Errefaktorizazio Laborategia

Egileak: Iker Galarraga, Ibai Oñatibia eta Jon Olea

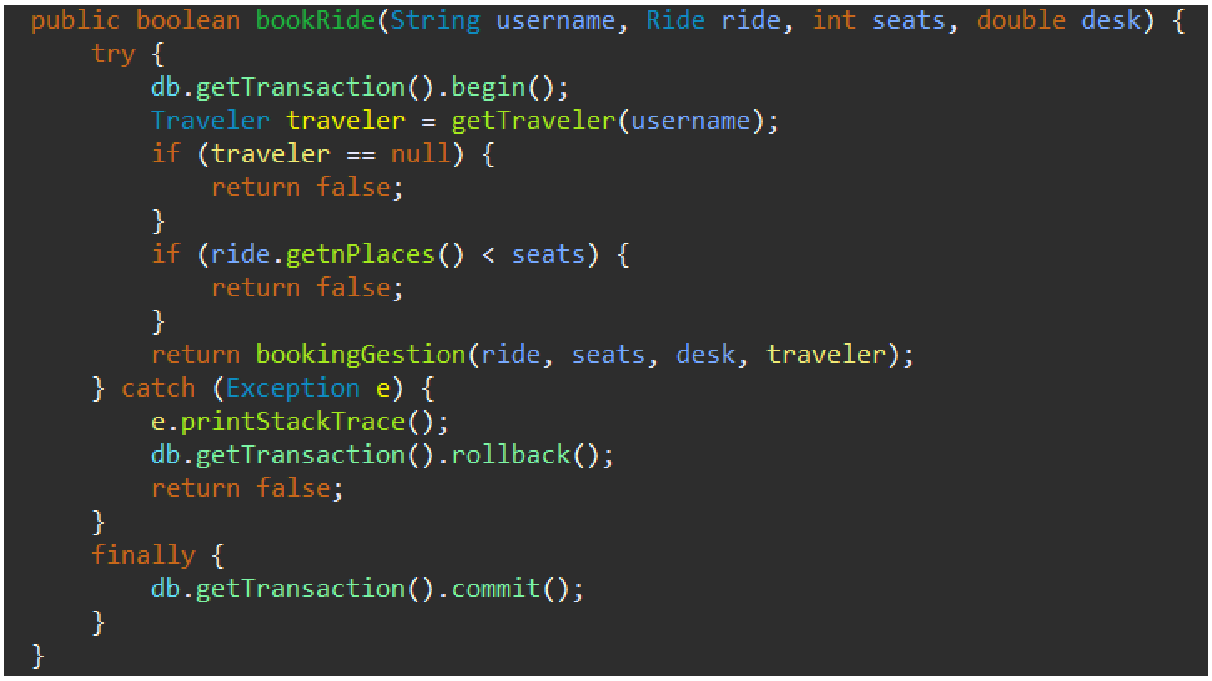
"Write short units of code"

bookRide (Jon Olea)

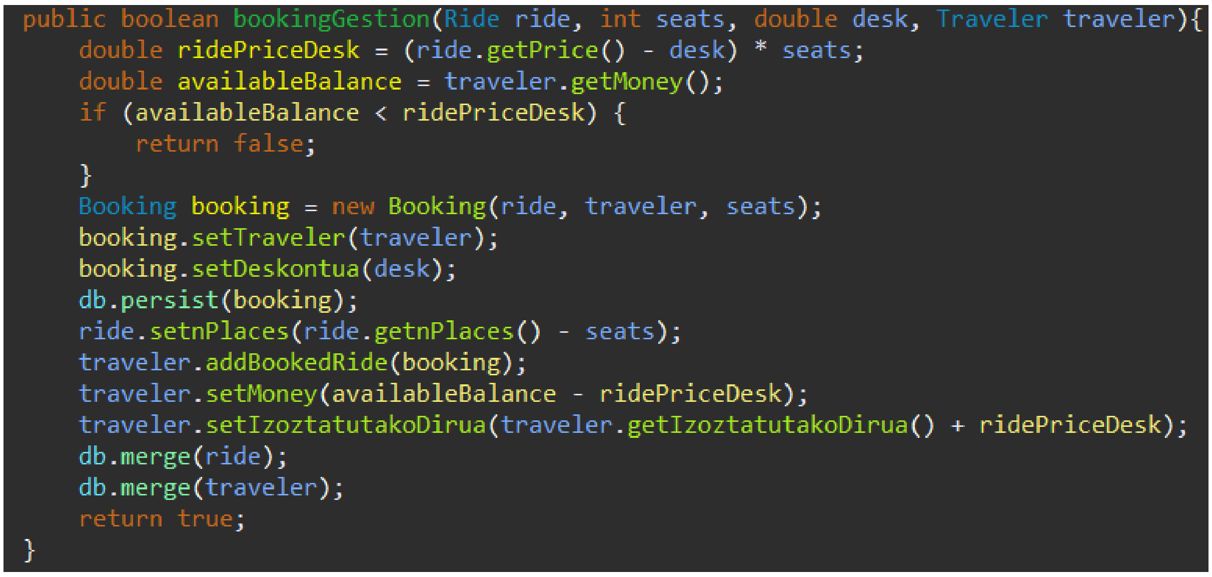
Hasiera Kodea:

```java
public boolean bookRide(String username, Ride ride, int seats, double desk) {
    try {
        db.getTransaction().begin();

        Traveler traveler = getTraveler(username);
        if (traveler == null) {
            return false;
        }

        if (ride.getnPlaces() < seats) {
            return false;
        }

        double ridePriceDesk = (ride.getPrice() - desk) * seats;
        double availableBalance = traveler.getMoney();
        if (availableBalance < ridePriceDesk) {
            return false;
        }

        Booking booking = new Booking(ride, traveler, seats);
        booking.setTraveler(traveler);
        booking.setDeskontua(desk);
        db.persist(booking);

        ride.setnPlaces(ride.getnPlaces() - seats);
        traveler.addBookedRide(booking);
        traveler.setMoney(availableBalance - ridePriceDesk);
        traveler.setIzoztatutakoDirua(traveler.getIzoztatutakoDirua() + ridePriceDesk);
        db.merge(ride);
        db.merge(traveler);
        return true;
    } catch (Exception e) {
        e.printStackTrace();
        db.getTransaction().rollback();
        return false;
    }
    finally {
        db.getTransaction().commit();
    }
}
```

Errefaktorizatutako kodea:



```
public boolean bookRide(String username, Ride ride, int seats, double desk) {
    try {
        db.getTransaction().begin();
        Traveler traveler = getTraveler(username);
        if (traveler == null) {
            return false;
        }
        if (ride.getnPlaces() < seats) {
            return false;
        }
        return bookingGestion(ride, seats, desk, traveler);
    } catch (Exception e) {
        e.printStackTrace();
        db.getTransaction().rollback();
        return false;
    }
    finally {
        db.getTransaction().commit();
    }
}
```



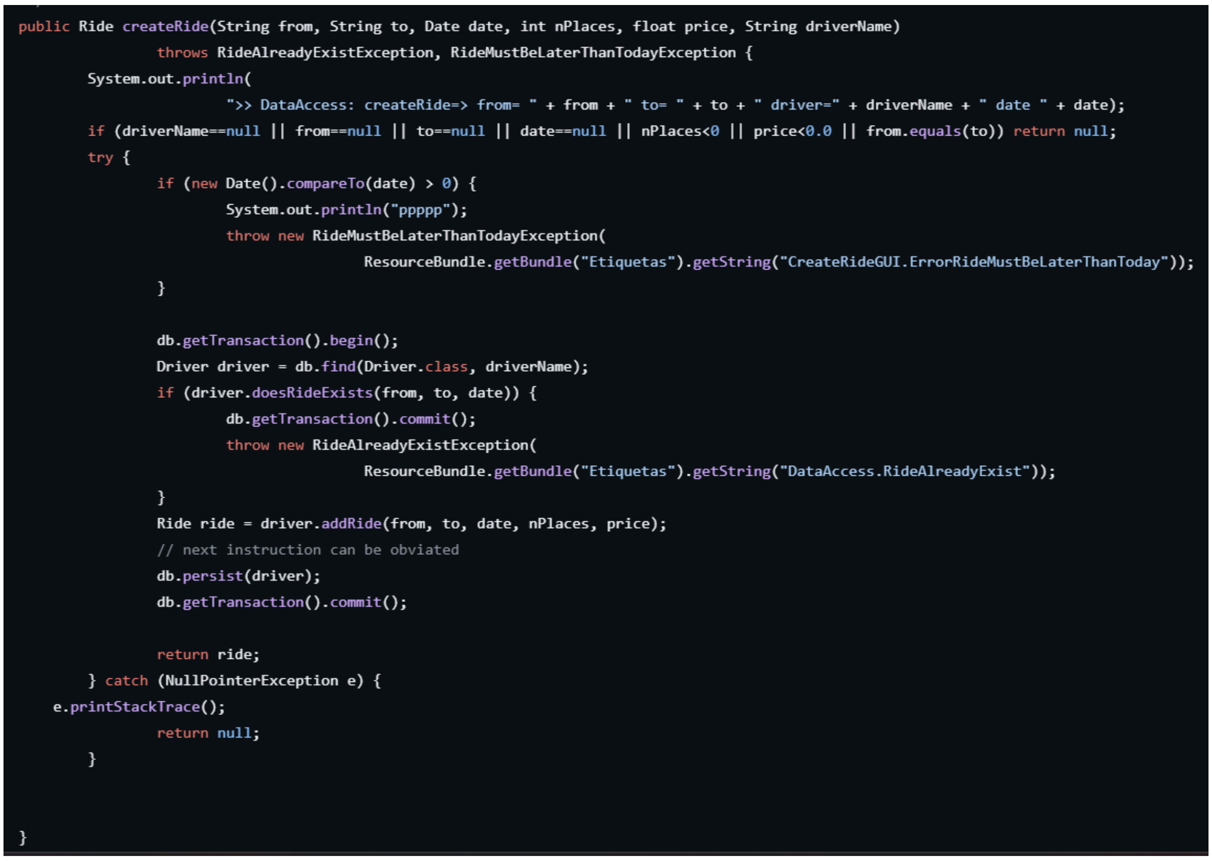
```
public boolean bookingGestion(Ride ride, int seats, double desk, Traveler traveler){
    double ridePriceDesk = (ride.getPrice() - desk) * seats;
    double availableBalance = traveler.getMoney();
    if (availableBalance < ridePriceDesk) {
        return false;
    }
    Booking booking = new Booking(ride, traveler, seats);
    booking.setTraveler(traveler);
    booking.setDeskontua(desk);
    db.persist(booking);
    ride.setnPlaces(ride.getnPlaces() - seats);
    traveler.addBookedRide(booking);
    traveler.setMoney(availableBalance - ridePriceDesk);
    traveler.setIzoztatutakoDirua(traveler.getIzoztatutakoDirua() + ridePriceDesk);
    db.merge(ride);
    db.merge(traveler);
    return true;
}
```

Deskribapena:

bookRide metodoak 15 lerro baina gehiago zituenez, bi metodotan banatu dut; bookRide metodoa (nagusia), eta bookingGestion, erreserba egitean dirua gestionatzeko metodoa. Horrela, metodo bakoitzak ez ditu 15 lerro baina gehiago.

