## **2. Diagramy sekwencji**

2.1. Logowanie

Obraz zawierający tekst, mapa

Opis wygenerowany automatycznie

Obraz zawierający zrzut ekranu, mapa

Opis wygenerowany automatycznie

**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. Rejestracja

Obraz zawierający tekst, mapa

Opis wygenerowany automatycznie

Obraz zawierający mapa, tekst

Opis wygenerowany automatycznie

**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);  
}

2.3. Wylogowanie

Obraz zawierający tekst, mapa

Opis wygenerowany automatycznie

Obraz zawierający zrzut ekranu

Opis wygenerowany automatycznie

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

2.4. Usuwanie płatności

Obraz zawierający tekst, mapa

Opis wygenerowany automatycznie

Obraz zawierający zrzut ekranu

Opis wygenerowany automatycznie

**public void** usunPlatnosc(Platnosc platnosc) {  
 platnosc.getKursant().getPlatnosci().remove(platnosc);  
 **for** (Platnosc p : **platnosci**)  
 **if** (p.equals(platnosc))  
 **platnosci**.remove(p);  
}

2.5. Odwoływanie rezerwacji

Obraz zawierający tekst, mapa

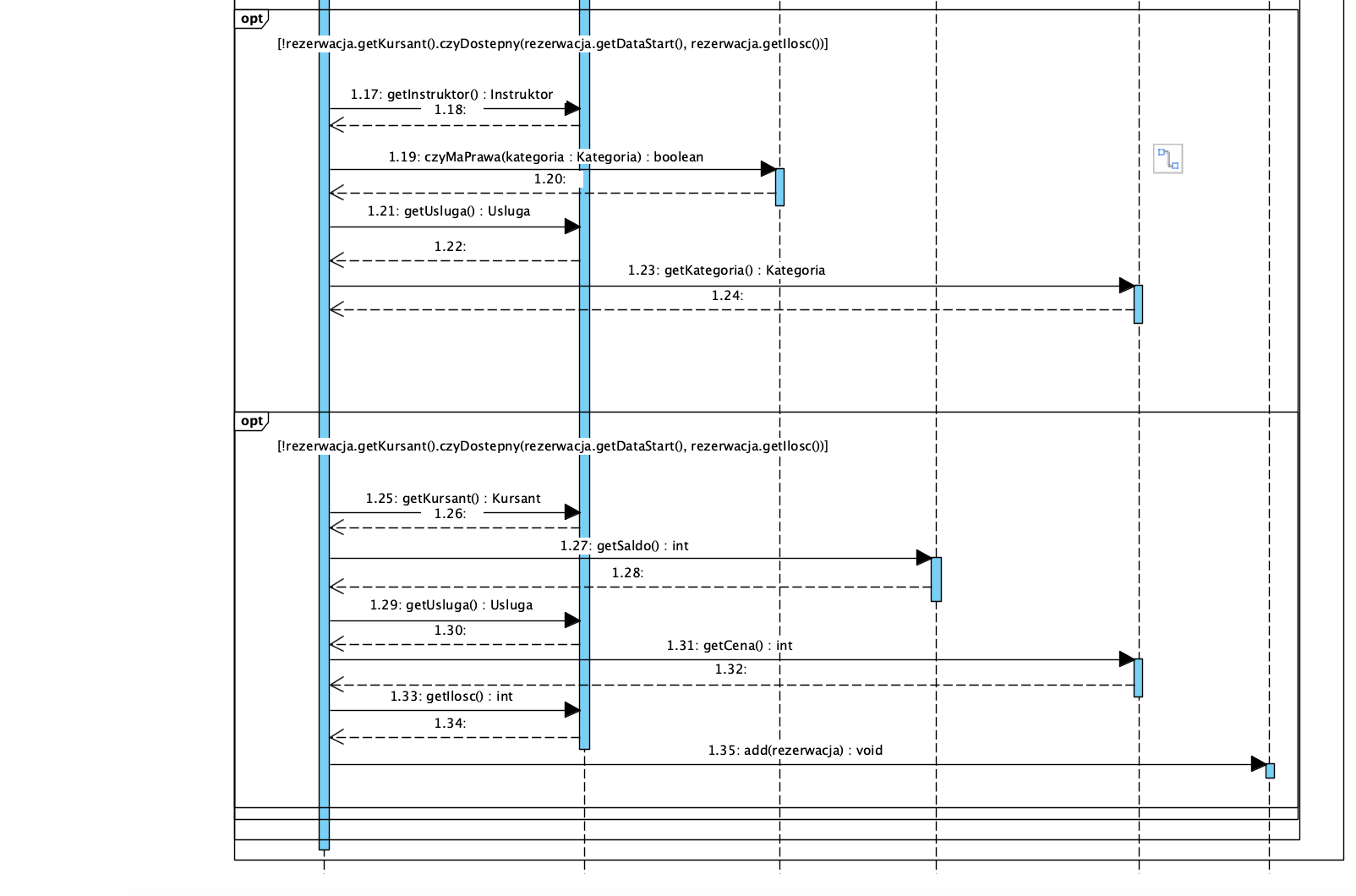
Opis wygenerowany automatycznie

Obraz zawierający mapa, tekst

Opis wygenerowany automatycznieObraz zawierający mapa, tekst

Opis wygenerowany automatycznie

**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);  
}

Obraz zawierający zrzut ekranu

Opis wygenerowany automatycznie2.6. Dodawanie rezerwacji

Obraz zawierający tekst, mapa

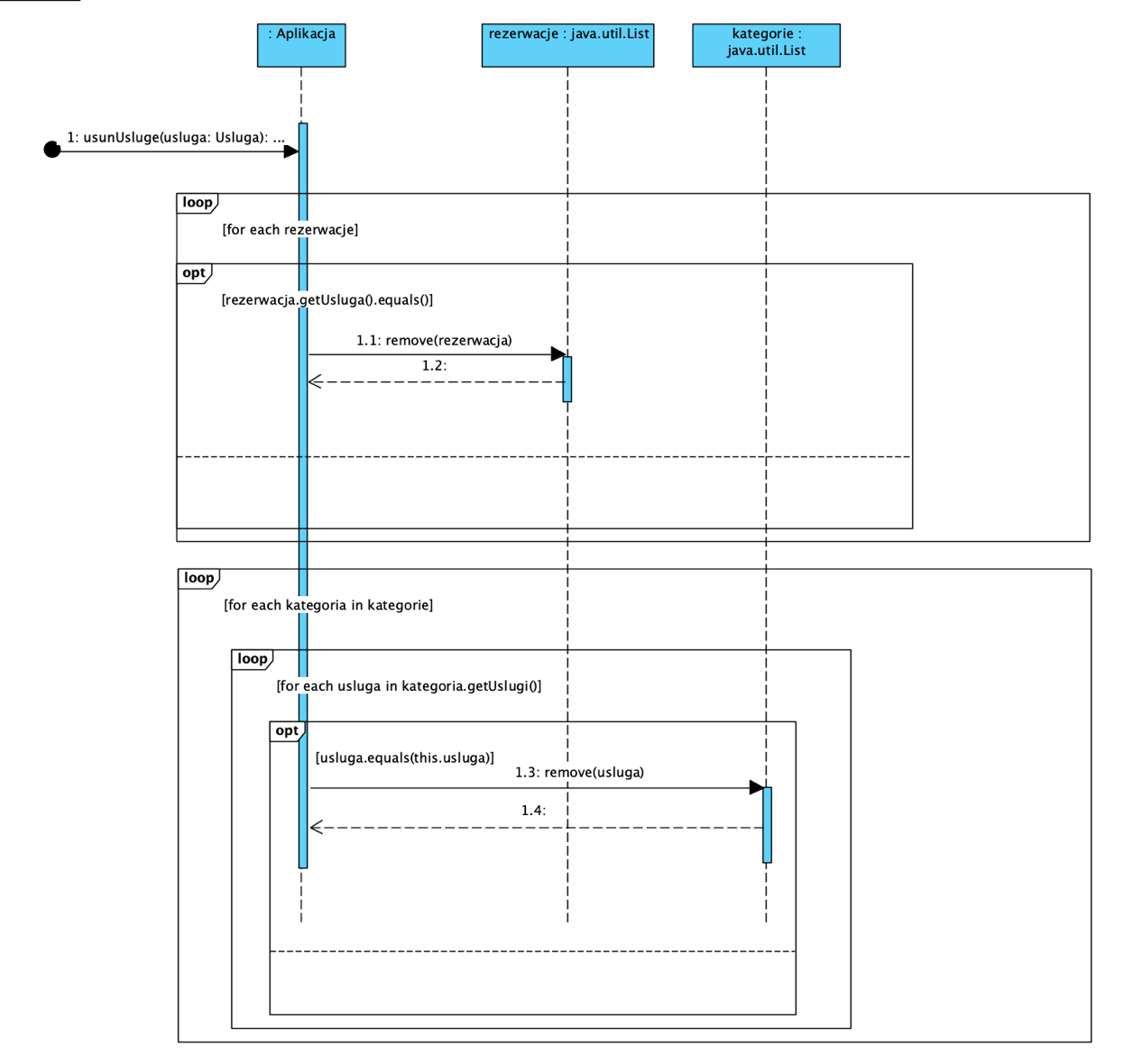
Opis wygenerowany automatycznie

**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);  
 }  
}

2.7. Usuwanie usługi

Obraz zawierający mapa, tekst

Opis wygenerowany automatycznie



**public void** usunUsluge(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);  
}

