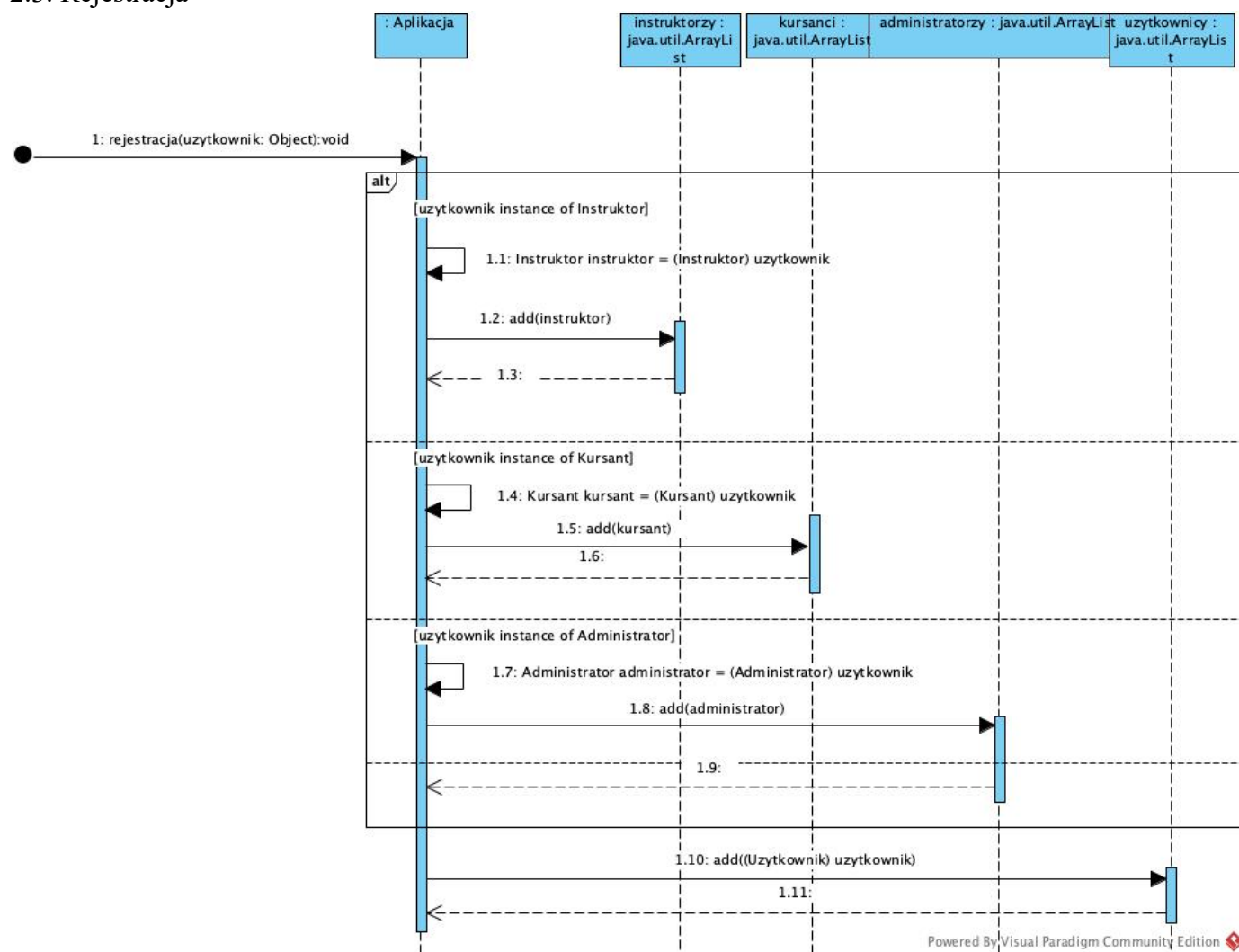
2.2. Wylogowanie



```

public void wylogowanie(Uzytkownik uzytkownik) {
    sesjaUzytkownikow.remove(uzytkownik);
}
    
```

2.3. Rejestracja

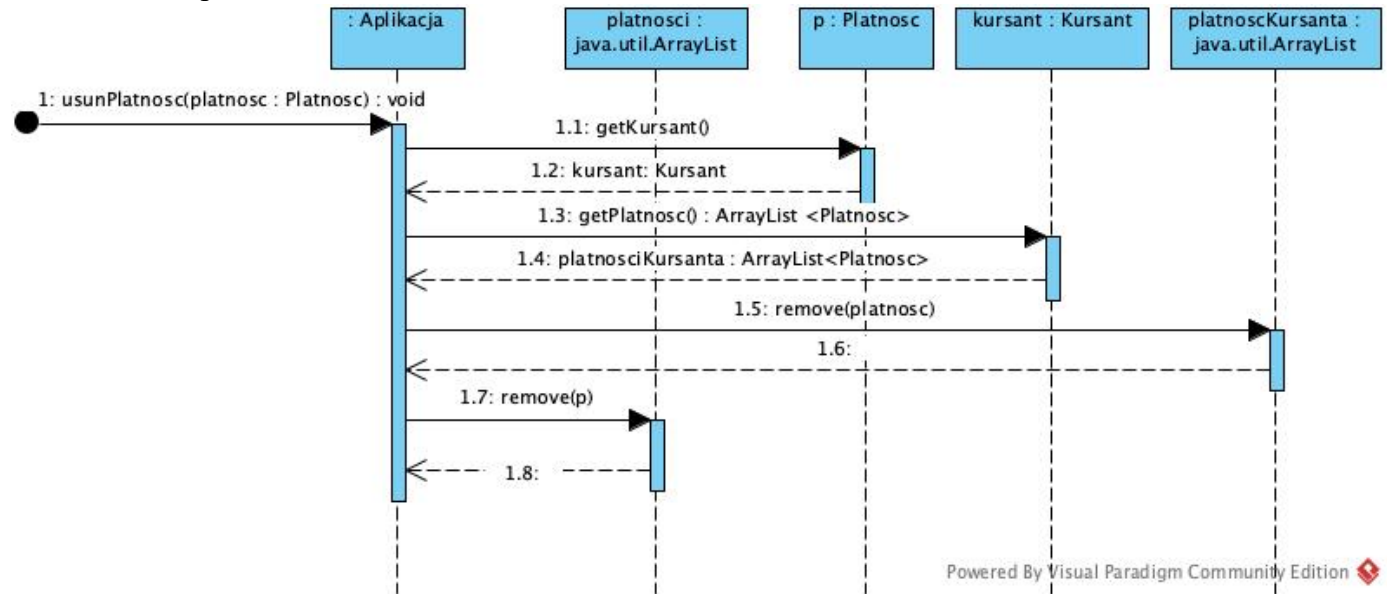


```

public void rejestracja(Object uzytkownik) {
    if(uzytkownik instanceof Instruktor)
    {
        Instruktor instruktor = (Instruktor) uzytkownik;
        instruktorzy.add(instruktor);
    }
    else if(uzytkownik instanceof Kursant){
        Kursant kursant = (Kursant) uzytkownik;
        kursanci.add(kursant);
    }
    else if(uzytkownik instanceof Administrator){
        Administrator administrator = (Administrator) uzytkownik;
        administratorzy.add(administrator);
    }
    uzytkownicy.add((Uzytkownik)uzytkownik);
}

```

2.4. Usuwanie płatności

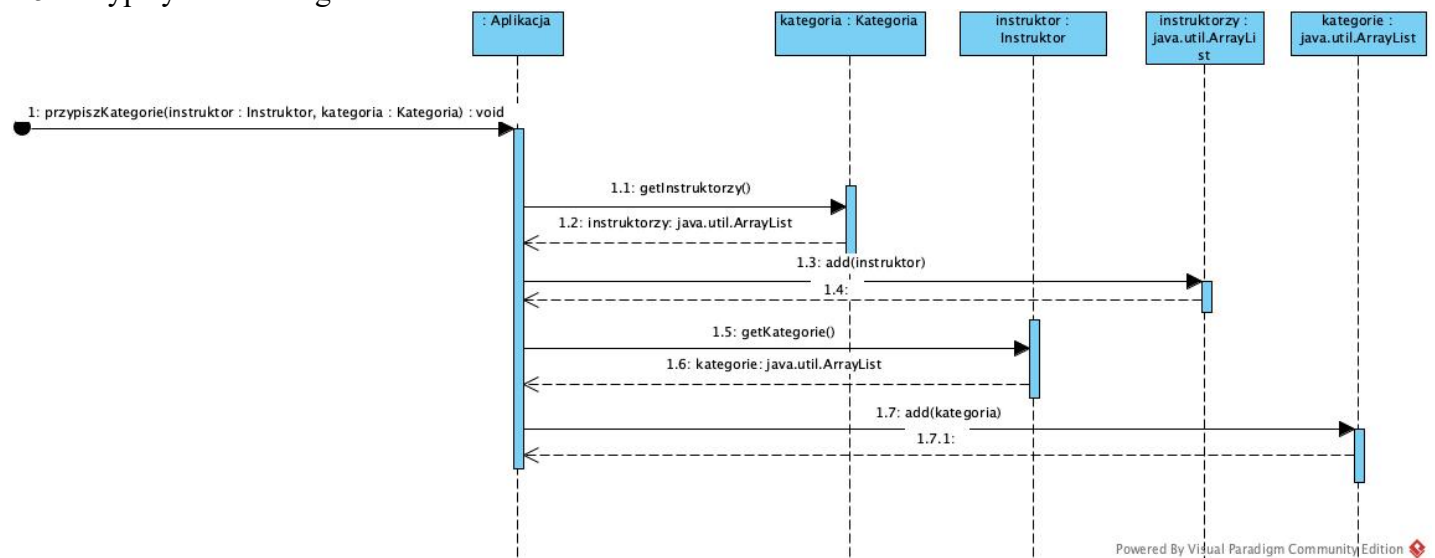


```

public void usunPlatnosc(Platnosc platnosc) {
    platnosc.getKursant().getPlatnosci().remove(platnosc);
    platnosci.remove(platnosc);
}

