Git helbidea:

Martin Perez

1. getEvents metodoa vector bat bueltatu ordez ArrayList bat bueltatzea egin dut **change method signature** bidez. Hau sortu du beste hainbat klasetan eskuz aldaketak egitea eta vector batekin tratatu ordez arraylist batekin tratatzea egitea.

Lehen:

```
public Vector<Event> getEvents(Date date) {
    System.out.println(">> DataAccess: getEvents");
    Vector<Event> res = new Vector<Event>();
    TypedQuery<Event> query = db.createQuery("SELECT ev FROM Event ev WHERE ev.eventDate=?1"
    query.setParameter(1, date);
    List<Event> events = query.getResultList();
    for (Event ev : events) {
        System.out.println(ev.toString());
        res.add(ev);
    }
    return res;
}
```

Orain:

```
public ArrayList<Event> getEvents(Date date) {
    System.out.println(">> DataAccess: getEvents");
    ArrayList<Event> res = new ArrayList<Event>();
    TypedQuery<Event> query = db.createQuery("SELECT ev FROM Event ev WHERE ev.eventDate=?1"
    query.setParameter(1, date);
    List<Event> events = query.getResultList();
    for (Event ev : events) {
        System.out.println(ev.toString());
        res.add(ev);
    }
    return res;
}
```

2. ApustuaEgin metodoa luzeegia zen (“write short units of code”), ainbat metodoetan zatitu du **extract method** erabiliz. Hau ere balio du “write simple units of code” code smell-a zuzentzeko konplexutasuna jaitsi delako.

Lehen:

```
public boolean apustuEgin(Registered user, float dirua, List<String> kuotak, boolean kopiatuta, int boleto) {
    Registered r = (Registered) db.find(User.class, user);
    Boleto b = null;
    if (!kopiatuta || boleto != 0) {
        b = db.find(Boleto.class, boleto);
    }
    Apustua apustua = null;
    if (r.getDirua() - dirua >= 0) {
        Vector<Kuota> kl = new Vector<Kuota>();
        for (String str : kuotak) {
            Kuota k = db.find(Kuota.class, str);
            kl.add(k);
        }
        db.getTransaction().begin();
        apustua = new Apustua(dirua, user, kl, kopiatuta, b);
        Mugimendua m = new Mugimendua(dirua, ResourceBundle.getBundle("Etiquetas").getString("HaveStaked"), user);
        r.setDirua(r.getDirua() - dirua);
        r.addApustua(apustua);
        r.addMugimendua(m);
        r.getBoletoak().remove(b);
        for (Kuota k : kl) {
            k.addApustua(apustua);
        }
        db.getTransaction().commit();
        for (Jarraitzailea f : r.getFollowers()) {
            apustuEgin(f.getJarraitzailea(), dirua * f.getMurriztapena(), kuotak, true, 0);
        }
        return true;
    } else
        return false;
}
```

Horain:

```
public boolean apustuEgin(Registered user, float dirua, List<String> kuotak, boolean kopiatuta, int boleto) {
    Registered r = (Registered) db.find(User.class, user);
    Boleto b = null;
    b = boletoBaliozko(kopiatuta, boleto, b);
    if (r.getDirua() - dirua >= 0) {
        Vector<Kuota> kl = new Vector<Kuota>();
        getKuotakAndAdd(kuotak, kl);
        db.getTransaction().begin();
        apustua(user, dirua, kopiatuta, r, b, kl);
        db.getTransaction().commit();
        doApustuaForFollowers(dirua, kuotak, r);
        return true;
    } else
        return false;
}

private void doApustuaForFollowers(float dirua, List<String> kuotak, Registered r) {
    for (Jarraitzailea f : r.getFollowers()) {
        apustuEgin(f.getJarraitzailea(), dirua * f.getMurriztapena(), kuotak, true, 0);
    }
}
```

```
private void apustua(Registered user, float dirua, boolean kopiatuta, Registered r, Boleto b, Vector<Kuota> kl) {
    Apustua apustua;
    apustua = new Apustua(dirua, user, kl, kopiatuta, b);
    Mugimendua m = new Mugimendua(dirua, ResourceBundle.getBundle("Etiquetas").getString("HaveStaked"), user);
    r.setDirua(r.getDirua() - dirua);
    r.addApustua(apustua);
    r.addMugimendua(m);
    r.getBoletoak().remove(b);
    for (Kuota k : kl) {
        k.addApustua(apustua);
    }
}

private void getKuotakAndAdd(List<String> kuotak, Vector<Kuota> kl) {
    for (String str : kuotak) {
        Kuota k = db.find(Kuota.class, str);
        kl.add(k);
    }
}

private Boleto boletoBaliozko(boolean kopiatuta, int boleto, Boleto b) {
    if (!kopiatuta || boleto != 0) {
        b = db.find(Boleto.class, boleto);
    }
    return b;
}
```

3. Errepikatuta zegoen initialize stringa clasearen zehar (hemen bakarrik bi aldiz agertzen da soilik baina gehiago zeuden) beran **local variable** bezala atera dugu. (“Duplicate code”)