2.8. Dodawanie usługi

Obraz zawierający tekst, mapa

Opis wygenerowany automatycznie

Obraz zawierający zrzut ekranu

Opis wygenerowany automatycznie

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

2.9. Przypisywanie kategorii instruktorowi

Obraz zawierający tekst, mapa

Opis wygenerowany automatycznie

Obraz zawierający zrzut ekranu

Opis wygenerowany automatycznie

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

2.10. Dodawanie nowej kategorii

Obraz zawierający tekst, mapa

Opis wygenerowany automatycznie

Obraz zawierający zrzut ekranu

Opis wygenerowany automatycznie

**public void** dodajKategorie(Kategoria kategoria) {  
 **boolean** czyDuplikat = **false**;  
 **for** (Kategoria k : **kategorie**)  
 **if** (k.equals(kategoria))  
 czyDuplikat = **true**;  
 **if** (!czyDuplikat)  
 **kategorie**.add(kategoria);  
}

2.11. Usuwanie kategorii

Obraz zawierający tekst, mapa

Opis wygenerowany automatycznie

Obraz zawierający zrzut ekranu, mapa

Opis wygenerowany automatycznie

**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);  
}

2.12. Karta pracy instruktora

Obraz zawierający mapa, tekst

Opis wygenerowany automatycznie

Obraz zawierający zrzut ekranu

Opis wygenerowany automatycznie

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

Obraz zawierający mapa

Opis wygenerowany automatycznieObraz zawierający shoji

Opis wygenerowany automatycznie2.13. Sprawdzanie dostępności kursanta

Obraz zawierający tekst, mapa

Opis wygenerowany automatycznie

**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

Obraz zawierający tekst, mapa

Opis wygenerowany automatycznie

Obraz zawierający zrzut ekranu

Opis wygenerowany automatycznie

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