```

2.5. Przypisywanie kategorii instruktorowi

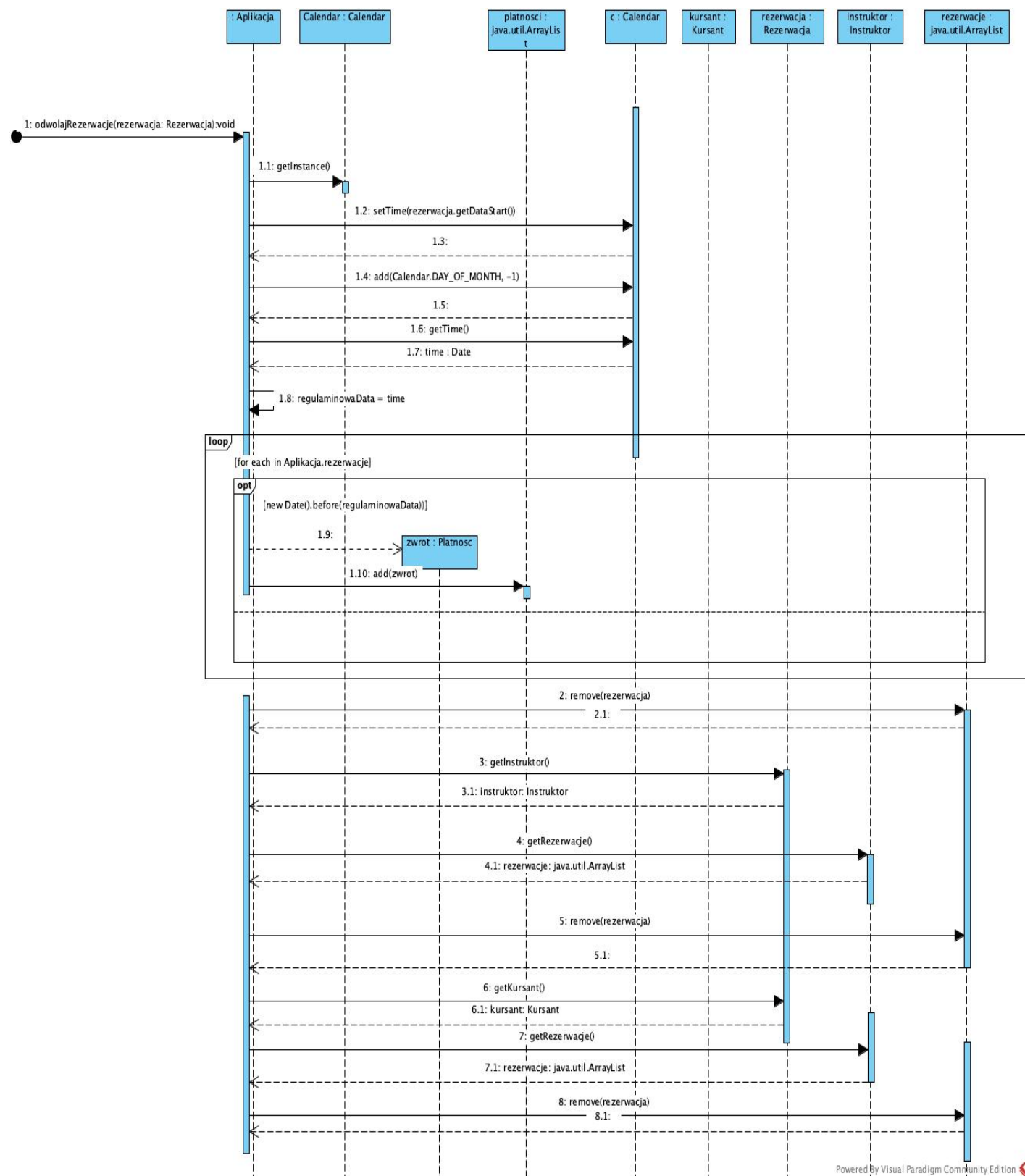


```

public void przypiszKategorie(Kategoria kategoria, Instruktor instruktor) {
    kategoria.getInstruktorzy().add(instruktor);
    instruktor.getKategorieInstruktora().add(kategoria);
}

```

2.6. Odwoływanie rezerwacji

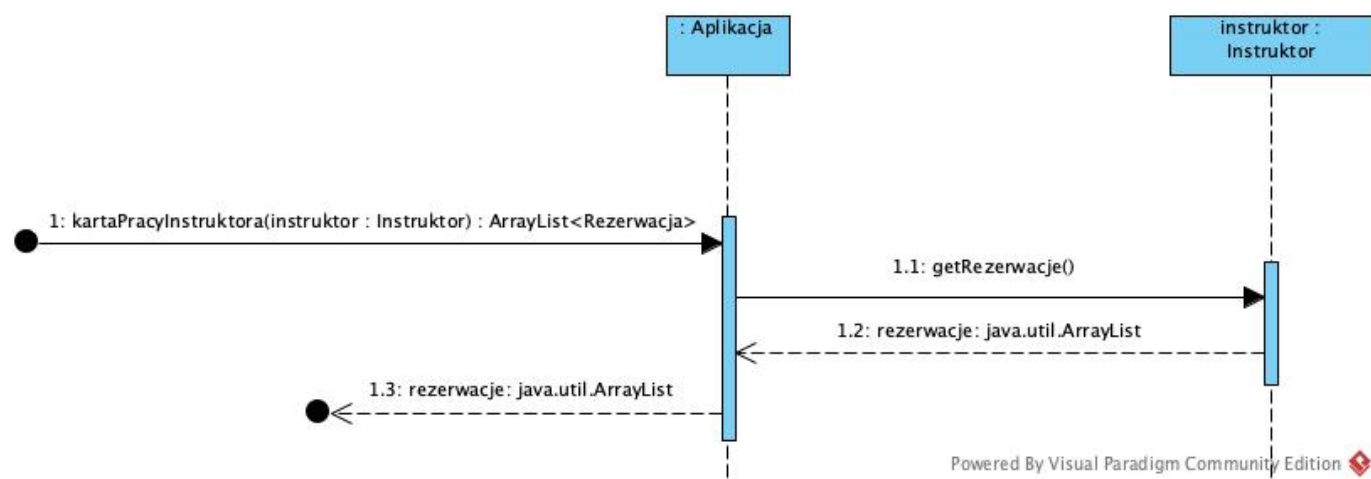


```

public void odwolajRezerwacje(Rezerwacja rezerwacja) {
    Calendar c = Calendar.getInstance();
    c.setTime(rezerwacja.getDataStart());
    c.add(Calendar.DAY_OF_MONTH, -1);
    Date regulaminowaData = c.getTime();
    for (Rezerwacja r : rezerwacje)
        if (new Date().before(regulaminowaData)) {
            Platnosc zwrot = new Platnosc(new Date(), rezerwacja.getKursant(),
            rezerwacja.getIlosc() * rezerwacja.getUsluga().getCena());
            platnosci.add(zwrot);
        }
    rezerwacja.getInstruktor().getRezerwacje().remove(rezerwacja);
    rezerwacja.getKursant().getRezerwacje().remove(rezerwacja);
    rezerwacje.remove(rezerwacja);
}