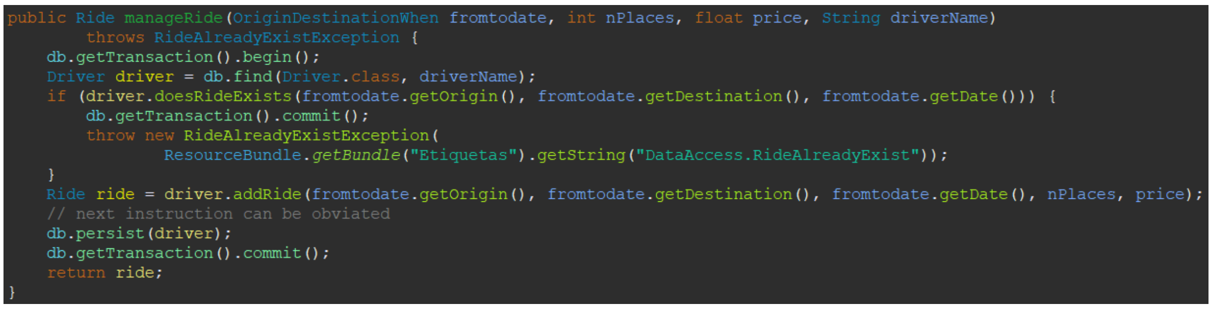
createRide(Iker)

Hasiera Kodea:

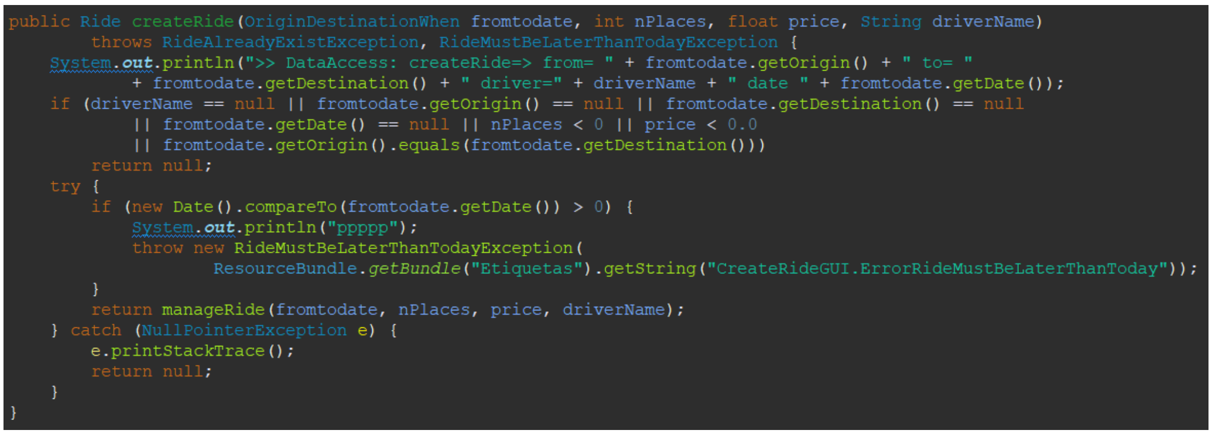


```java
public Ride createRide(String from, String to, Date date, int nPlaces, float price, String driverName)
                throws RideAlreadyExistException, RideMustBeLaterThanTodayException {
        System.out.println(
                        ">> DataAccess: createRide=> from= " + from + " to= " + to + " driver=" + driverName + " date " + date);
        if (driverName==null || from==null || to==null || date==null || nPlaces<0 || price<0.0 || from.equals(to)) return null;
        try {
                if (new Date().compareTo(date) > 0) {
                        System.out.println("ppppp");
                        throw new RideMustBeLaterThanTodayException(
                                        ResourceBundle.getBundle("Etiquetas").getString("CreateRideGUI.ErrorRideMustBeLaterThanToday"));
                }

                db.getTransaction().begin();
                Driver driver = db.find(Driver.class, driverName);
                if (driver.doesRideExists(from, to, date)) {
                        db.getTransaction().commit();
                        throw new RideAlreadyExistException(
                                        ResourceBundle.getBundle("Etiquetas").getString("DataAccess.RideAlreadyExist"));
                }
                Ride ride = driver.addRide(from, to, date, nPlaces, price);
                // next instruction can be obviated
                db.persist(driver);
                db.getTransaction().commit();

                return ride;
        } catch (NullPointerException e) {
    e.printStackTrace();
                return null;
        }


}
```

Errefaktorizatutako kodea:



```java
public Ride manageRide(OriginDestinationWhen fromtodate, int nPlaces, float price, String driverName)
        throws RideAlreadyExistException {
    db.getTransaction().begin();
    Driver driver = db.find(Driver.class, driverName);
    if (driver.doesRideExists(fromtodate.getOrigin(), fromtodate.getDestination(), fromtodate.getDate())) {
        db.getTransaction().commit();
        throw new RideAlreadyExistException(
                ResourceBundle.getBundle("Etiquetas").getString("DataAccess.RideAlreadyExist"));
    }
    Ride ride = driver.addRide(fromtodate.getOrigin(), fromtodate.getDestination(), fromtodate.getDate(), nPlaces, price);
    // next instruction can be obviated
    db.persist(driver);
    db.getTransaction().commit();
    return ride;
}
```



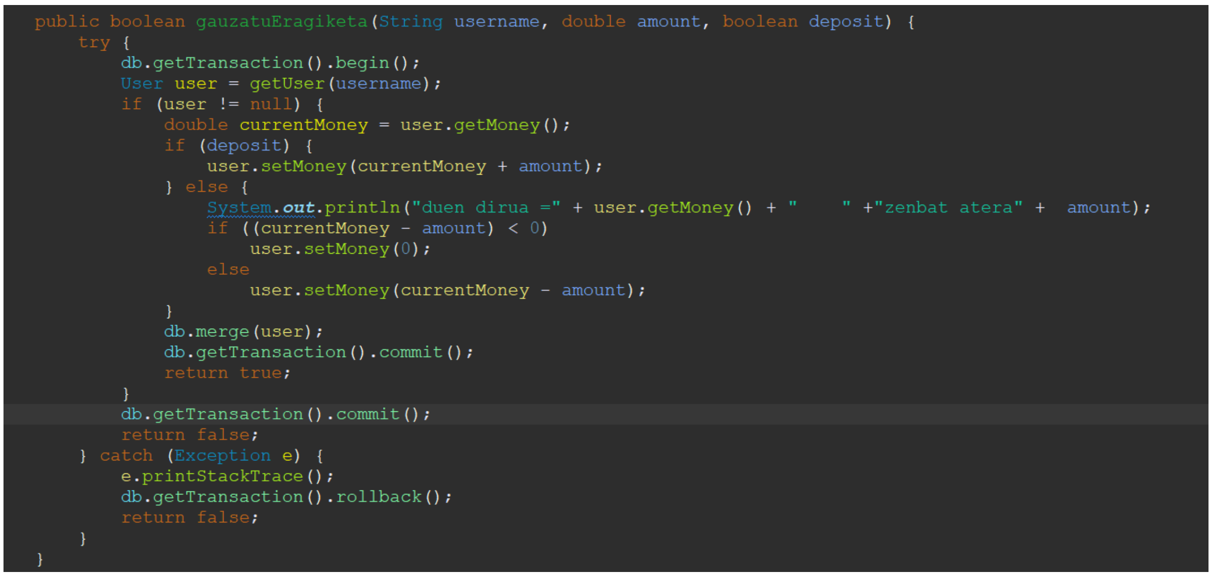
```java
public Ride createRide(OriginDestinationWhen fromtodate, int nPlaces, float price, String driverName)
        throws RideAlreadyExistException, RideMustBeLaterThanTodayException {
    System.out.println(">> DataAccess: createRide=> from= " + fromtodate.getOrigin() + " to= "
            + fromtodate.getDestination() + " driver=" + driverName + " date " + fromtodate.getDate());
    if (driverName == null || fromtodate.getOrigin() == null || fromtodate.getDestination() == null
            || fromtodate.getDate() == null || nPlaces < 0 || price < 0.0
            || fromtodate.getOrigin().equals(fromtodate.getDestination()))
        return null;
    try {
        if (new Date().compareTo(fromtodate.getDate()) > 0) {
            System.out.println("ppppp");
            throw new RideMustBeLaterThanTodayException(
                    ResourceBundle.getBundle("Etiquetas").getString("CreateRideGUI.ErrorRideMustBeLaterThanToday"));
        }
        return manageRide(fromtodate, nPlaces, price, driverName);
    } catch (NullPointerException e) {
        e.printStackTrace();
        return null;
    }
}
```

Deskribapena:

createRide metodoak 15 kode lerro baina gehiago zituen, errefaktorizazioa gauzatzeko beste metodo bat inplementatu dut, manageRide izenekoa. Metodo berriari esker createRide metodoak orain ez dauzka 15 kode lerro baina gehiago.

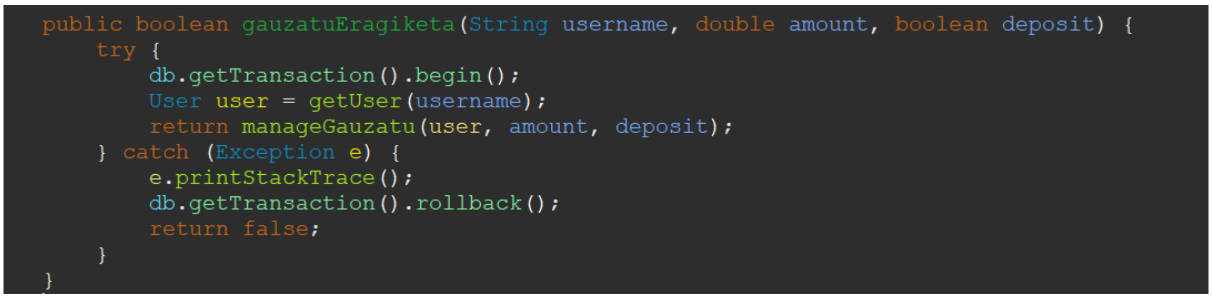
gauzatuEragiketa (Ibai Oñatibia)

Hasierako kodea:

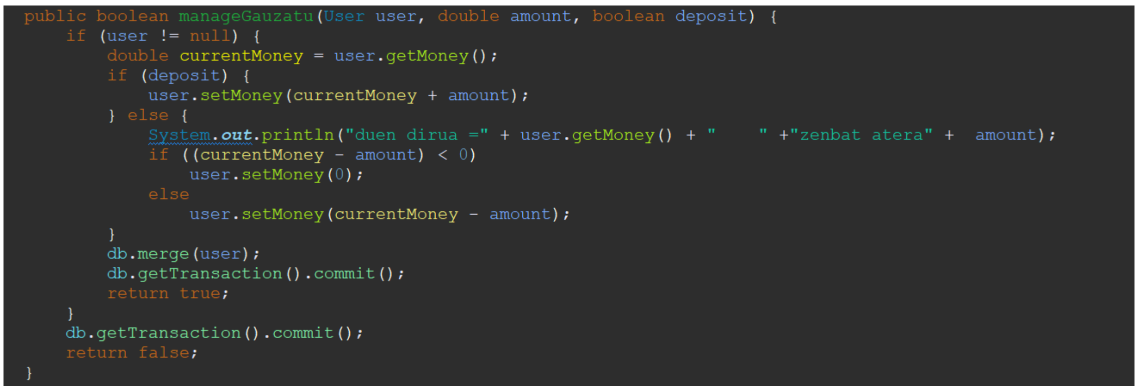


```
public boolean gauzatuEragiketa(String username, double amount, boolean deposit) {
    try {
        db.getTransaction().begin();
        User user = getUser(username);
        if (user != null) {
            double currentMoney = user.getMoney();
            if (deposit) {
                user.setMoney(currentMoney + amount);
            } else {
                System.out.println("duen dirua =" + user.getMoney() + "     " +"zenbat atera" +  amount);
                if ((currentMoney - amount) < 0)
                    user.setMoney(0);
                else
                    user.setMoney(currentMoney - amount);
            }
            db.merge(user);
            db.getTransaction().commit();
            return true;
        }
        db.getTransaction().commit();
        return false;
    } catch (Exception e) {
        e.printStackTrace();
        db.getTransaction().rollback();
        return false;
    }
}
```