```

2.7. Karta pracy instruktora

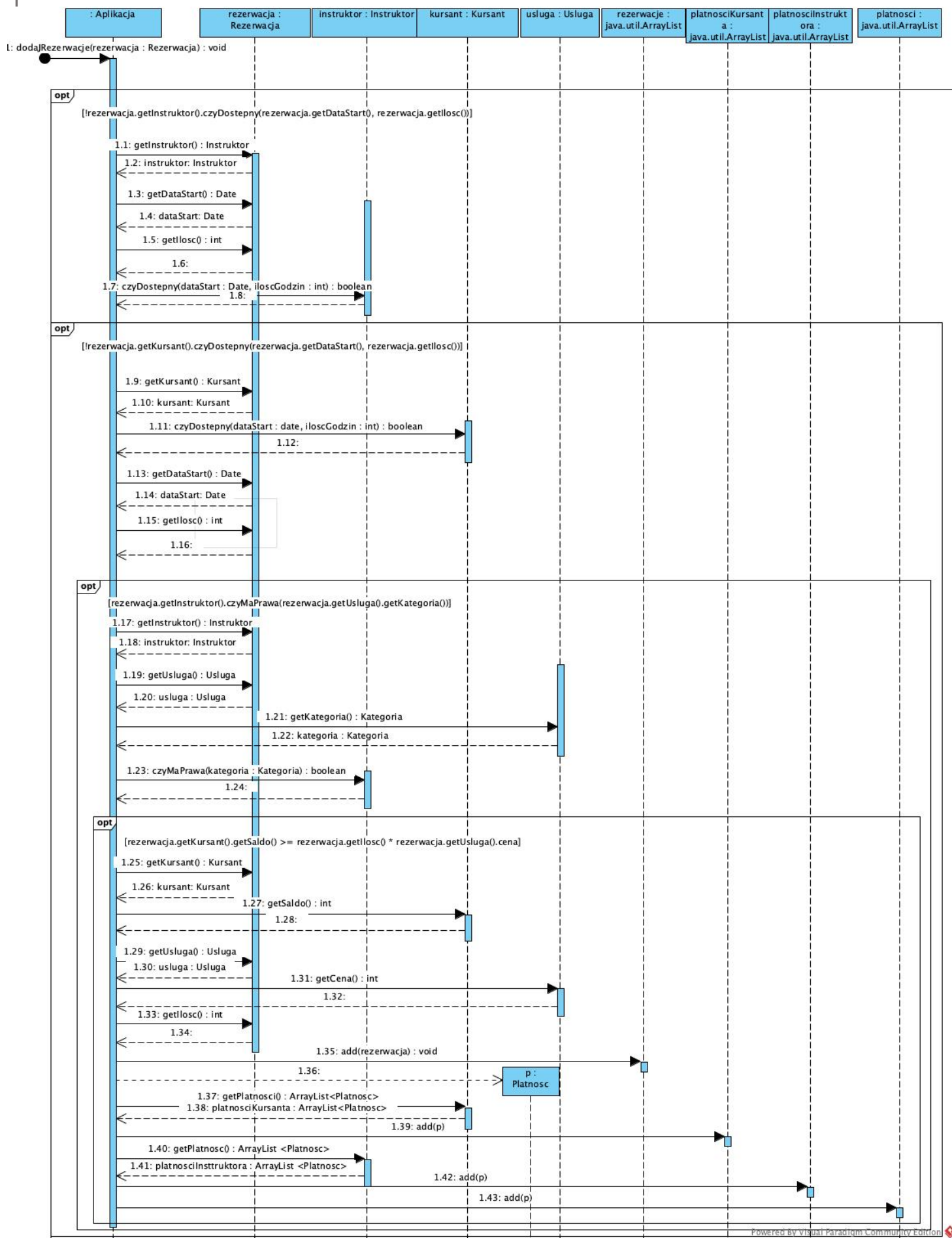


```

public ArrayList<Rezerwacja> kartaPracyInstruktora(Instruktor instruktor) {
    return instruktor.getRezerwacje();
}

```


2.8. Dodawanie rezerwacji

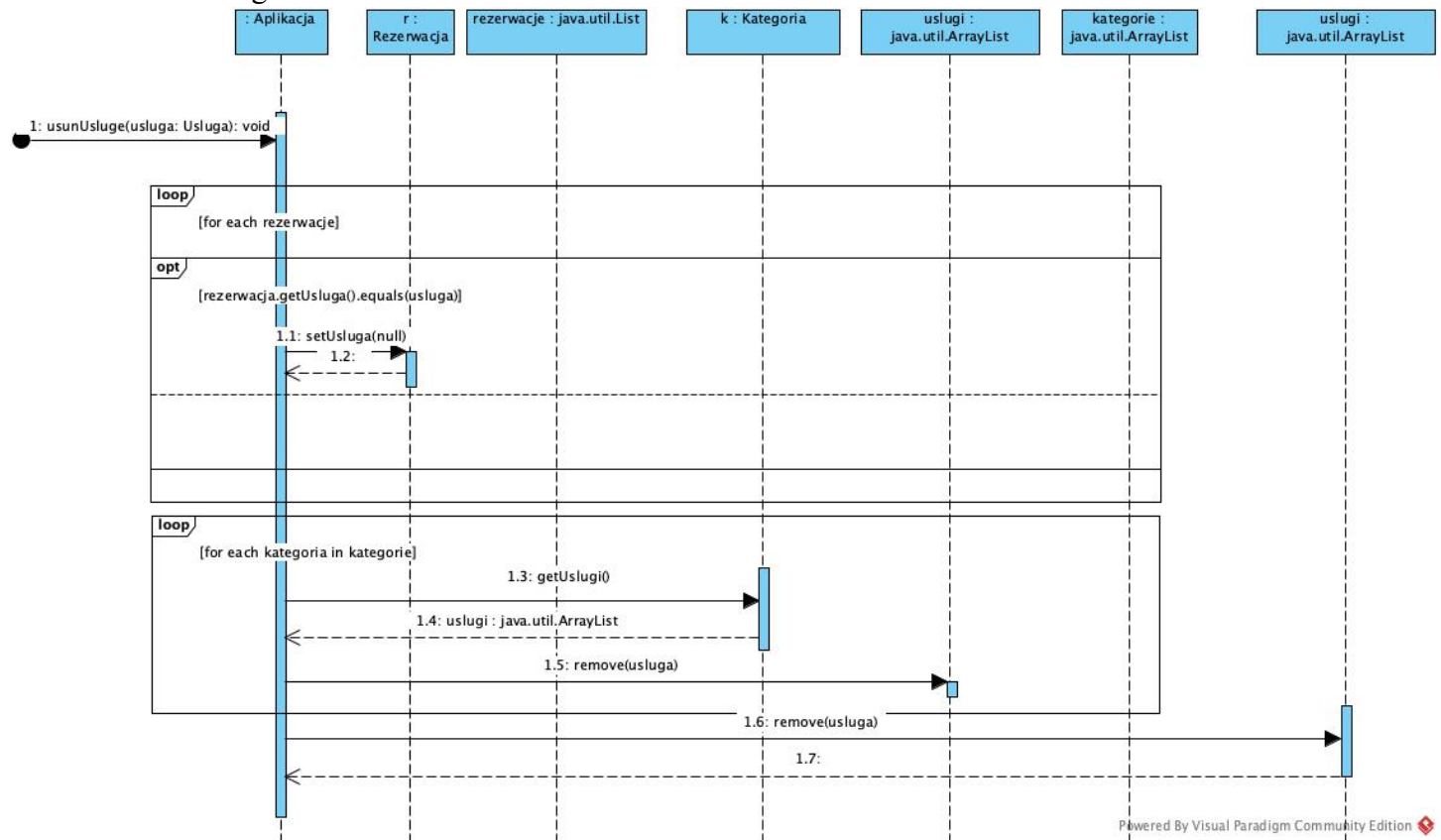


```

public void dodajRezerwacje(Rezerwacja rezerwacja) {
    if (rezerwacja.getInstruktor().czyDostepny(rezerwacja.getDataStart(),
rezerwacja.getIlosc()))
        if (rezerwacja.getKursant().czyDostepny(rezerwacja.getDataStart(),
rezerwacja.getIlosc()))
            if
(rezerwacja.getInstruktor().czyMaPrawa(rezerwacja.getUsluga().getKategoria()))
                if (rezerwacja.getKursant().getSaldo() >= rezerwacja.getIlosc() *
rezerwacja.getUsluga().cena) {
                    rezerwacje.add(rezerwacja);
                    rezerwacja.getInstruktor().getRezerwacje().add(rezerwacja);
                    rezerwacja.getKursant().getRezerwacje().add(rezerwacja);
                    Platnosc p = new Platnosc(new Date(),rezerwacja.kursant,(-
1)*rezerwacja.getUsluga().getCena());
                    rezerwacja.getKursant().getPlatnosci().add(p);
                    platnosci.add(p);
                }
}

```

2.9. Usuwanie usługi

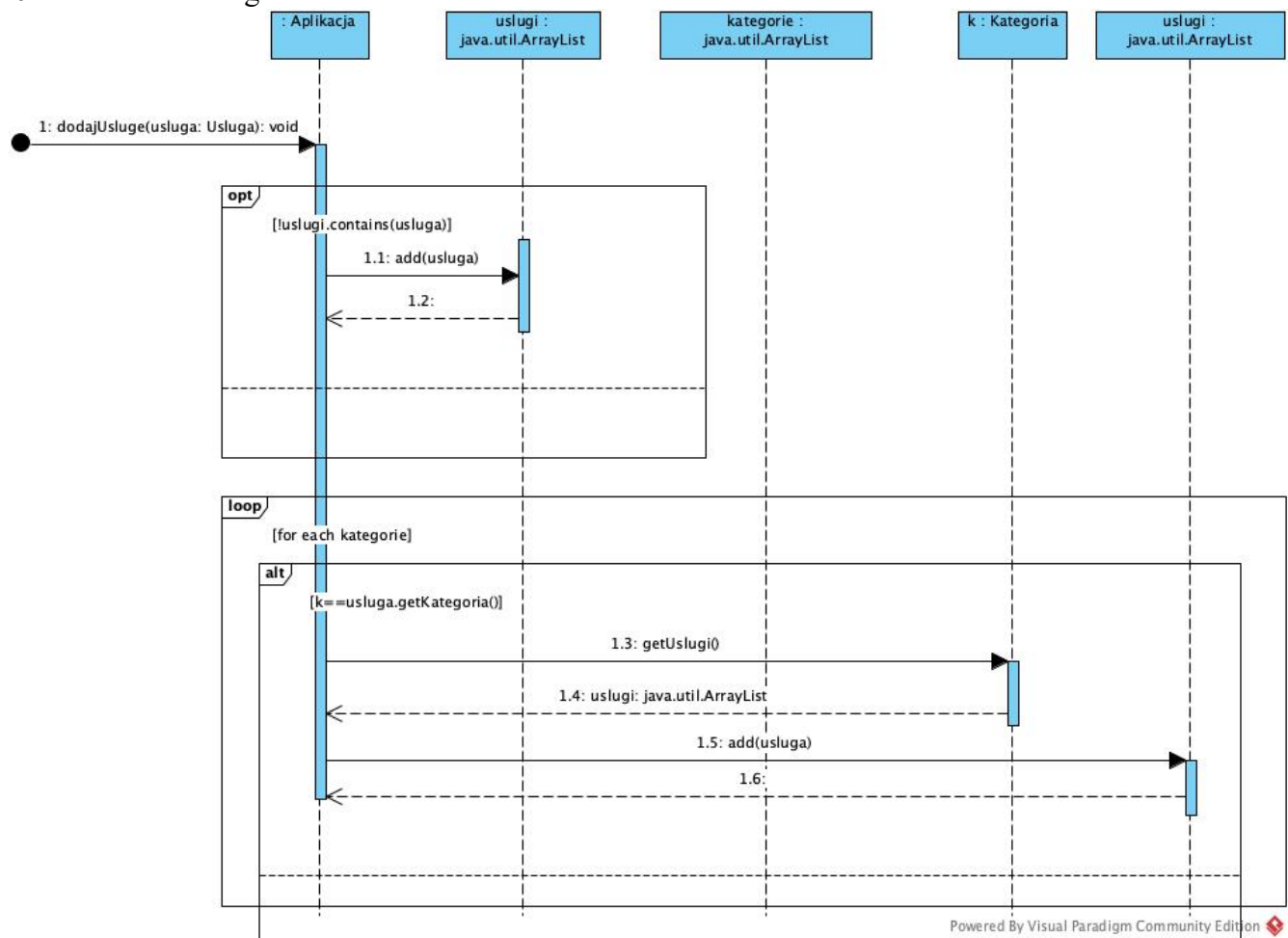


```

public void usunUsluga(Usluga usluga) {
    for (Rezerwacja r : rezerwacje)
        if (r.getUsluga().equals(usluga))
            r.setUsluga(null);
    for (Kategoria k : kategorie)
        k.getUslugi().remove(usluga);
    uslugi.remove(usluga);
}

```

2.10. Dodawanie usługi

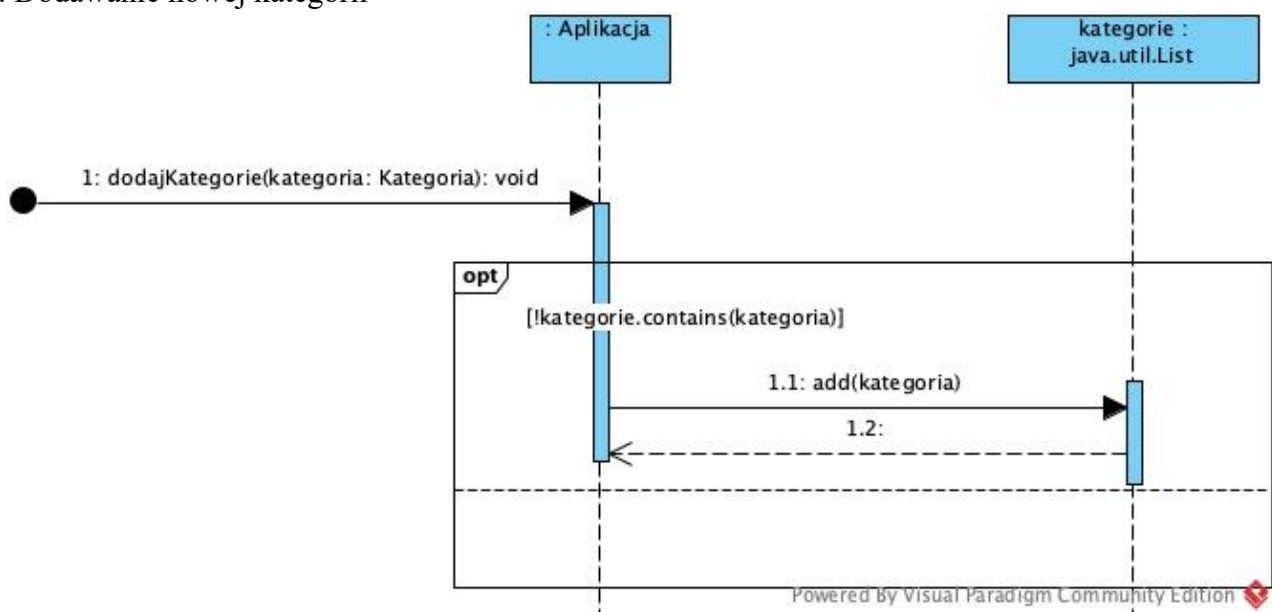


```

public void dodajUsluge(Usluga usluga) {
    if (!uslugi.contains(usluga))
        uslugi.add(usluga);
    for (Kategoria k : kategorie)
        if (k == usluga.getKategoria())
            k.getUslugi().add(usluga);
}

```

2.11. Dodawanie nowej kategorii

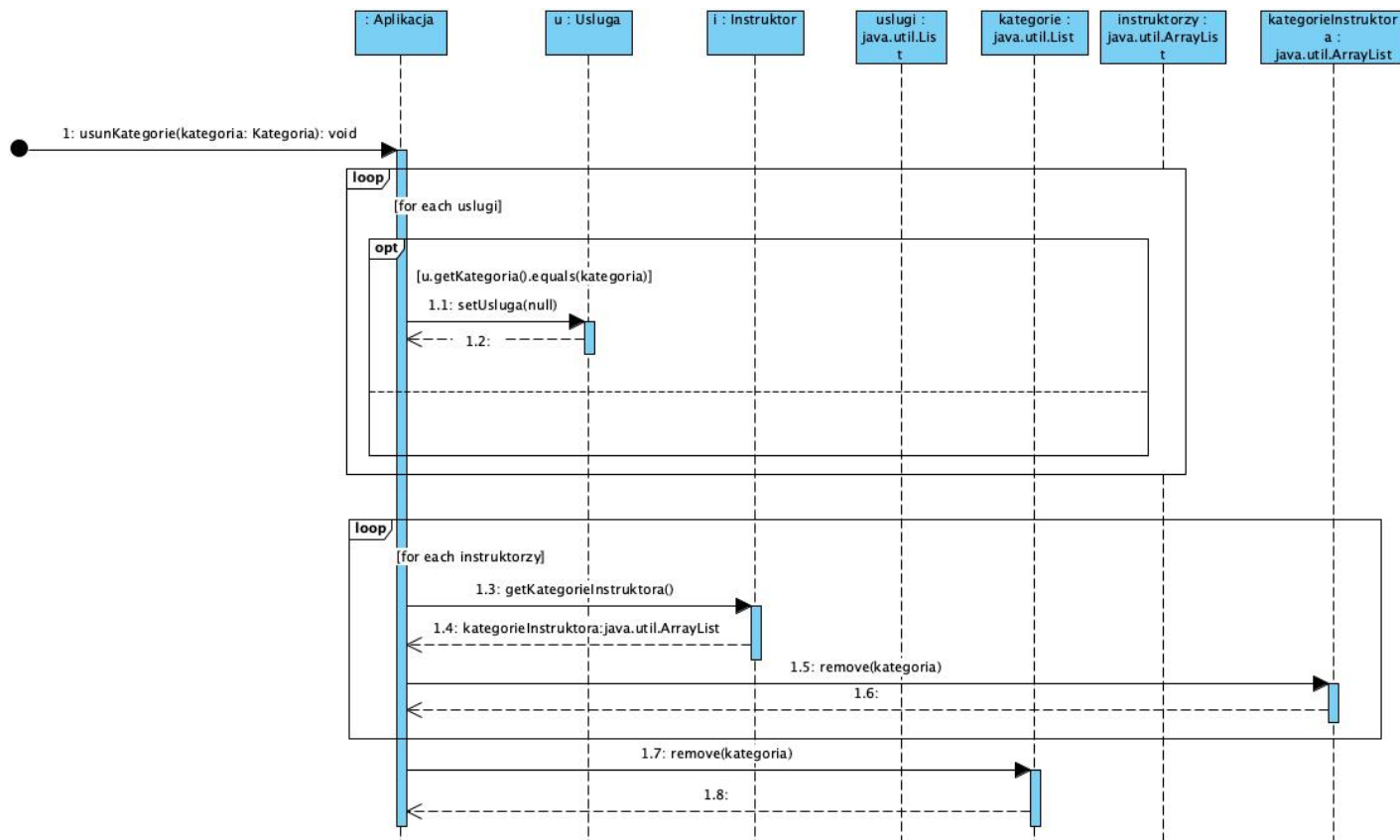


```

public void dodajKategorie(Kategoria kategoria) {
    if (!kategorie.contains(kategoria))
        kategorie.add(kategoria);
}