Errefaktorizatutako kodea:



```
public boolean gauzatuEragiketa(String username, double amount, boolean deposit) {
    try {
        db.getTransaction().begin();
        User user = getUser(username);
        return manageGauzatu(user, amount, deposit);
    } catch (Exception e) {
        e.printStackTrace();
        db.getTransaction().rollback();
        return false;
    }
}
```



```
public boolean manageGauzatu(User user, double amount, boolean deposit) {
    if (user != null) {
        double currentMoney = user.getMoney();
        if (deposit) {
            user.setMoney(currentMoney + amount);
        } else {
            System.out.println("duen dirua =" + user.getMoney() + "     " +"zenbat atera" +  amount);
            if ((currentMoney - amount) < 0)
                user.setMoney(0);
            else
                user.setMoney(currentMoney - amount);
        }
        db.merge(user);
        db.getTransaction().commit();
        return true;
    }
    db.getTransaction().commit();
    return false;
}
```

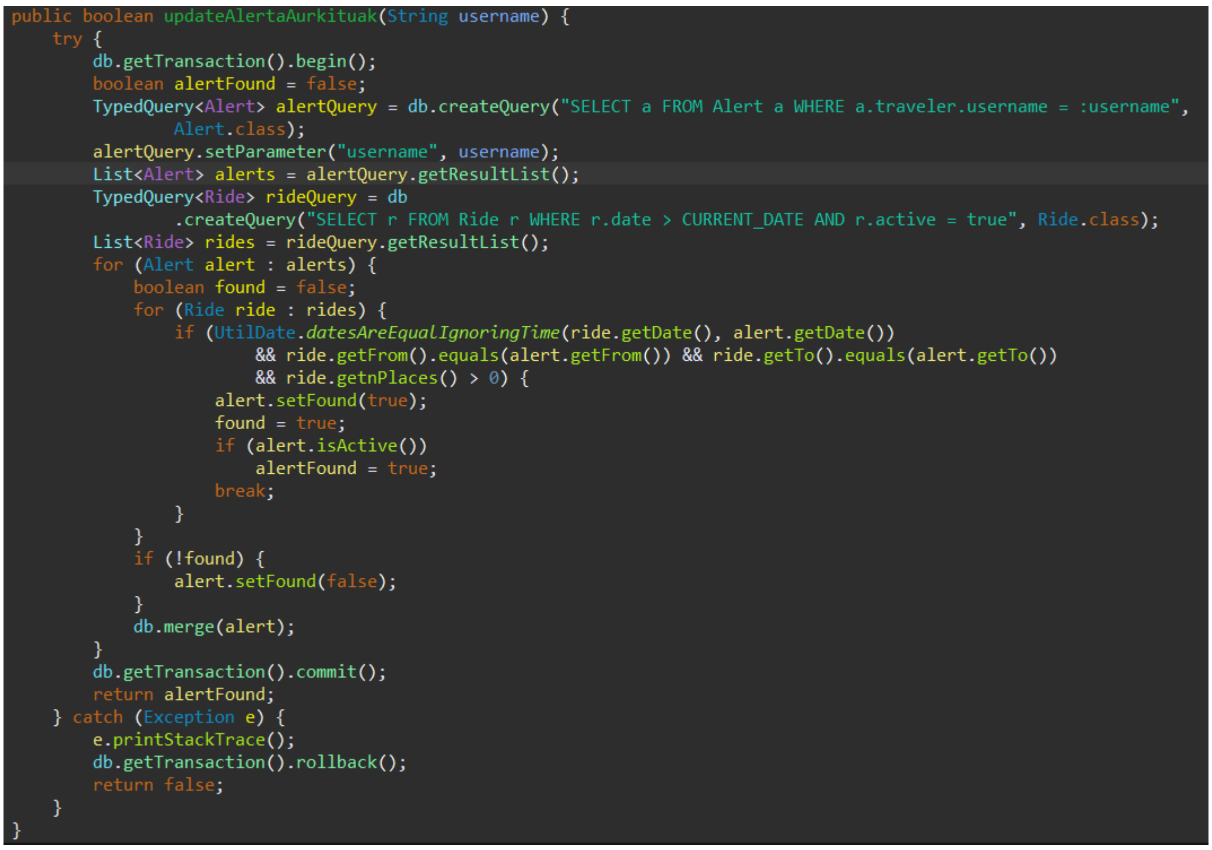
Deskribapena:

gauzatuEragiketa metodoa 15 lerrokoa baina handiagoa zenez , manageGauzatu metodoa erabiliz bi zatitan banatu dut, horrela bi metodoak ez dauzkate 15 lerro bainan gehiago.

"Write simple units of code"

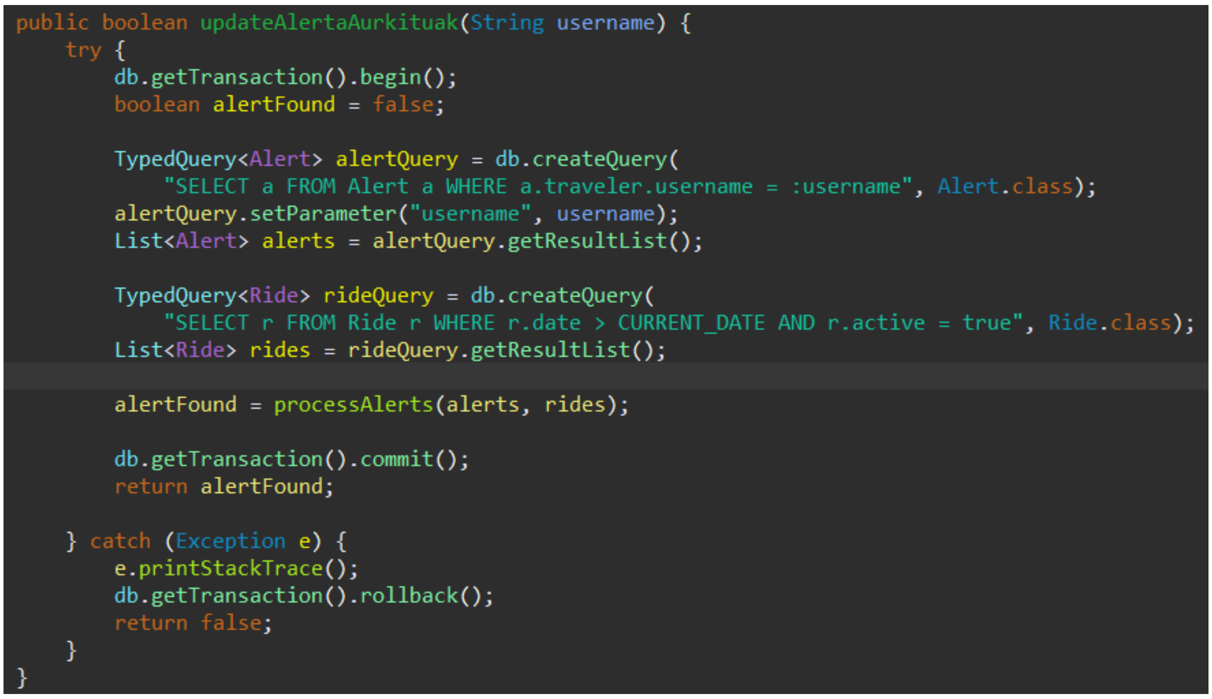
updateAlertaAurkituak (Jon Olea)

Hasiera Kodea:

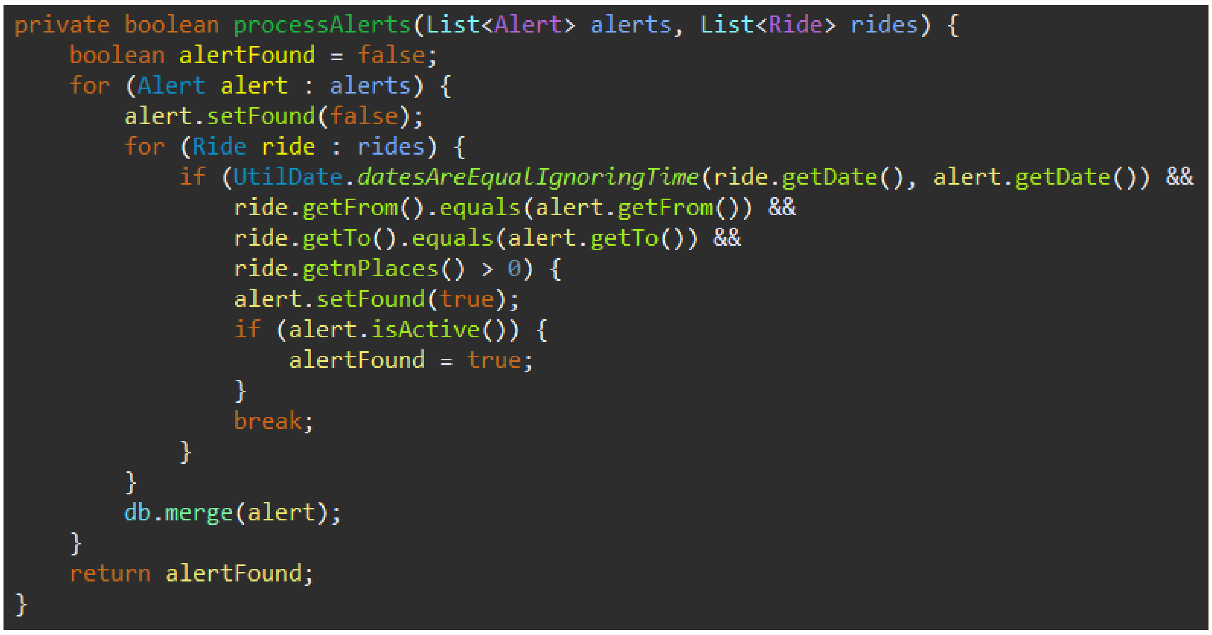


```java
public boolean updateAlertaAurkituak(String username) {
    try {
        db.getTransaction().begin();
        boolean alertFound = false;
        TypedQuery<Alert> alertQuery = db.createQuery("SELECT a FROM Alert a WHERE a.traveler.username = :username",
                Alert.class);
        alertQuery.setParameter("username", username);
        List<Alert> alerts = alertQuery.getResultList();
        TypedQuery<Ride> rideQuery = db
                .createQuery("SELECT r FROM Ride r WHERE r.date > CURRENT_DATE AND r.active = true", Ride.class);
        List<Ride> rides = rideQuery.getResultList();
        for (Alert alert : alerts) {
            boolean found = false;
            for (Ride ride : rides) {
                if (UtilDate.datesAreEqualIgnoringTime(ride.getDate(), alert.getDate())
                        && ride.getFrom().equals(alert.getFrom()) && ride.getTo().equals(alert.getTo())
                        && ride.getnPlaces() > 0) {
                    alert.setFound(true);
                    found = true;
                    if (alert.isActive())
                        alertFound = true;
                    break;
                }
            }
            if (!found) {
                alert.setFound(false);
            }
            db.merge(alert);
        }
        db.getTransaction().commit();
        return alertFound;
    } catch (Exception e) {
        e.printStackTrace();
        db.getTransaction().rollback();
        return false;
    }
}
```

Errefaktorizatutako kodea:



```java
public boolean updateAlertaAurkituak(String username) {
    try {
        db.getTransaction().begin();
        boolean alertFound = false;

        TypedQuery<Alert> alertQuery = db.createQuery(
            "SELECT a FROM Alert a WHERE a.traveler.username = :username", Alert.class);
        alertQuery.setParameter("username", username);
        List<Alert> alerts = alertQuery.getResultList();

        TypedQuery<Ride> rideQuery = db.createQuery(
            "SELECT r FROM Ride r WHERE r.date > CURRENT_DATE AND r.active = true", Ride.class);
        List<Ride> rides = rideQuery.getResultList();

        alertFound = processAlerts(alerts, rides);

        db.getTransaction().commit();
        return alertFound;

    } catch (Exception e) {
        e.printStackTrace();
        db.getTransaction().rollback();
        return false;
    }
}
```