```

2.12. Usuwanie kategorii

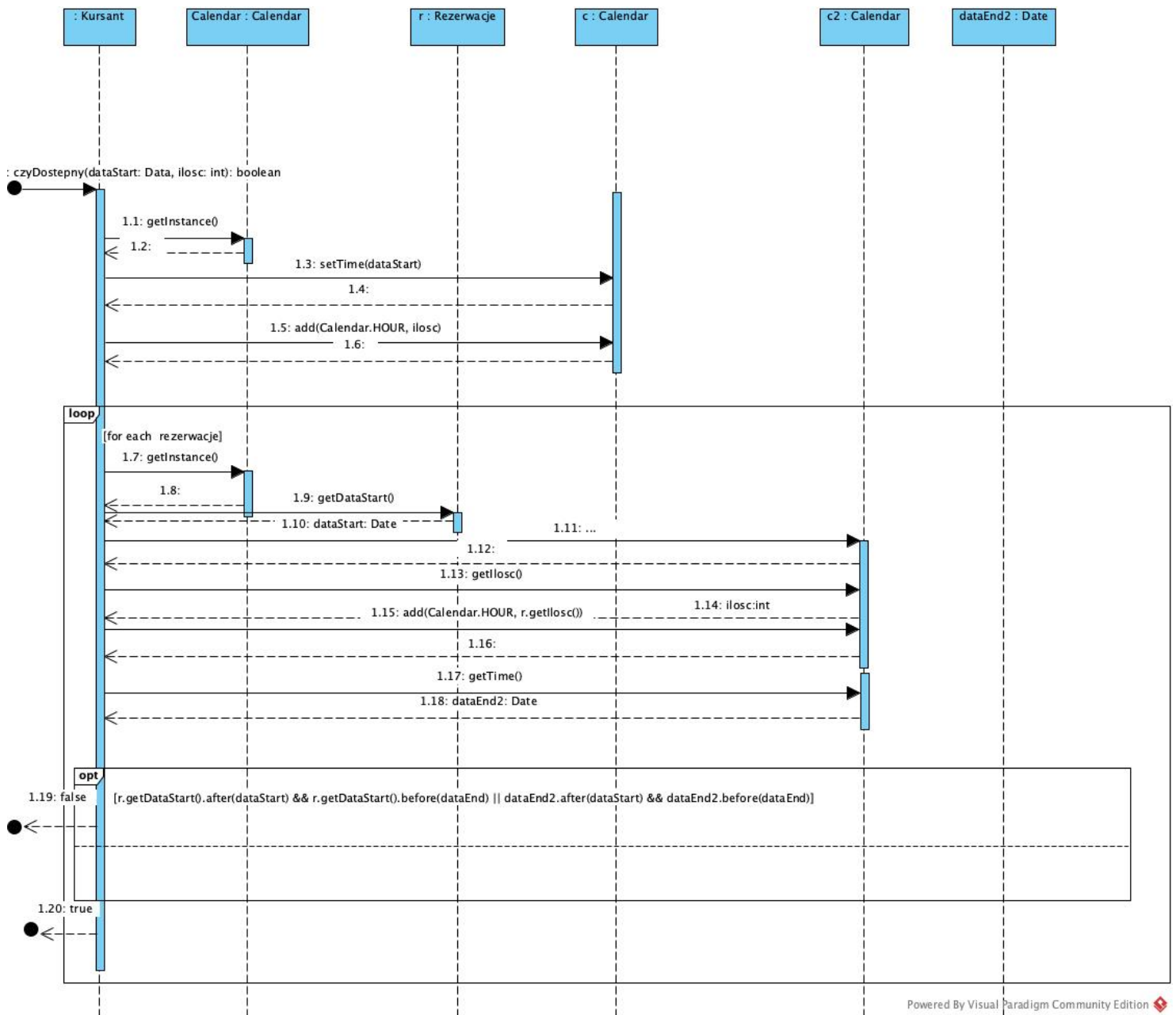


```

public void usunKategorie(Kategoria kategoria) {
    for (Usluga u : uslugi)
        if (u.getKategoria().equals(kategoria))
            u.setKategoria(null);
    for (Instruktor i : instruktorzy)
        i.getKategorieInstruktora().remove(kategoria);
    kategorie.remove(kategoria);
}

```

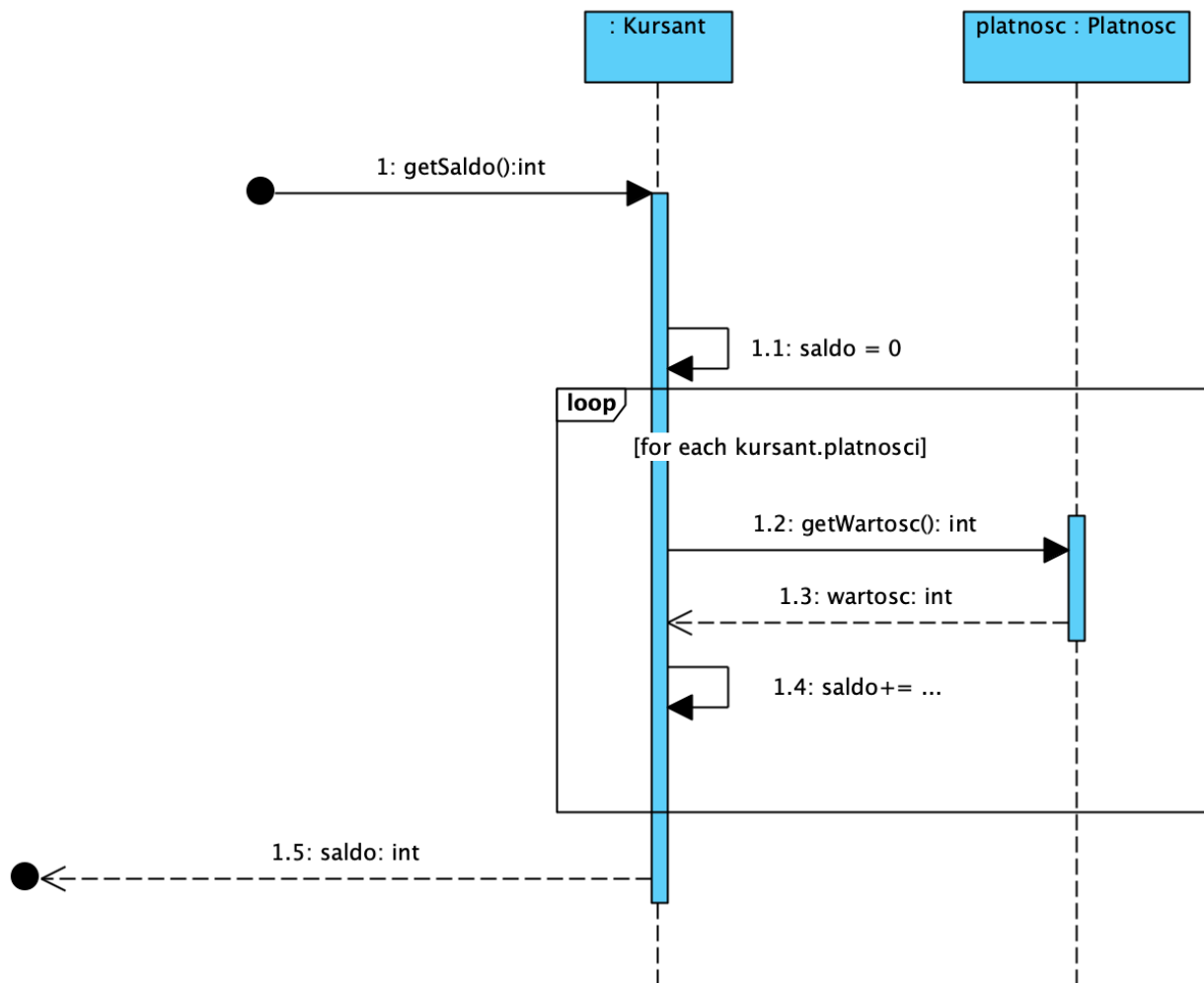
2.13. Sprawdzanie dostępności kursanta



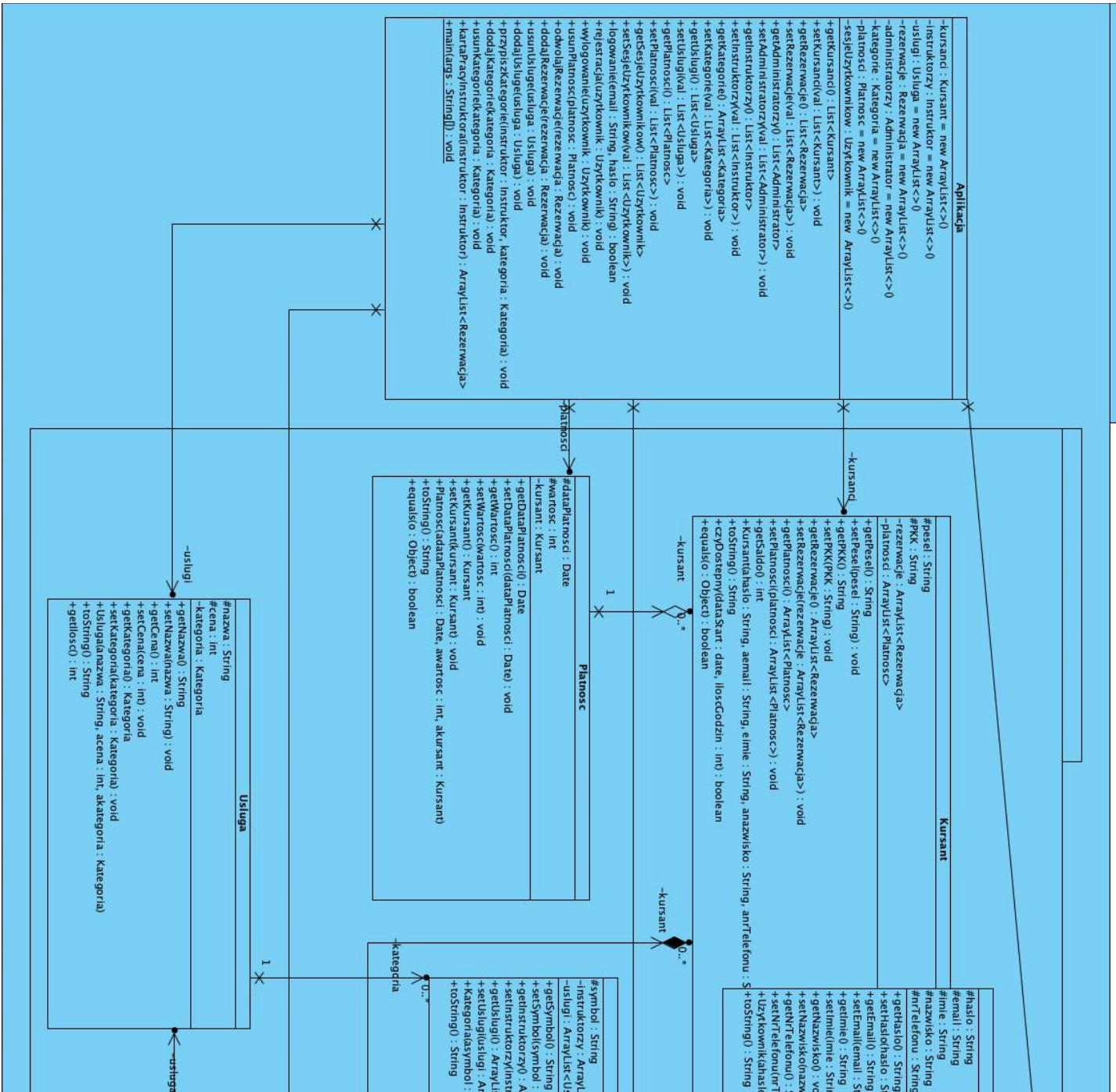
```

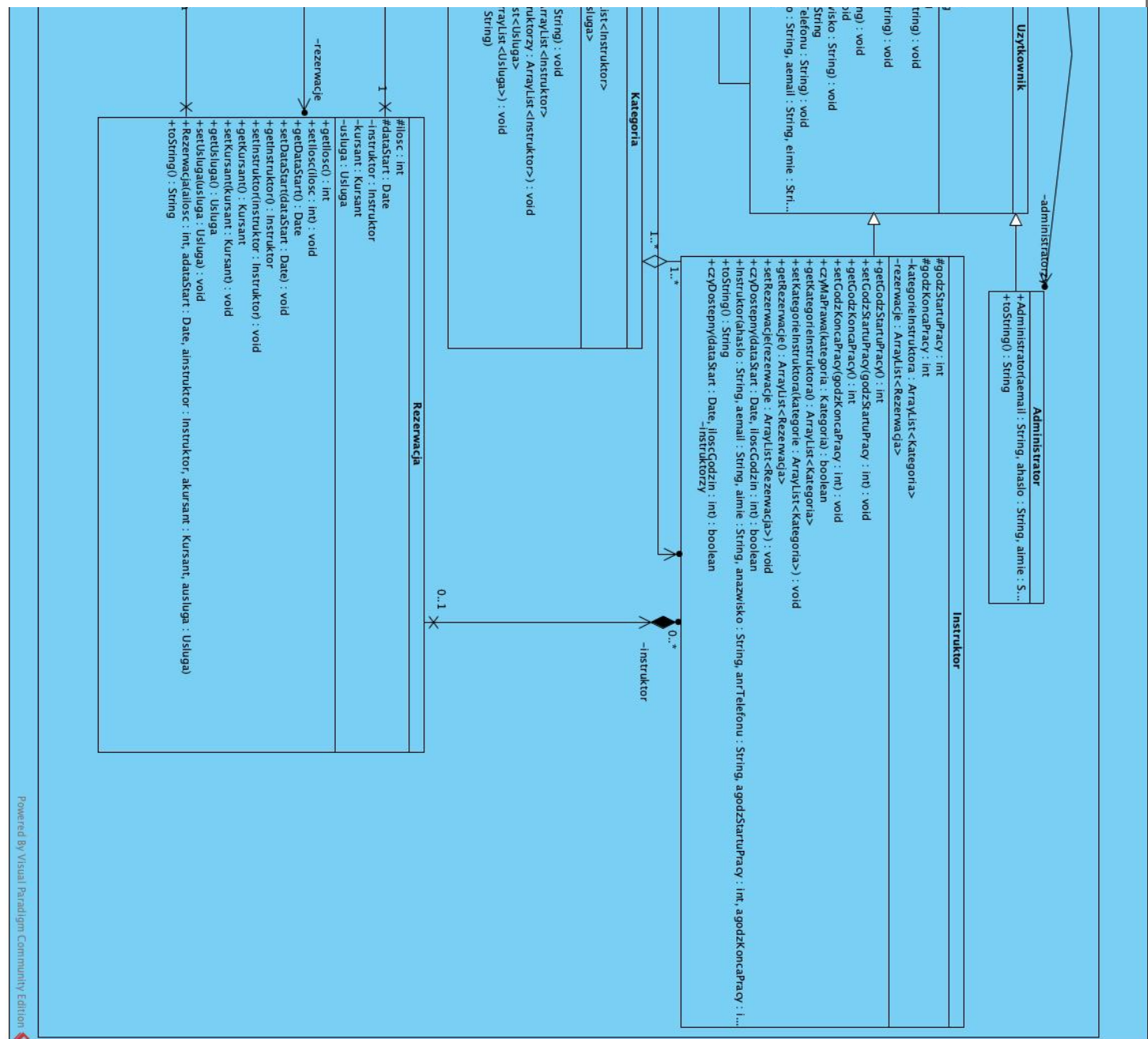
public boolean czyDostepny(Date dataStart, int ilosc) {
    Calendar c = Calendar.getInstance();
    c.setTime(dataStart);
    c.add(Calendar.HOUR, ilosc);
    Date dataEnd = c.getTime();
    for (Rezerwacja r : rezerwacje) {
        Calendar c2 = Calendar.getInstance();
        c2.setTime(r.getDataStart());
        c2.add(Calendar.HOUR, r.getIlosc());
        Date dataEnd2 = c2.getTime();
        if (r.getDataStart().after(dataStart) && r.getDataStart().before(dataEnd) ||
            dataEnd2.after(dataStart) && dataEnd2.before(dataEnd))
            return false;
    }
    return true;
}
  
```

2.14. Sprawdzanie salda kursanta



```
public int getSaldo() {  
    int saldo = 0;  
    for (Platnosc p : platnosci)  
        saldo += p.getWartosc();  
    return saldo;  
}
```



3. Kod klas

Uzytkownik.java

```
import java.util.Objects;

public class Uzytkownik {
    protected String haslo;
    protected String email;
    protected String imie;
    protected String nazwisko;
    protected String nrTelefonu;

    public Uzytkownik(String haslo, String email, String imie, String nazwisko, String nrTelefonu) {
        this.haslo=haslo;
        this.email=email;
        this.imie=imie;
        this.nazwisko=nazwisko;
        this.nrTelefonu=nrTelefonu;
    }

    public String getHaslo() {
        return haslo;
    }

    public void setHaslo(String haslo) {
        this.haslo = haslo;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String getImie() {
        return imie;
    }

    public void setImie(String imie) {
        this.imie = imie;
    }

    public String getNazwisko() {
        return nazwisko;
    }

    public void setNazwisko(String nazwisko) {
        this.nazwisko = nazwisko;
    }

    public String getNrTelefonu() {
        return nrTelefonu;
    }

    public void setNrTelefonu(String nrTelefonu) {
        this.nrTelefonu = nrTelefonu;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Uzytkownik that = (Uzytkownik) o;
        return Objects.equals(haslo, that.haslo) &&
            Objects.equals(email, that.email) &&
            Objects.equals(imie, that.imie) &&
            Objects.equals(nazwisko, that.nazwisko) &&
            Objects.equals(nrTelefonu, that.nrTelefonu);
    }

    @Override
    public String toString() {
        return "Uzytkownik{" +
            "email='" + email + '\'' +
            ", imie='" + imie + '\'' +
            ", nazwisko='" + nazwisko + '\'' +
            ", nrTelefonu='" + nrTelefonu + '\'' +
            '}';
    }
}
```