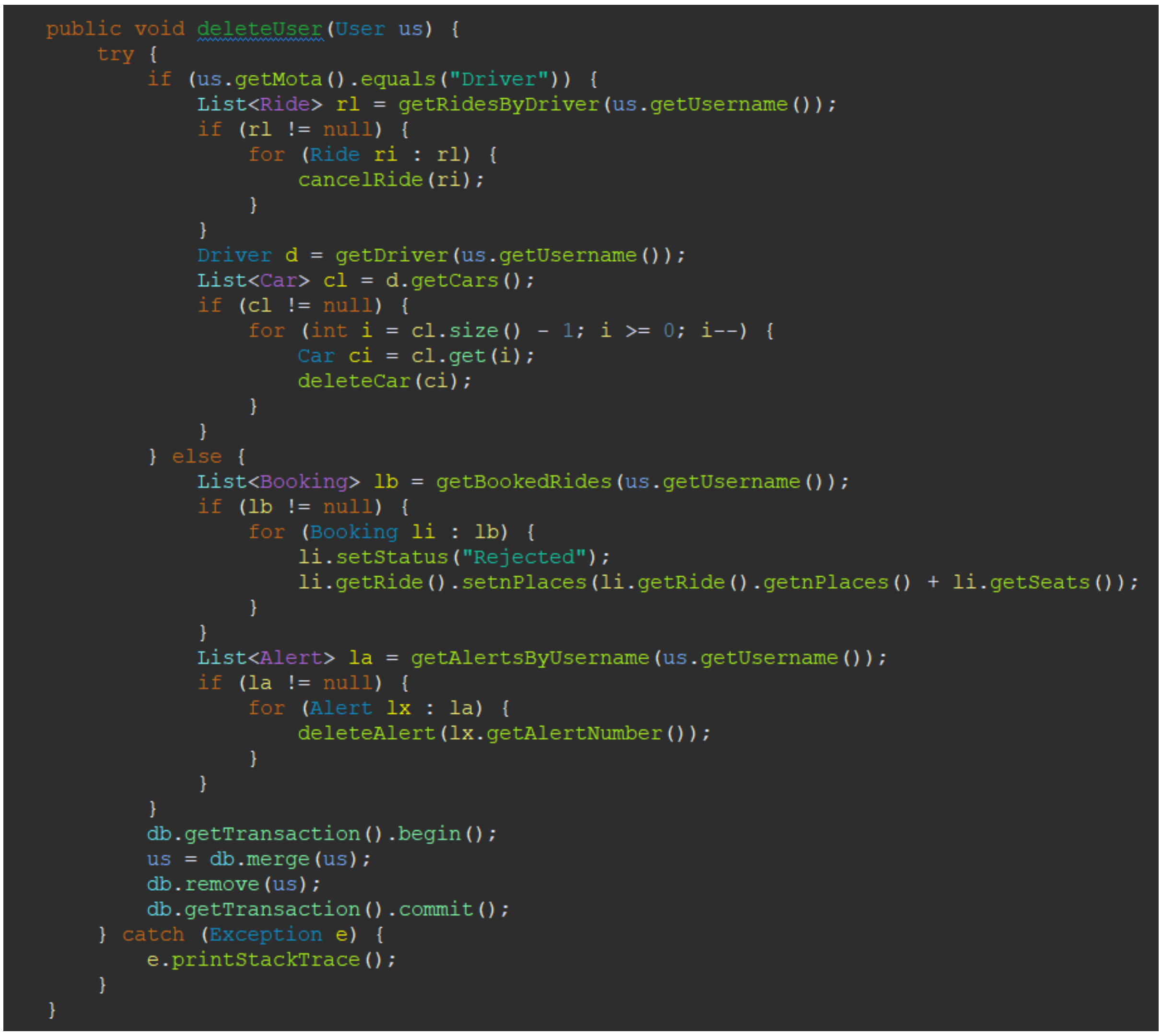
```java
private boolean processAlerts(List<Alert> alerts, List<Ride> rides) {
    boolean alertFound = false;
    for (Alert alert : alerts) {
        alert.setFound(false);
        for (Ride ride : rides) {
            if (UtilDate.datesAreEqualIgnoringTime(ride.getDate(), alert.getDate()) &&
                ride.getFrom().equals(alert.getFrom()) &&
                ride.getTo().equals(alert.getTo()) &&
                ride.getnPlaces() > 0) {
                alert.setFound(true);
                if (alert.isActive()) {
                    alertFound = true;
                }
                break;
            }
        }
        db.merge(alert);
    }
    return alertFound;
}
```

Deskribapena:

updateAlertaAurkituak metodoak hasiera 2 for eta 3 if zituen, beraz 4 “branch point” zituen. Hori konpontzeko, kodean aldaketak egin ditut, eta hasieran alarma guztiei setFound(false) jarri dizkiet, ondoren if bat kendu ahal izateko. Gainera, metodoa 2 metodotan banatu dut, alde batetik nagusia, query guztiak egiteko, eta bestetik alarmak kudeatzeko metodoa (bertan branch point guztiak).

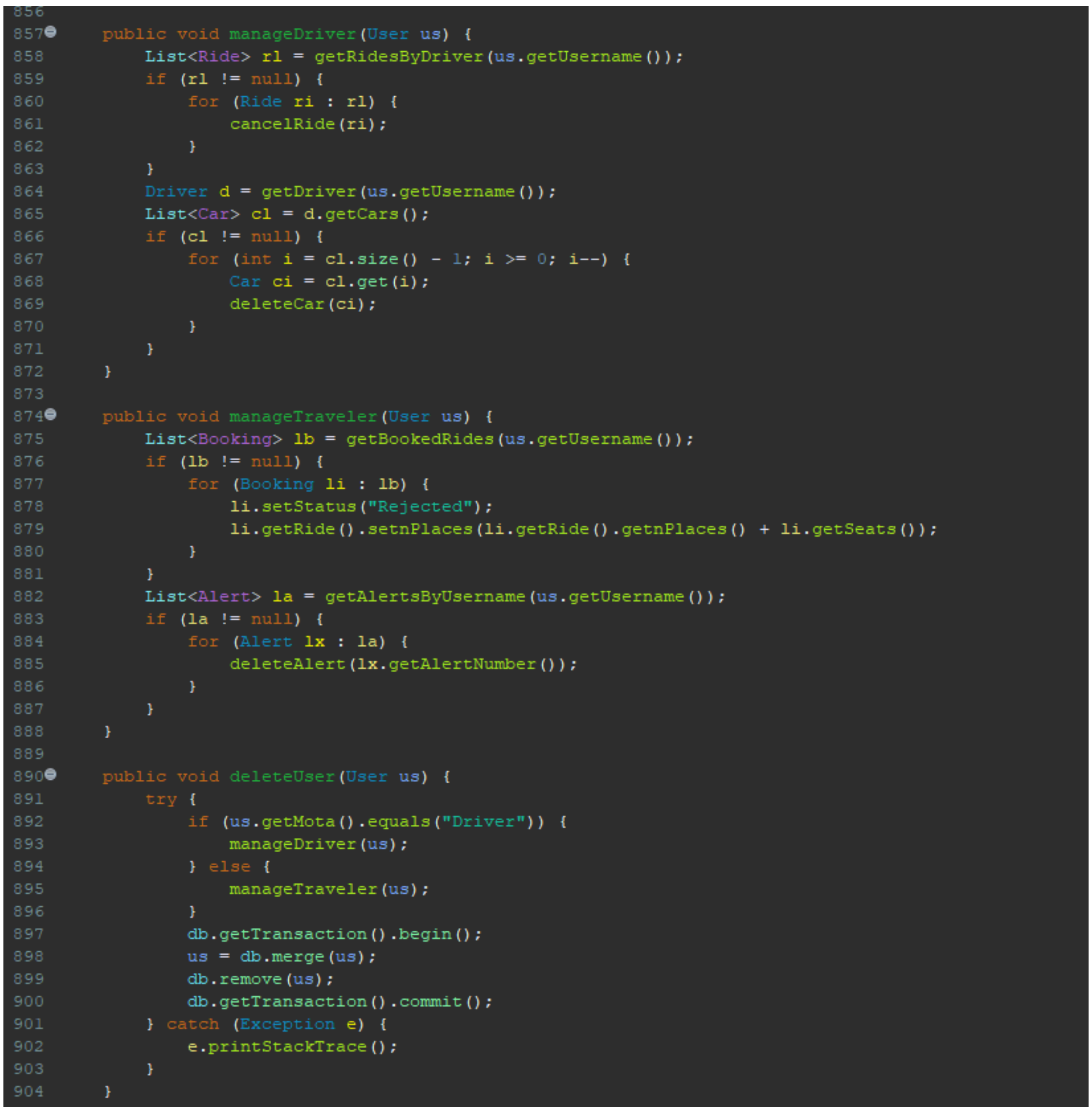
<u>deleteUser(Iker)</u>

Hasierako kodea:



```java
public void deleteUser(User us) {
    try {
        if (us.getMota().equals("Driver")) {
            List<Ride> rl = getRidesByDriver(us.getUsername());
            if (rl != null) {
                for (Ride ri : rl) {
                    cancelRide(ri);
                }
            }
            Driver d = getDriver(us.getUsername());
            List<Car> cl = d.getCars();
            if (cl != null) {
                for (int i = cl.size() - 1; i >= 0; i--) {
                    Car ci = cl.get(i);
                    deleteCar(ci);
                }
            }
        } else {
            List<Booking> lb = getBookedRides(us.getUsername());
            if (lb != null) {
                for (Booking li : lb) {
                    li.setStatus("Rejected");
                    li.getRide().setnPlaces(li.getRide().getnPlaces() + li.getSeats());
                }
            }
            List<Alert> la = getAlertsByUsername(us.getUsername());
            if (la != null) {
                for (Alert lx : la) {
                    deleteAlert(lx.getAlertNumber());
                }
            }
        }
        db.getTransaction().begin();
        us = db.merge(us);
        db.remove(us);
        db.getTransaction().commit();
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

Errefaktorizatutako kodea:

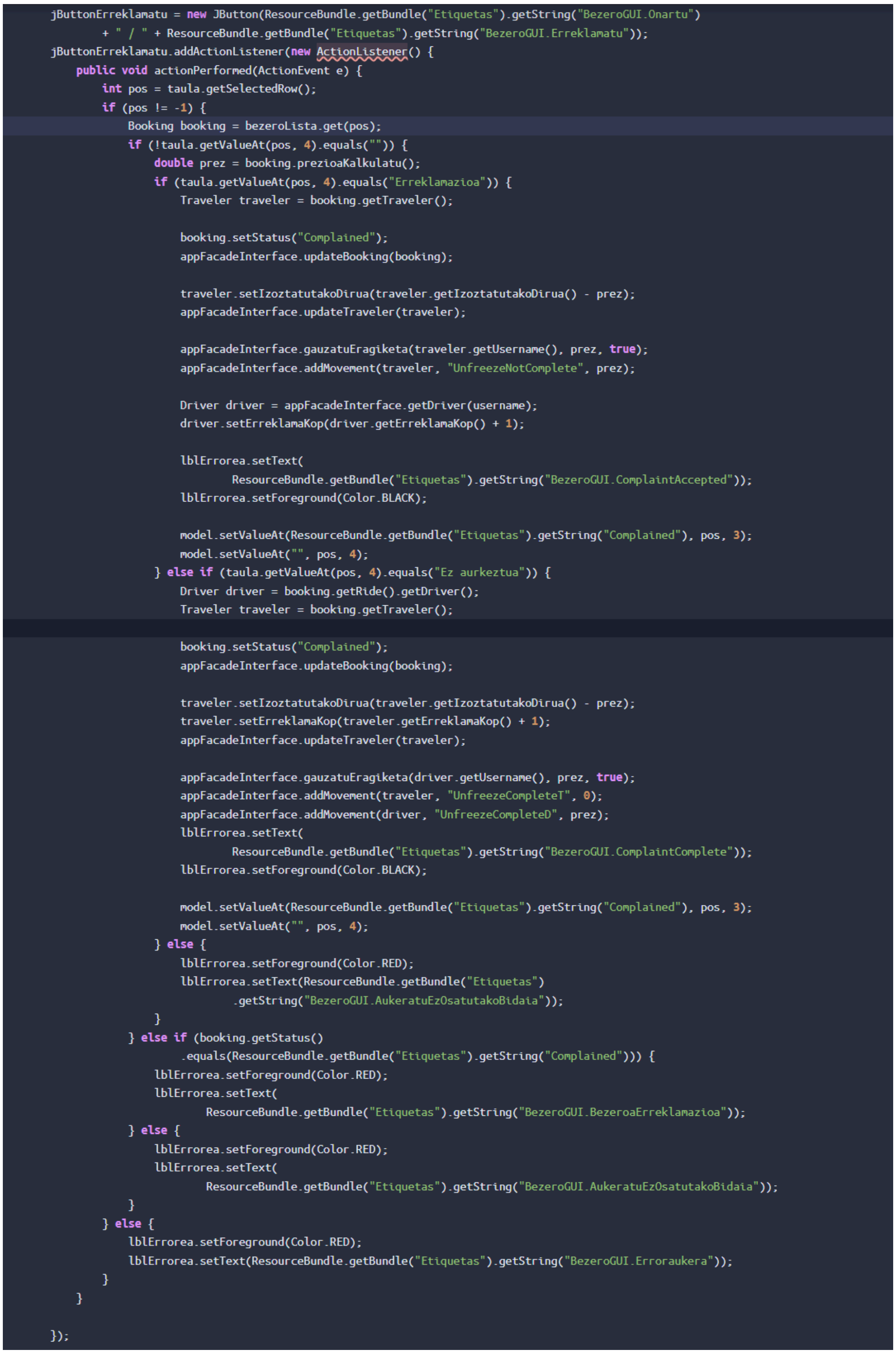


```java
    public void manageDriver(User us) {
        List<Ride> rl = getRidesByDriver(us.getUsername());
        if (rl != null) {
            for (Ride ri : rl) {
                cancelRide(ri);
            }
        }
        Driver d = getDriver(us.getUsername());
        List<Car> cl = d.getCars();
        if (cl != null) {
            for (int i = cl.size() - 1; i >= 0; i--) {
                Car ci = cl.get(i);
                deleteCar(ci);
            }
        }
    }

    public void manageTraveler(User us) {
        List<Booking> lb = getBookedRides(us.getUsername());
        if (lb != null) {
            for (Booking li : lb) {
                li.setStatus("Rejected");
                li.getRide().setnPlaces(li.getRide().getnPlaces() + li.getSeats());
            }
        }
        List<Alert> la = getAlertsByUsername(us.getUsername());
        if (la != null) {
            for (Alert lx : la) {
                deleteAlert(lx.getAlertNumber());
            }
        }
    }

    public void deleteUser(User us) {
        try {
            if (us.getMota().equals("Driver")) {
                manageDriver(us);
            } else {
                manageTraveler(us);
            }
            db.getTransaction().begin();
            us = db.merge(us);
            db.remove(us);
            db.getTransaction().commit();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

Deskribapena:

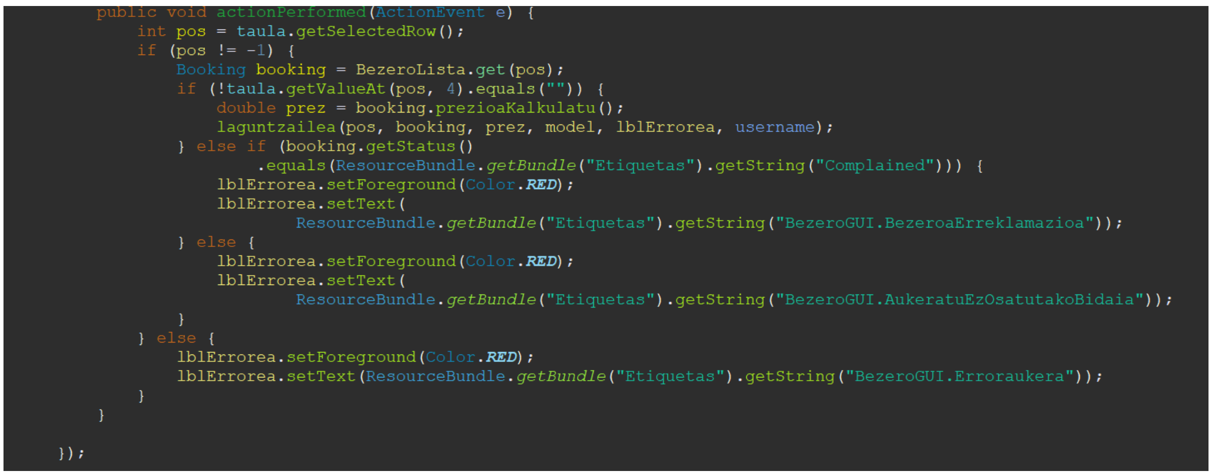
deleteUser metodoaren ‘branch’ kopurua 10ekoa zen (try + 2 if + for + if + for + if + for + if + for) = 10. ‘Branch’ kopurua 4ean mantentzeko bi metodo laguntzaile inplementatu ditut, hauek manageDriver eta manageTraveler izanik. Orain bi metodo berrien konplexutasun ziklomatikoa 5ekoa da, 4 ‘branch’ + 1 eta deleteUser metodoarena 3 (try + if) + 1.

BezeroGUI-ko ActionListenner-a (Ibai Oñatibia)

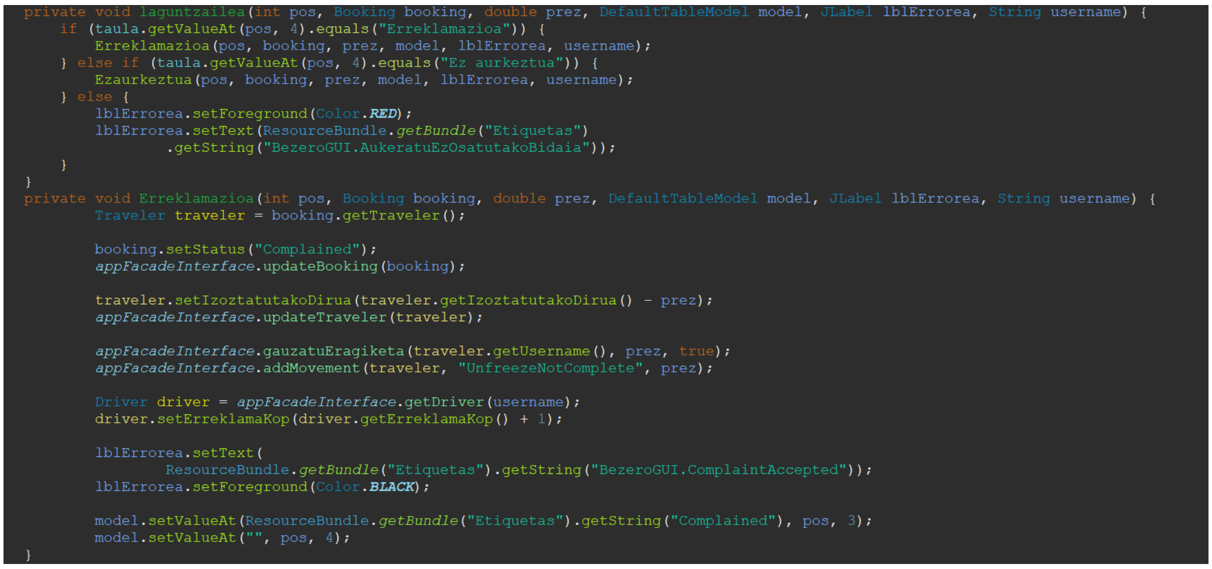


```java
jButtonErreklamatu = new JButton(ResourceBundle.getBundle("Etiquetas").getString("BezeroGUI.Onartu")
        + " / " + ResourceBundle.getBundle("Etiquetas").getString("BezeroGUI.Erreklamatu"));
jButtonErreklamatu.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        int pos = taula.getSelectedRow();
        if (pos != -1) {
            Booking booking = bezeroLista.get(pos);
            if (!taula.getValueAt(pos, 4).equals("")) {
                double prez = booking.prezioaKalkulatu();
                if (taula.getValueAt(pos, 4).equals("Erreklamazioa")) {
                    Traveler traveler = booking.getTraveler();

                    booking.setStatus("Complained");
                    appFacadeInterface.updateBooking(booking);

                    traveler.setIzoztatutakoDirua(traveler.getIzoztatutakoDirua() - prez);
                    appFacadeInterface.updateTraveler(traveler);

                    appFacadeInterface.gauzatuEragiketa(traveler.getUsername(), prez, true);
                    appFacadeInterface.addMovement(traveler, "UnfreezeNotComplete", prez);

                    Driver driver = appFacadeInterface.getDriver(username);
                    driver.setErreklamaKop(driver.getErreklamaKop() + 1);

                    lblErrorea.setText(
                            ResourceBundle.getBundle("Etiquetas").getString("BezeroGUI.ComplaintAccepted"));
                    lblErrorea.setForeground(Color.BLACK);

                    model.setValueAt(ResourceBundle.getBundle("Etiquetas").getString("Complained"), pos, 3);
                    model.setValueAt("", pos, 4);
                } else if (taula.getValueAt(pos, 4).equals("Ez aurkeztua")) {
                    Driver driver = booking.getRide().getDriver();
                    Traveler traveler = booking.getTraveler();

                    booking.setStatus("Complained");
                    appFacadeInterface.updateBooking(booking);

                    traveler.setIzoztatutakoDirua(traveler.getIzoztatutakoDirua() - prez);
                    traveler.setErreklamaKop(traveler.getErreklamaKop() + 1);
                    appFacadeInterface.updateTraveler(traveler);

                    appFacadeInterface.gauzatuEragiketa(driver.getUsername(), prez, true);
                    appFacadeInterface.addMovement(traveler, "UnfreezeCompleteT", 0);
                    appFacadeInterface.addMovement(driver, "UnfreezeCompleteD", prez);
                    lblErrorea.setText(
                            ResourceBundle.getBundle("Etiquetas").getString("BezeroGUI.ComplaintComplete"));
                    lblErrorea.setForeground(Color.BLACK);

                    model.setValueAt(ResourceBundle.getBundle("Etiquetas").getString("Complained"), pos, 3);
                    model.setValueAt("", pos, 4);
                } else {
                    lblErrorea.setForeground(Color.RED);
                    lblErrorea.setText(ResourceBundle.getBundle("Etiquetas")
                            .getString("BezeroGUI.AukeratuEzOsatutakoBidaia"));
                }
            } else if (booking.getStatus()
                    .equals(ResourceBundle.getBundle("Etiquetas").getString("Complained"))) {
                lblErrorea.setForeground(Color.RED);
                lblErrorea.setText(
                        ResourceBundle.getBundle("Etiquetas").getString("BezeroGUI.BezeroaErreklamazioa"));
            } else {
                lblErrorea.setForeground(Color.RED);
                lblErrorea.setText(
                        ResourceBundle.getBundle("Etiquetas").getString("BezeroGUI.AukeratuEzOsatutakoBidaia"));
            }
        } else {
            lblErrorea.setForeground(Color.RED);
            lblErrorea.setText(ResourceBundle.getBundle("Etiquetas").getString("BezeroGUI.Erroraukera"));
        }
    }

});
```