Usluga.java

```
import java.util.Objects;

public class Usluga {
    protected String nazwa;
    protected int cena;
    private Kategoria kategoria;

    public Usluga(String nazwa, int cena, Kategoria kategoria) {
        this.nazwa = nazwa;
        this.cena = cena;
        this.kategoria = kategoria;
    }

    public String getNazwa() {
        return nazwa;
    }

    public void setNazwa(String nazwa) {
        this.nazwa = nazwa;
    }

    public int getCena() {
        return cena;
    }

    public void setCena(int cena) {
        this.cena = cena;
    }

    public Kategoria getKategoria() {
        return kategoria;
    }

    public void setKategoria(Kategoria kategoria) {
        this.kategoria = kategoria;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Usluga usluga = (Usluga) o;
        return cena == usluga.cena &&
            Objects.equals(nazwa, usluga.nazwa) &&
            Objects.equals(kategoria, usluga.kategoria);
    }

    @Override
    public String toString() {
        return "Usluga{" +
            "nazwa='" + nazwa + '\'' +
            ", cena=" + cena +
            ", kategoria=" + kategoria +
            '}';
    }
}
```

Rezerwacja.java

```
import java.util.Date;
import java.util.Objects;

public class Rezerwacja {
    protected Instruktor instruktor;
    protected Kursant kursant;
    protected Usluga usluga;
    private int ilosc;
    private Date dataStart;

    public Rezerwacja(int ilosc, Date dataStart, Instruktor instruktor, Kursant kursant, Usluga usluga) {
        this.ilosc = ilosc;
        this.dataStart = dataStart;
        this.instruktor = instruktor;
    }
}
```

```

        this.kursant = kursant;
        this.usluga = usluga;
    }

    public int getIlosc() {
        return ilosc;
    }

    public void setIlosc(int ilosc) {
        this.ilosc = ilosc;
    }

    public Date getDataStart() {
        return dataStart;
    }

    public void setDataStart(Date dataStart) {
        this.dataStart = dataStart;
    }

    public Instruktor getInstruktor() {
        return instruktor;
    }

    public void setInstruktor(Instruktor instruktor) {
        this.instruktor = instruktor;
    }

    public Kursant getKursant() {
        return kursant;
    }

    public void setKursant(Kursant kursant) {
        this.kursant = kursant;
    }

    public Usługa getUsługa() {
        return usluga;
    }

    public void setUsługa(Usługa usluga) {
        this.usluga = usluga;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Rezerwacja that = (Rezerwacja) o;
        return ilosc == that.ilosc &&
            Objects.equals(dataStart, that.dataStart) &&
            Objects.equals(instruktor, that.instruktor) &&
            Objects.equals(kursant, that.kursant) &&
            Objects.equals(usluga, that.usluga);
    }

    @Override
    public String toString() {
        return "Rezerwacja{" +
            "ilosc=" + ilosc +
            ", dataStart=" + dataStart +
            ", instruktor=" + instruktor +
            ", kursant=" + kursant +
            ", usluga=" + usluga +
            '}';
    }

```

```

    }
}

Platnosc.java
import java.util.Date;
import java.util.Objects;

public class Platnosc {

    protected Date dataPlatnosci;
    protected int wartosc;
    private Kursant kursant;

    public Platnosc(Date dataPlatnosci, Kursant kursant, int wartosc) {
        this.dataPlatnosci = dataPlatnosci;
        this.kursant = kursant;
        this.wartosc = wartosc;
    }

    public Date getDataPlatnosci() {
        return dataPlatnosci;
    }

    public void setDataPlatnosci(Date dataPlatnosci) {
        this.dataPlatnosci = dataPlatnosci;
    }

    public Kursant getKursant() {
        return kursant;
    }

    public void setKursant(Kursant kursant) {
        this.kursant = kursant;
    }

    public int getWartosc() {
        return wartosc;
    }

    public void setWartosc(int wartosc) {
        this.wartosc = wartosc;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Platnosc platnosc = (Platnosc) o;
        return wartosc == platnosc.wartosc &&
            Objects.equals(dataPlatnosci, platnosc.dataPlatnosci) &&
            Objects.equals(kursant, platnosc.kursant);
    }

    @Override
    public String toString() {
        return "Platnosc{" +
            "dataPlatnosci=" + dataPlatnosci +
            ", kursant=" + kursant +
            ", wartosc=" + wartosc +
            '}';
    }
}

```

Kursant.java

```

import java.util.ArrayList;
import java.util.Calendar;
import java.util.Date;
import java.util.Objects;

public class Kursant extends Uzytkownik {
    protected String pesel;
    protected String PKK;
    private ArrayList<Rezerwacja> rezerwacje = new ArrayList<Rezerwacja>();
    private ArrayList<Platnosc> platnosci = new ArrayList<Platnosc>();

    public Kursant(String haslo, String email, String imie, String nazwisko, String nrTelefonu, String pesel, String PKK) {

```

```

        super(haslo, email, imie, nazwisko, nrTelefonu);
        this.pesel = pesel;
        this.PKK = PKK;
    }

    public boolean czyDostepny(Date dataStart, int ilosc) {
        Calendar c = Calendar.getInstance();
        c.setTime(dataStart);
        c.add(Calendar.HOUR, ilosc);
        Date dataEnd = c.getTime();
        for (Rezerwacja r : rezerwacje) {
            Calendar c2 = Calendar.getInstance();
            c2.setTime(r.getDataStart());
            c2.add(Calendar.HOUR, r.getIlosc());
            Date dataEnd2 = c2.getTime();
            if (r.getDataStart().after(dataStart) && r.getDataStart().before(dataEnd) ||
                dataEnd2.after(dataStart) && dataEnd2.before(dataEnd))
                return false;
        }
        return true;
    }

    public ArrayList<Rezerwacja> getRezerwacje() {
        return rezerwacje;
    }

    public void setRezerwacje(ArrayList<Rezerwacja> rezerwacje) {
        this.rezerwacje = rezerwacje;
    }

    public ArrayList<Platnosc> getPlatnosci() {
        return platnosci;
    }

    public void setPlatnosci(ArrayList<Platnosc> platnosci) {
        this.platnosci = platnosci;
    }

    public int getSaldo() {
        int saldo = 0;
        for (Platnosc p : platnosci)
            saldo += p.getWartosc();
        return saldo;
    }

    public String getPesel() {
        return pesel;
    }

    public void setPesel(String pesel) {
        this.pesel = pesel;
    }

    public String getPKK() {
        return PKK;
    }

    public void setPKK(String PKK) {
        this.PKK = PKK;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        if (!super.equals(o)) return false;
        Kursant kursant = (Kursant) o;
        return Objects.equals(pesel, kursant.pesel) &&
            Objects.equals(PKK, kursant.PKK);
    }

    @Override
    public String toString() {
        return "Kursant{" +
            "pesel='" + pesel + '\'' +
            ", PKK='" + PKK + '\'' +
            ", email='" + email + '\'' +
            ", imie='" + imie + '\'' +

```

```

        ", nazwisko='" + nazwisko + '\'' +
        ", nrTelefonu='" + nrTelefonu + '\'' +
        '}}';
    }
}

```

Kategoria.java

```

import java.util.ArrayList;

public class Kategoria {
    protected String symbol;
    private ArrayList<Instruktor> instruktorzy = new ArrayList<Instruktor>();
    private ArrayList<Usługa> usługi = new ArrayList<Usługa>();

    public Kategoria(String nazwa) {
        this.symbol = nazwa;
    }

    public ArrayList<Instruktor> getInstruktorzy() {
        return instruktorzy;
    }

    public void setInstruktorzy(ArrayList<Instruktor> instruktorzy) {
        this.instruktorzy = instruktorzy;
    }

    public String getSymbol() {
        return symbol;
    }

    public void setSymbol(String symbol) {
        this.symbol = symbol;
    }

    public ArrayList<Usługa> getUsługi() {
        return usługi;
    }

    public void setUsługi(ArrayList<Usługa> usługi) {
        this.usługi = usługi;
    }

    @Override
    public boolean equals(Object o) {
        return ((Kategoria) o).getSymbol() == this.symbol;
    }

    @Override
    public String toString() {
        return "Kategoria{" +
            "nazwa='" + symbol + '\'' +
            ", instruktorzy=" + instruktorzy +
            '}}';
    }
}

```

Instruktor.java

```

import java.util.ArrayList;
import java.util.Calendar;
import java.util.Date;
import java.util.Objects;

public class Instruktor extends Uzytkownik {
    protected int godzStartuPracy;
    protected int godzKoncaPracy;
    private ArrayList<Rezerwacja> rezerwacje = new ArrayList<Rezerwacja>();
    private ArrayList<Kategoria> kategorieInstruktora = new ArrayList<Kategoria>();

    public Instruktor(String haslo, String email, String imie, String nazwisko, String nrTelefonu, int
godzStartuPracy, int godzKoncaPracy) {
        super(haslo, email, imie, nazwisko, nrTelefonu);
        this.godzStartuPracy = godzStartuPracy;
        this.godzKoncaPracy = godzKoncaPracy;
    }
}