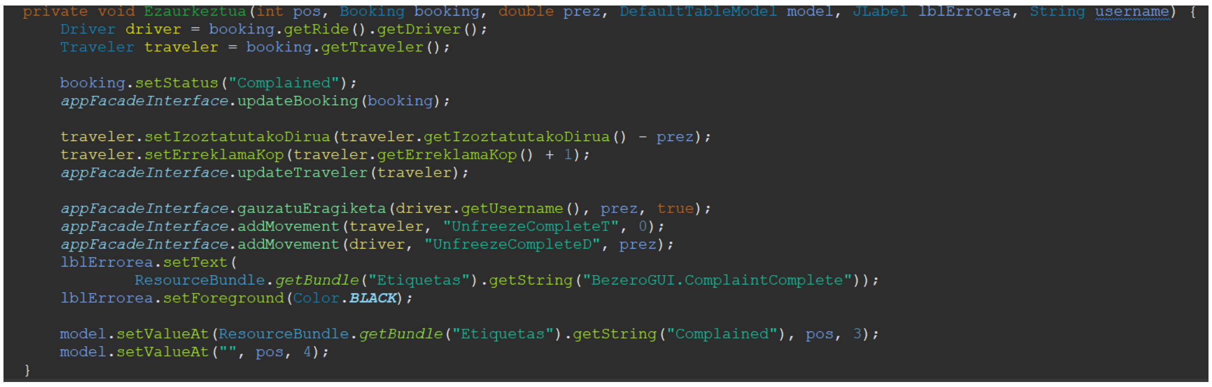
Errefaktorizatutako kodea:



```java
        public void actionPerformed(ActionEvent e) {
            int pos = taula.getSelectedRow();
            if (pos != -1) {
                Booking booking = BezeroLista.get(pos);
                if (!taula.getValueAt(pos, 4).equals("")) {
                    double prez = booking.prezioaKalkulatu();
                    laguntzailea(pos, booking, prez, model, lblErrorea, username);
                } else if (booking.getStatus()
                        .equals(ResourceBundle.getBundle("Etiquetas").getString("Complained"))) {
                    lblErrorea.setForeground(Color.RED);
                    lblErrorea.setText(
                            ResourceBundle.getBundle("Etiquetas").getString("BezeroGUI.BezeroaErreklamazioa"));
                } else {
                    lblErrorea.setForeground(Color.RED);
                    lblErrorea.setText(
                            ResourceBundle.getBundle("Etiquetas").getString("BezeroGUI.AukeratuEzOsatutakoBidaia"));
                }
            } else {
                lblErrorea.setForeground(Color.RED);
                lblErrorea.setText(ResourceBundle.getBundle("Etiquetas").getString("BezeroGUI.Erroraukera"));
            }
        }

    });
```



```java
private void laguntzailea(int pos, Booking booking, double prez, DefaultTableModel model, JLabel lblErrorea, String username) {
    if (taula.getValueAt(pos, 4).equals("Erreklamazioa")) {
        Erreklamazioa(pos, booking, prez, model, lblErrorea, username);
    } else if (taula.getValueAt(pos, 4).equals("Ez aurkeztua")) {
        Ezaurkeztua(pos, booking, prez, model, lblErrorea, username);
    } else {
        lblErrorea.setForeground(Color.RED);
        lblErrorea.setText(ResourceBundle.getBundle("Etiquetas")
                .getString("BezeroGUI.AukeratuEzOsatutakoBidaia"));
    }
}
private void Erreklamazioa(int pos, Booking booking, double prez, DefaultTableModel model, JLabel lblErrorea, String username) {
        Traveler traveler = booking.getTraveler();

        booking.setStatus("Complained");
        appFacadeInterface.updateBooking(booking);

        traveler.setIzoztatutakoDirua(traveler.getIzoztatutakoDirua() - prez);
        appFacadeInterface.updateTraveler(traveler);

        appFacadeInterface.gauzatuEragiketa(traveler.getUsername(), prez, true);
        appFacadeInterface.addMovement(traveler, "UnfreezeNotComplete", prez);

        Driver driver = appFacadeInterface.getDriver(username);
        driver.setErreklamaKop(driver.getErreklamaKop() + 1);

        lblErrorea.setText(
                ResourceBundle.getBundle("Etiquetas").getString("BezeroGUI.ComplaintAccepted"));
        lblErrorea.setForeground(Color.BLACK);

        model.setValueAt(ResourceBundle.getBundle("Etiquetas").getString("Complained"), pos, 3);
        model.setValueAt("", pos, 4);
}
```



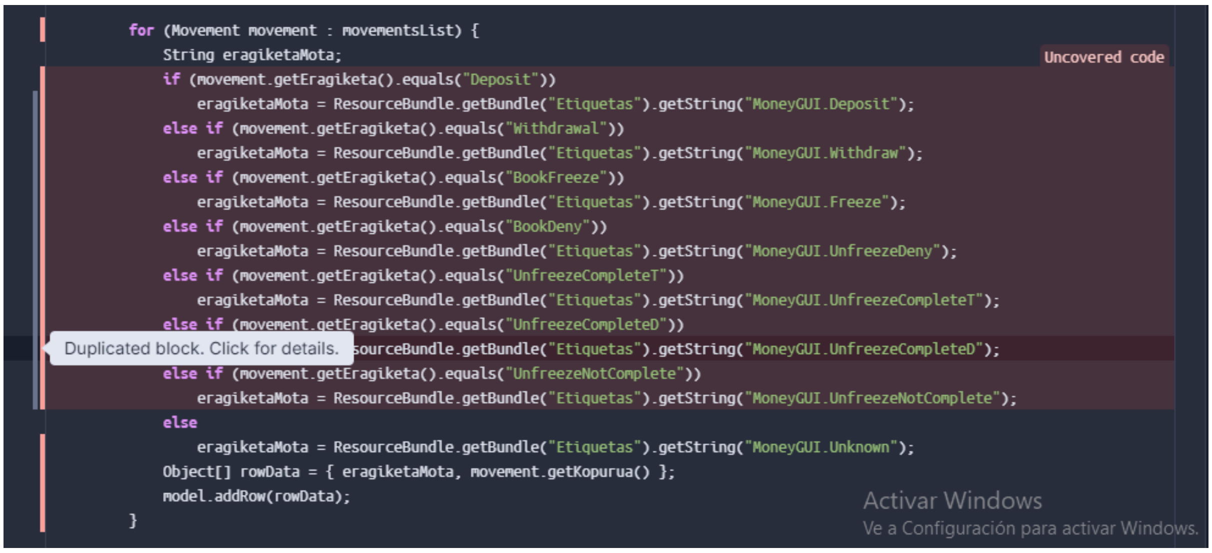
```java
private void Ezaurkeztua(int pos, Booking booking, double prez, DefaultTableModel model, JLabel lblErrorea, String username) {
    Driver driver = booking.getRide().getDriver();
    Traveler traveler = booking.getTraveler();

    booking.setStatus("Complained");
    appFacadeInterface.updateBooking(booking);

    traveler.setIzoztatutakoDirua(traveler.getIzoztatutakoDirua() - prez);
    traveler.setErreklamaKop(traveler.getErreklamaKop() + 1);
    appFacadeInterface.updateTraveler(traveler);

    appFacadeInterface.gauzatuEragiketa(driver.getUsername(), prez, true);
    appFacadeInterface.addMovement(traveler, "UnfreezeCompleteT", 0);
    appFacadeInterface.addMovement(driver, "UnfreezeCompleteD", prez);
    lblErrorea.setText(
            ResourceBundle.getBundle("Etiquetas").getString("BezeroGUI.ComplaintComplete"));
    lblErrorea.setForeground(Color.BLACK);

    model.setValueAt(ResourceBundle.getBundle("Etiquetas").getString("Complained"), pos, 3);
    model.setValueAt("", pos, 4);
}
```

Deskribapena:
 Metodo honek 8 branch point izanik metodo desberdinetan banatu dut eta “write short unit codes” errespetatzeko ere beste bi metodo sortu ditut Erreklamazio eta Ezaurkeztua izenekoak.

“Duplicated code”

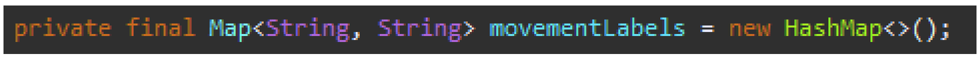
<u>MovementsGUI klaseko atal bat (Jon Olea)</u>

Hasierako kodea:

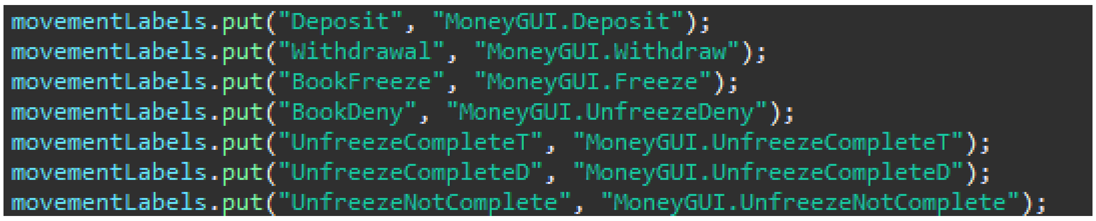

Uncovered code
Duplicated block. Click for details.
Activar Windows
Ve a Configuración para activar Windows.

```
        for (Movement movement : movementsList) {
            String eragiketaMota;
            if (movement.getEragiketa().equals("Deposit"))
                eragiketaMota = ResourceBundle.getBundle("Etiquetas").getString("MoneyGUI.Deposit");
            else if (movement.getEragiketa().equals("Withdrawal"))
                eragiketaMota = ResourceBundle.getBundle("Etiquetas").getString("MoneyGUI.Withdraw");
            else if (movement.getEragiketa().equals("BookFreeze"))
                eragiketaMota = ResourceBundle.getBundle("Etiquetas").getString("MoneyGUI.Freeze");
            else if (movement.getEragiketa().equals("BookDeny"))
                eragiketaMota = ResourceBundle.getBundle("Etiquetas").getString("MoneyGUI.UnfreezeDeny");
            else if (movement.getEragiketa().equals("UnfreezeCompleteT"))
                eragiketaMota = ResourceBundle.getBundle("Etiquetas").getString("MoneyGUI.UnfreezeCompleteT");
            else if (movement.getEragiketa().equals("UnfreezeCompleteD"))
                      sourceBundle.getBundle("Etiquetas").getString("MoneyGUI.UnfreezeCompleteD");
            else if (movement.getEragiketa().equals("UnfreezeNotComplete"))
                eragiketaMota = ResourceBundle.getBundle("Etiquetas").getString("MoneyGUI.UnfreezeNotComplete");
            else
                eragiketaMota = ResourceBundle.getBundle("Etiquetas").getString("MoneyGUI.Unknown");
            Object[] rowData = { eragiketaMota, movement.getKopurua() };
            model.addRow(rowData);
        }
```

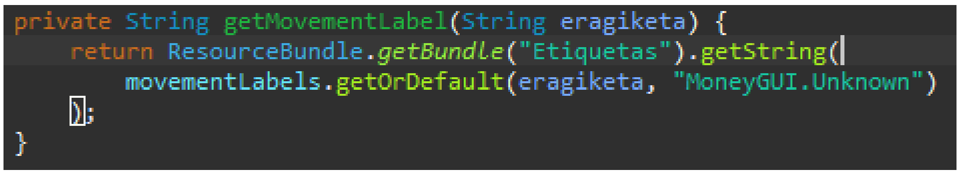
Errefaktorizatutako kodea:



```
private final Map<String, String> movementLabels = new HashMap<>();
```



```
movementLabels.put("Deposit", "MoneyGUI.Deposit");
movementLabels.put("Withdrawal", "MoneyGUI.Withdraw");
movementLabels.put("BookFreeze", "MoneyGUI.Freeze");
movementLabels.put("BookDeny", "MoneyGUI.UnfreezeDeny");
movementLabels.put("UnfreezeCompleteT", "MoneyGUI.UnfreezeCompleteT");
movementLabels.put("UnfreezeCompleteD", "MoneyGUI.UnfreezeCompleteD");
movementLabels.put("UnfreezeNotComplete", "MoneyGUI.UnfreezeNotComplete");
```