```

```

public boolean czyDostepny(Date dataStart, int ilosc) {
    Calendar c = Calendar.getInstance();
    c.setTime(dataStart);
    c.add(Calendar.HOUR, ilosc);
    Date dataEnd = c.getTime();
    for (Rezerwacja r : rezerwacje) {
        Calendar c2 = Calendar.getInstance();
        c2.setTime(r.getDataStart());
        c2.add(Calendar.HOUR, r.getIlosc());
        Date dataEnd2 = c2.getTime();
        if (r.getDataStart().after(dataStart) && r.getDataStart().before(dataEnd) ||
            dataEnd2.after(dataStart) && dataEnd2.before(dataEnd))
            return false;
    }
    return true;
}

public boolean czyMaPrawa(Kategoria kategoria) {
    for (Kategoria k : kategorieInstruktora)
        if (kategoria.equals(k))
            return true;
    return false;
}

public int getGodzStartuPracy() {
    return godzStartuPracy;
}

public void setGodzStartuPracy(int godzStartuPracy) {
    this.godzStartuPracy = godzStartuPracy;
}

public int getGodzKoncaPracy() {
    return godzKoncaPracy;
}

public void setGodzKoncaPracy(int godzKoncaPracy) {
    this.godzKoncaPracy = godzKoncaPracy;
}

public ArrayList<Rezerwacja> getRezerwacje() {
    return rezerwacje;
}

public void setRezerwacje(ArrayList<Rezerwacja> rezerwacje) {
    this.rezerwacje = rezerwacje;
}

public ArrayList<Kategoria> getKategorieInstruktora() {
    return kategorieInstruktora;
}

public void setKategorieInstruktora(ArrayList<Kategoria> kategorieInstruktora) {
    this.kategorieInstruktora = kategorieInstruktora;
}

@Override
public boolean equals(Object o) {
    if (this == o) return true;
    if (o == null || getClass() != o.getClass()) return false;
    if (!super.equals(o)) return false;
    Instruktor that = (Instruktor) o;
    return godzStartuPracy == that.godzStartuPracy &&
        godzKoncaPracy == that.godzKoncaPracy &&
        Objects.equals(rezerwacje, that.rezerwacje) &&
        Objects.equals(kategorieInstruktora, that.kategorieInstruktora);
}

@Override
public String toString() {
    return "Instruktor{" +
        " imie='" + imie + '\'' +
        ", nazwisko='" + nazwisko + '\'' +
        ", nrTelefonu='" + nrTelefonu + '\'' +
        ", godzStartuPracy=" + godzStartuPracy +
        ", godzKoncaPracy=" + godzKoncaPracy +
        ", email='" + email + '\'' +
        '}';
}
}

```

Aplikacja.java

```
import java.util.ArrayList;
import java.util.Calendar;
import java.util.Date;

public class Aplikacja {

    private boolean zalogowany=false;
    private ArrayList<Kursant> kursanci = new ArrayList<>();
    private ArrayList<Instruktor> instruktorzy = new ArrayList<>();
    private ArrayList<Usługa> usługi = new ArrayList<>();
    private ArrayList<Rezerwacja> rezerwacje = new ArrayList<>();
    private ArrayList<Administrator> administratorzy = new ArrayList<>();
    private ArrayList<Kategoria> kategorie = new ArrayList<>();
    private ArrayList<Platnosc> platnosci = new ArrayList<>();
    private ArrayList<Uzytkownik> uzytkownicy = new ArrayList<>();
    private ArrayList<Uzytkownik> sesjaUzytkownikow = new ArrayList<Uzytkownik>();

    public ArrayList<Kursant> getKursanci() {
        return kursanci;
    }

    public void setKursanci(ArrayList<Kursant> kursanci) {
        this.kursanci = kursanci;
    }

    public ArrayList<Rezerwacja> getRezerwacje() {
        return rezerwacje;
    }

    public void setRezerwacje(ArrayList<Rezerwacja> rezerwacje) {
        this.rezerwacje = rezerwacje;
    }

    public ArrayList<Administrator> getAdministratorzy() {
        return administratorzy;
    }

    public void setAdministratorzy(ArrayList<Administrator> administratorzy) {
        this.administratorzy = administratorzy;
    }

    public ArrayList<Instruktor> getInstruktorzy() {
        return instruktorzy;
    }

    public void setInstruktorzy(ArrayList<Instruktor> instruktorzy) {
        this.instruktorzy = instruktorzy;
    }

    public ArrayList<Kategoria> getKategorie() {
        return kategorie;
    }

    public void setKategorie(ArrayList<Kategoria> kategorie) {
        this.kategorie = kategorie;
    }

    public ArrayList<Usługa> getUsługi() {
        return usługi;
    }

    public void setUsługi(ArrayList<Usługa> usługi) {
        this.usługi = usługi;
    }

    public ArrayList<Platnosc> getPlatnosci() {
        return platnosci;
    }

    public void setPlatnosci(ArrayList<Platnosc> platnosci) {
        this.platnosci = platnosci;
    }

    public boolean logowanie (String email, String haslo){
        for(Uzytkownik u : uzytkownicy)
            if(u.email==email&&u.haslo==haslo)
                {
```



```

        sesjaUzytkownikow.add(u);
        return true;
    }
    return false;
}

public void rejestracja(Object uzytkownik) {
    if(uzytkownik instanceof Instruktor)
    {
        Instruktor instruktor = (Instruktor) uzytkownik;
        instruktorzy.add(instruktor);
    }
    else if(uzytkownik instanceof Kursant){
        Kursant kursant = (Kursant) uzytkownik;
        kursanci.add(kursant);
    }
    else if(uzytkownik instanceof Administrator){
        Administrator administrator = (Administrator) uzytkownik;
        administratorzy.add(administrator);
    }
    uzytkownicy.add((Uzytkownik)uzytkownik);
}

public void wylogowanie(Uzytkownik uzytkownik) {
    sesjaUzytkownikow.remove(uzytkownik);
}

public void usunPlatnosc(Platnosc platnosc) {
    platnosc.getKursant().getPlatnosci().remove(platnosc);
    platnosci.remove(platnosc);
}

public void odwolajRezerwacje(Rezerwacja rezerwacja) {
    Calendar c = Calendar.getInstance();
    c.setTime(rezerwacja.getDataStart());
    c.add(Calendar.DAY_OF_MONTH, -1);
    Date regulaminowaData = c.getTime();
    for (Rezerwacja r : rezerwacje)
        if (new Date().before(regulaminowaData)) {
            Platnosc zwrot = new Platnosc(new Date(), rezerwacja.getKursant(), rezerwacja.getIlosc() *
rezerwacja.getUsługa().getCena());
            platnosci.add(zwrot);
        }
    rezerwacja.getInstruktor().getRezerwacje().remove(rezerwacja);
    rezerwacja.getKursant().getRezerwacje().remove(rezerwacja);
    rezerwacje.remove(rezerwacja);
}

public void dodajRezerwacje(Rezerwacja rezerwacja) {
    if (rezerwacja.getInstruktor().czyDostepny(rezerwacja.getDataStart(), rezerwacja.getIlosc()))
        if (rezerwacja.getKursant().czyDostepny(rezerwacja.getDataStart(), rezerwacja.getIlosc()))
            if (rezerwacja.getInstruktor().czyMaPrawa(rezerwacja.getUsługa().getKategoria()))
                if (rezerwacja.getKursant().getSaldo() >= rezerwacja.getIlosc() *
rezerwacja.getUsługa().cena) {
                    rezerwacje.add(rezerwacja);
                    rezerwacja.getInstruktor().getRezerwacje().add(rezerwacja);
                    rezerwacja.getKursant().getRezerwacje().add(rezerwacja);
                    Platnosc p = new Platnosc(new Date(), rezerwacja.kursant, (-
1)*rezerwacja.getUsługa().getCena());
                    rezerwacja.getKursant().getPlatnosci().add(p);
                    platnosci.add(p);
                }
}

public void usunUslugę(Usługa usługa) {
    for (Rezerwacja r : rezerwacje)
        if (r.getUsługa().equals(usługa))
            r.setUsługa(null);
    for (Kategoria k : kategorie)
        k.getUsługi().remove(usługa);
    uslugi.remove(usługa);
}

public void dodajUslugę(Usługa usługa) {
    if (!uslugi.contains(usługa))
        uslugi.add(usługa);
    for (Kategoria k : kategorie)
        if (k == usługa.getKategoria())
            k.getUsługi().add(usługa);
}

```

```

    public void przypiszKategorie(Kategoria kategoria, Instruktor instruktor) {
        kategoria.getInstruktorzy().add(instruktor);
        instruktor.getKategorieInstruktora().add(kategoria);
    }

    public void dodajKategorie(Kategoria kategoria) {
        if (!kategorie.contains(kategoria))
            kategorie.add(kategoria);
    }

    public void usunKategorie(Kategoria kategoria) {
        for (Usluga u : uslugi)
            if (u.getKategoria().equals(kategoria))
                u.setKategoria(null);
        for (Instruktor i : instruktorzy)
            i.getKategorieInstruktora().remove(kategoria);
        kategorie.remove(kategoria);
    }

    public ArrayList<Rezerwacja> kartaPracyInstruktora(Instruktor instruktor) {
        return instruktor.getRezerwacje();
    }

    public static void main(String[] args) {
}}

```

Administrator.java

```

public class Administrator extends Uzytkownik {

    public Administrator(String haslo, String email, String imie, String nazwisko, String nrTelefonu) {
        super(haslo, email, imie, nazwisko, nrTelefonu);
    }

    @Override
    public String toString() {
        return "Administrator{" +
            "email='" + email + '\'' +
            ", imie='" + imie + '\'' +
            ", nazwisko='" + nazwisko + '\'' +
            ", nrTelefonu='" + nrTelefonu + '\'' +
            '}';
    }
}

```