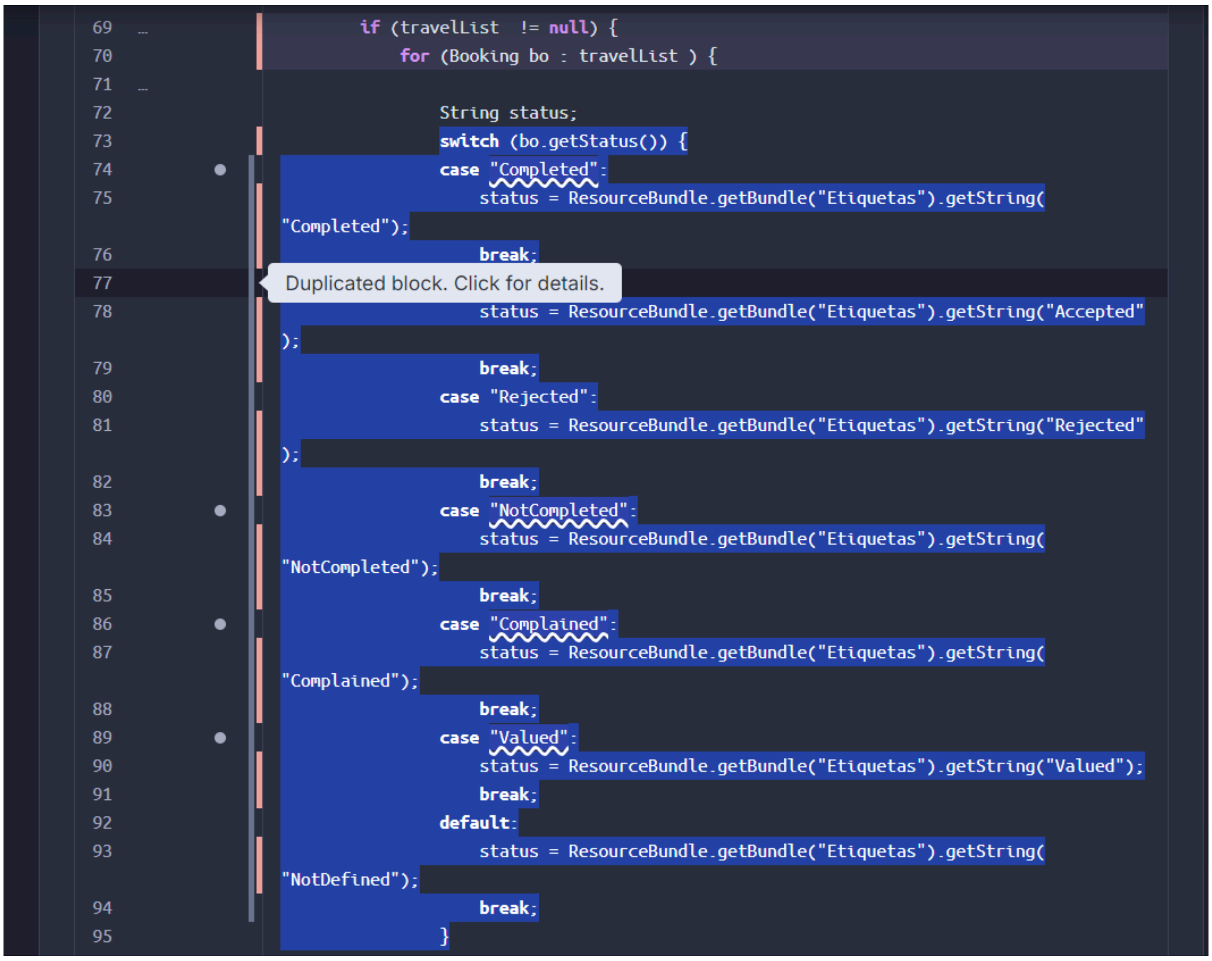
```
private String getMovementLabel(String eragiketa) {
    return ResourceBundle.getBundle("Etiquetas").getString(
        movementLabels.getOrDefault(eragiketa, "MoneyGUI.Unknown")
    );
}
```

Deskribapena:

Klase barruko atal batea, String baten balio baten arabera aldagai batean balio bat edo beste gordetzen zen if-else asko erabilita, horrek “duplicated code” adierazten zuen. Konpontzeko, hashmap bat sortu dut, String-aren arabera balio ezberdina itzultzeko.
if-else-ak ordez, metodo bat sortu dut, parametro bezala String-a pasaz, eta map-aren arabera balioa itzultzeko.

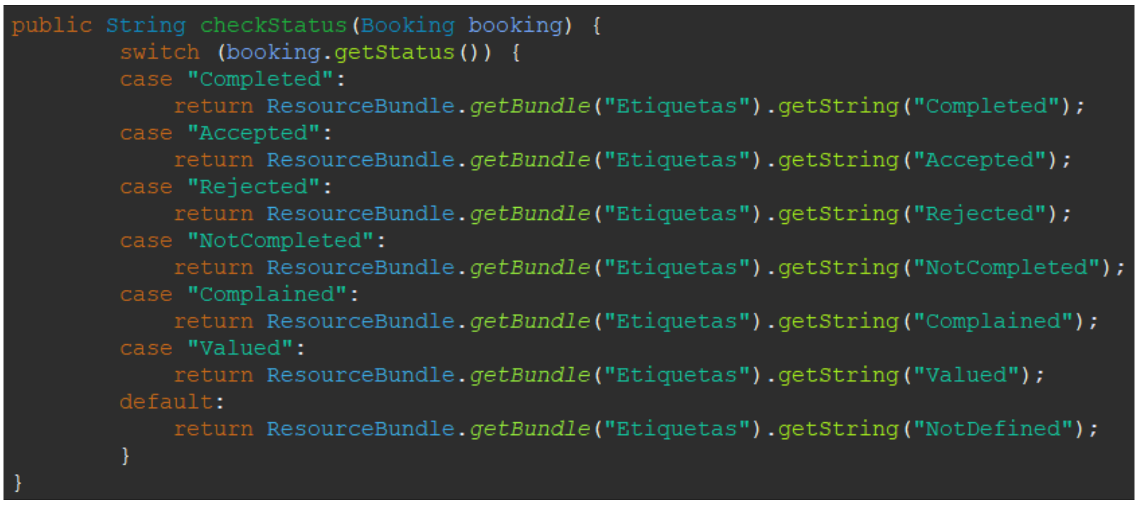
BezeroGUI/EgoeraGUI/ErreserbaOnartuGUI(Iker)

Hasierako kodea:



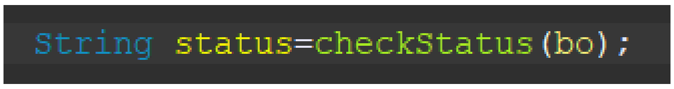
```
        if (travelList  != null) {
            for (Booking bo : travelList ) {

                String status;
                switch (bo.getStatus()) {
                case "Completed":
                    status = ResourceBundle.getBundle("Etiquetas").getString(
"Completed");
                    break;
 Duplicated block. Click for details.
                    status = ResourceBundle.getBundle("Etiquetas").getString("Accepted"
);
                    break;
                case "Rejected":
                    status = ResourceBundle.getBundle("Etiquetas").getString("Rejected"
);
                    break;
                case "NotCompleted":
                    status = ResourceBundle.getBundle("Etiquetas").getString(
"NotCompleted");
                    break;
                case "Complained":
                    status = ResourceBundle.getBundle("Etiquetas").getString(
"Complained");
                    break;
                case "Valued":
                    status = ResourceBundle.getBundle("Etiquetas").getString("Valued");
                    break;
                default:
                    status = ResourceBundle.getBundle("Etiquetas").getString(
"NotDefined");
                    break;
                }
```

Errefaktorizatutako kodea:



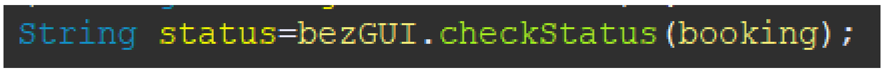
```
public String checkStatus(Booking booking) {
        switch (booking.getStatus()) {
        case "Completed":
            return ResourceBundle.getBundle("Etiquetas").getString("Completed");
        case "Accepted":
            return ResourceBundle.getBundle("Etiquetas").getString("Accepted");
        case "Rejected":
            return ResourceBundle.getBundle("Etiquetas").getString("Rejected");
        case "NotCompleted":
            return ResourceBundle.getBundle("Etiquetas").getString("NotCompleted");
        case "Complained":
            return ResourceBundle.getBundle("Etiquetas").getString("Complained");
        case "Valued":
            return ResourceBundle.getBundle("Etiquetas").getString("Valued");
        default:
            return ResourceBundle.getBundle("Etiquetas").getString("NotDefined");
        }
}
```

BezeroGUI:



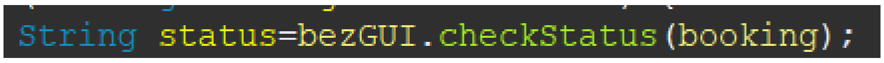
```
String status=checkStatus(bo);
```

EgoeraGUI:



```
String status=bezGUI.checkStatus(booking);
```

ErreserbaOnartuGUI:



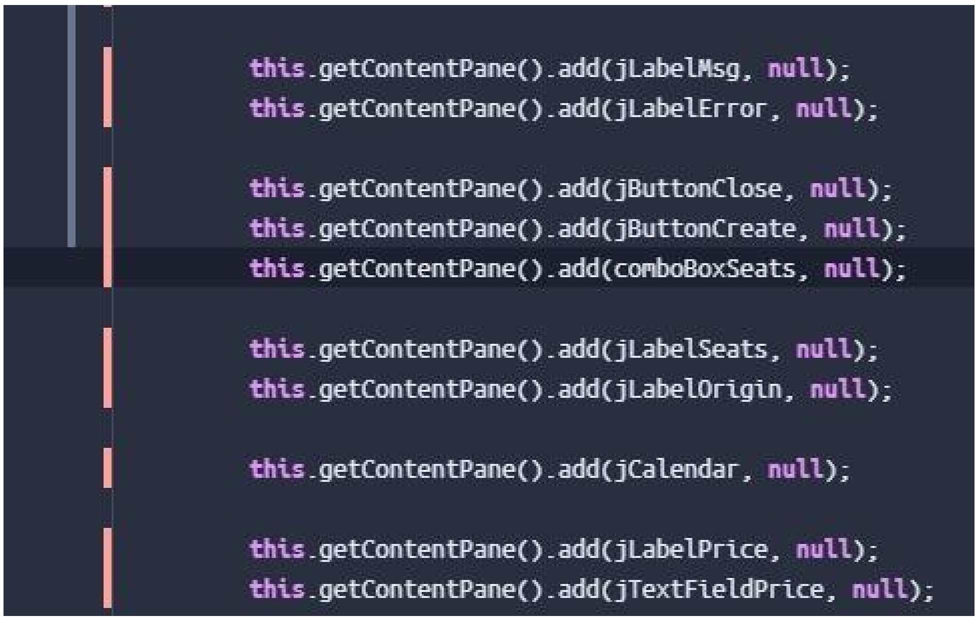
```
String status=bezGUI.checkStatus(booking);
```

Deskribapena:

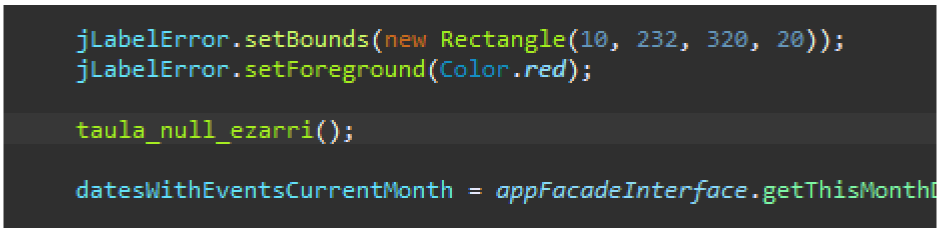
Hasierako kodearen irudian ikusi dezakegunez, etiketen esleipenaren kode bloke hori errepikatuta agertzen da, kasu honetan hiru klase ezberdinetan: BezeroGUI, EgoeraGUI eta ErreserbaOnartuGUI klaseetan hurrenez hurren. Kode duplikatuaren arazoa konpontzeko checkStatus metodoa inplementatu dut, bertan gauzatzeko status atributuaren esleipena, beste klaseetan berrerabili ahal izateko.

gui/CreateRideGUI (Ibai Oñatibia)

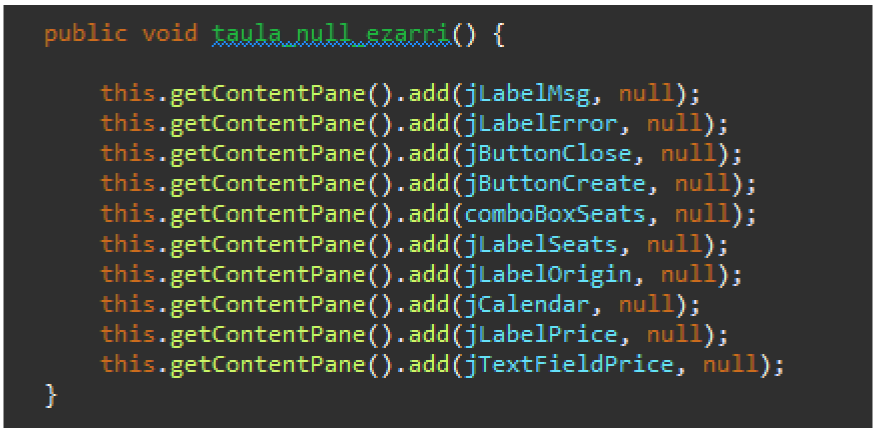
Errefaktorizazio kodea:



```
        this.getContentPane().add(jLabelMsg, null);
        this.getContentPane().add(jLabelError, null);

        this.getContentPane().add(jButtonClose, null);
        this.getContentPane().add(jButtonCreate, null);
        this.getContentPane().add(comboBoxSeats, null);

        this.getContentPane().add(jLabelSeats, null);
        this.getContentPane().add(jLabelOrigin, null);

        this.getContentPane().add(jCalendar, null);

        this.getContentPane().add(jLabelPrice, null);
        this.getContentPane().add(jTextFieldPrice, null);
```



```
    jLabelError.setBounds(new Rectangle(10, 232, 320, 20));
    jLabelError.setForeground(Color.red);

    taula_null_ezarri();

    datesWithEventsCurrentMonth = appFacadeInterface.getThisMonthD
```



```
public void taula_null_ezarri() {

    this.getContentPane().add(jLabelMsg, null);
    this.getContentPane().add(jLabelError, null);
    this.getContentPane().add(jButtonClose, null);
    this.getContentPane().add(jButtonCreate, null);
    this.getContentPane().add(comboBoxSeats, null);
    this.getContentPane().add(jLabelSeats, null);
    this.getContentPane().add(jLabelOrigin, null);
    this.getContentPane().add(jCalendar, null);
    this.getContentPane().add(jLabelPrice, null);
    this.getContentPane().add(jTextFieldPrice, null);
}
```

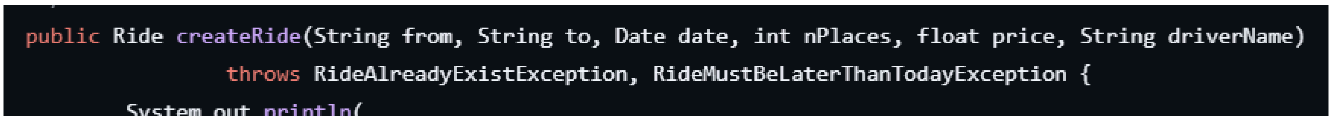
Deskribapena:

Kode zatian kodea errepikatzen zenez, bakarrik string-ak aldatuz, gauza bera egiteko metodo bat sortu dut. Ondoren gauza bera egin nahi bada beste leku batean, metodoa berrerabili ahal izateko.

"Keep unit interfaces small"

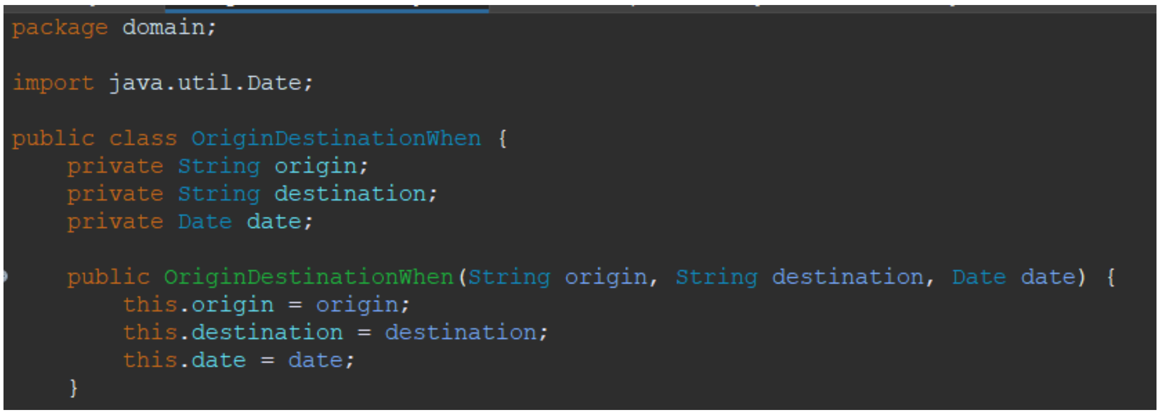
<u>createRide(Iker)</u>

Hasierako kodea:

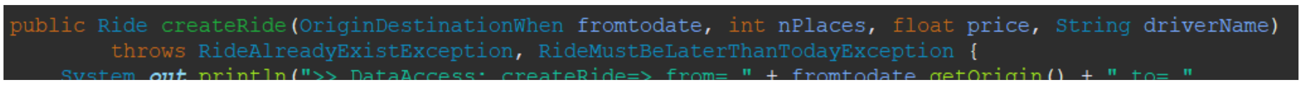


```
public Ride createRide(String from, String to, Date date, int nPlaces, float price, String driverName)
                throws RideAlreadyExistException, RideMustBeLaterThanTodayException {
        System.out.println(
```

Errefaktorizatutako kodea:



```java
package domain;

import java.util.Date;

public class OriginDestinationWhen {
    private String origin;
    private String destination;
    private Date date;

    public OriginDestinationWhen(String origin, String destination, Date date) {
        this.origin = origin;
        this.destination = destination;
        this.date = date;
    }
```



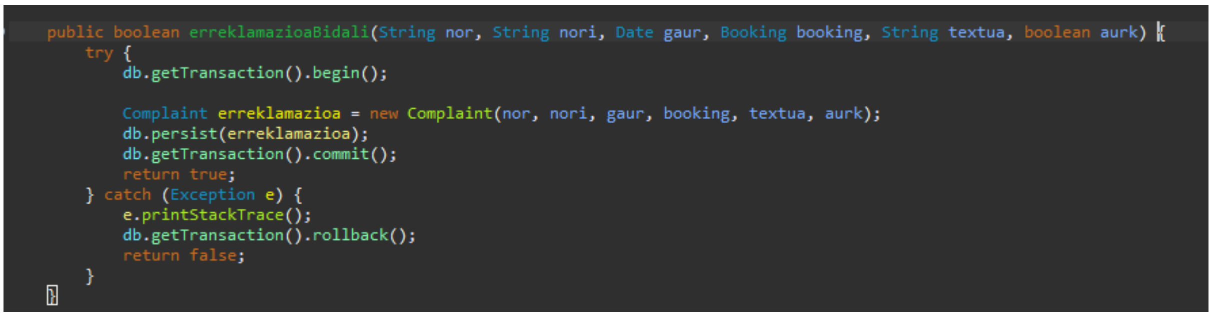
```java
public Ride createRide(OriginDestinationWhen fromtodate, int nPlaces, float price, String driverName)
        throws RideAlreadyExistException, RideMustBeLaterThanTodayException {
    System.out.println(">> DataAccess: createRide=> from= " + fromtodate.getOrigin() + " to= "
```

Deskribapena:

createRide metodoak 6 parametro erabiltzen ditu. Parametro kopurua 4 izan dadin OriginDestinationWhen klase berri bat sortu dut createRide metodoak erabiltzen dituen lehen hiru parametroak (from, to, date) jasotzeko.

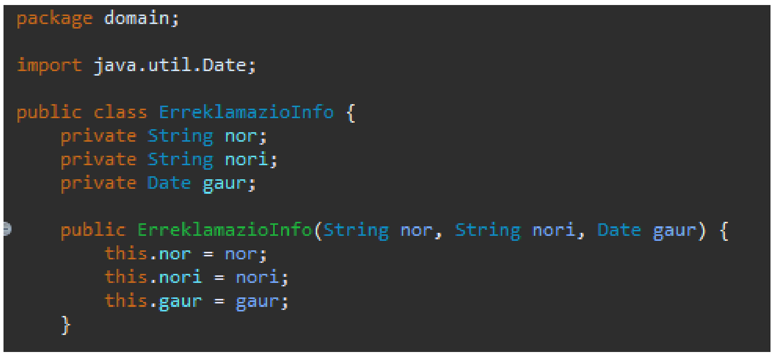
erreklamazioaBidali (Jon Olea)

Hasierako kodea:

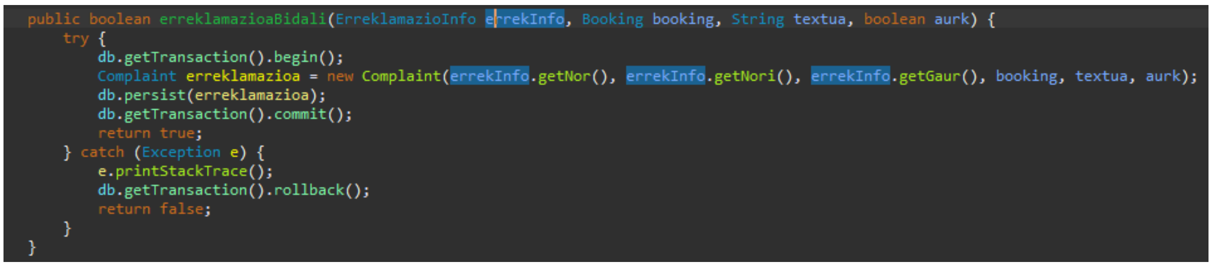


```java
    public boolean erreklamazioaBidali(String nor, String nori, Date gaur, Booking booking, String textua, boolean aurk) {
        try {
            db.getTransaction().begin();

            Complaint erreklamazioa = new Complaint(nor, nori, gaur, booking, textua, aurk);
            db.persist(erreklamazioa);
            db.getTransaction().commit();
            return true;
        } catch (Exception e) {
            e.printStackTrace();
            db.getTransaction().rollback();
            return false;
        }
    }
```

Errefaktorizatutako kodea:



```java
package domain;

import java.util.Date;

public class ErreklamazioInfo {
    private String nor;
    private String nori;
    private Date gaur;

    public ErreklamazioInfo(String nor, String nori, Date gaur) {
        this.nor = nor;
        this.nori = nori;
        this.gaur = gaur;
    }
```



```java
public boolean erreklamazioaBidali(ErreklamazioInfo errekInfo, Booking booking, String textua, boolean aurk) {
    try {
        db.getTransaction().begin();
        Complaint erreklamazioa = new Complaint(errekInfo.getNor(), errekInfo.getNori(), errekInfo.getGaur(), booking, textua, aurk);
        db.persist(erreklamazioa);
        db.getTransaction().commit();
        return true;
    } catch (Exception e) {
        e.printStackTrace();
        db.getTransaction().rollback();
        return false;
    }
}
```

Deskribapena:

erreklamazioaBidali metodoak 6 parametro zituenez, klase berri bat sortu dut erreklamazio informazioa gordetzeko (nork sortu duen, norentzat den eta data). Metodoa, klase berria erabili dut, guztira 4 parametro erabiliz.

Beste metodorik ez dugu aurkitu “Keep unit interfaces small” egiteko.