Lehen:

```
public BLFacadeImplementation()  {
    System.out.println("Creating BLFacadeImplementation instance");
    ConfigXML c=ConfigXML.getInstance();

    if (c.getDataBaseOpenMode().equals("initialize")) {
        dbManager=new DataAccess(c.getDataBaseOpenMode().equals("initialize"));
        dbManager.initializeDB();
        } else
         dbManager=new DataAccess();
    dbManager.close();
```

Horain:

```
public BLFacadeImplementation()  {
    System.out.println("Creating BLFacadeImplementation instance");
    ConfigXML c=ConfigXML.getInstance();

    String initialize = "initialize";
    if (c.getDataBaseOpenMode().equals(initialize)) {
        dbManager=new DataAccess(c.getDataBaseOpenMode().equals(initialize));
        dbManager.initializeDB();
        } else
         dbManager=new DataAccess();
    dbManager.close();
```

4. 5 parametro zituenez, object berri bat sortu dut honekin laguntzeko **introduce parameter object** refactorizazioaz baliatuta

Lehen:

```
public boolean apustuEgin(Registered user, float dirua, List<String> kuotak, boolean kopiatuta, int boleto) {
    Registered r = (Registered) db.find(User.class, user);
    Boleto b = null;
    b = boletoBaliozko(kopiatuta, boleto, b);
    if (r.getDirua() - dirua >= 0) {
        Vector<Kuota> kl = new Vector<Kuota>();
        getKuotakAndAdd(kuotak, kl);
        db.getTransaction().begin();
        apustua(user, dirua, kopiatuta, r, b, kl);
        db.getTransaction().commit();
        doApustuaForFollowers(dirua, kuotak, r);
        return true;
    } else
        return false;
}
```

Horain:

```
public boolean apustuEgin(ApustuEginParameter parameterObject) {
    Registered r = (Registered) db.find(User.class, parameterObject.user);
    Boleto b = null;
    b = boletoBaliozko(parameterObject.kopiatuta, parameterObject.boleto, b);
    if (r.getDirua() - parameterObject.dirua >= 0) {
        Vector<Kuota> kl = new Vector<Kuota>();
        getKuotakAndAdd(parameterObject.kuotak, kl);
        db.getTransaction().begin();
        apustua(parameterObject.user, parameterObject.dirua, parameterObject.kopiatuta, r, b, kl);
        db.getTransaction().commit();
        doApustuaForFollowers(parameterObject.dirua, parameterObject.kuotak, r);
        return true;
    } else
        return false;
}
```

Ibon Garcia

1. Apustua klaseko getKuotak metodoa Vector bat itzuli ordez, ArrayList bat itzultzea (**change method signature**)

Lehen:

```
public Vector<Kuota> getKuotak() {
    return kuotak;
}
```

Orain:

```
public ArrayList<Kuota> getKuotak() {
    return kuotak;
}
```

2. Registered klaseko getBoletoak Vector itzuli ordez, ArrayList bat itzultzea (**change method signature**)

Lehen:

```
public Vector<Boleto> getBoletoak() {
    return boletoak;
}
```

Orain:

```
public ArrayList<Boleto> getBoletoak() {
    return boletoak;
}
```

3. Kuota klaseko getApustuak metodoa Vector itzuli ordez, ArrayList bat itzultzea (**change method signature**)

Lehen:

```
public Vector<Apustua> getApustuak() {
    return apustuak;
}
```

Orain:

```
public ArrayList<Apustua> getApustuak() {
    return apustuak;
}
```

4. for begizta metodo baten bidez aldatzea User klaseko getMezuaId metodoan. (**Extracting a Method from Code**)

Lehen:

```
public Mezua getMezuaId(int id) {
    Mezua ema = null;
    for (Mezua m:Mezuak) {
        if (m.getMezuId()==id) {
            ema = m;
        }
    }
    return ema;
}
```

Orain:

```
public Mezua getMezuaId(int id) {
    Mezua ema = null;
    ema = mezuId(id, ema);
    return ema;
}
```

```
private Mezua mezuId(int id, Mezua ema) {
    for (Mezua m:Mezuak) {
        if (m.getMezuId()==id) {
            ema = m;
        }
    }
    return ema;
}
```

MARKEL MUTUBERRIA

1.- Question klaseko getFees metodoa Vector itzuli ordez, ArrayList bat itzultzea (**change method signature**)

Lehen:

```
public Vector<Kuota> getFees() {
    return fees;
}
```

Orain:

```
public ArrayList<Kuota> getFees() {
    return fees;
}
```

2.- DataAcces klaske getAllQuestions metodoan **change method signature** egin List<Question> bat itzuli ordez ArrayList<Question> bat itzultzeko amaieran cast bat eginez

Lehen:

```
public List<Question> getAllQuestions(){
    TypedQuery<Question> query = db.createQuery("SELECT q FROM Question q", Question.class);
    return query.getResultList();
}
```

Horain:

```
public ArrayList<Question> getAllQuestions(Object newParam){
    TypedQuery<Question> query = db.createQuery("SELECT q FROM Question q", Question.class);
    return (ArrayList<Question>) query.getResultList();
}
```

3.- User klaseko getMezuaErabiltzaile metodoan **Extracting a Method from Code** erabili for begizta aldatzeko.

Lehen:

```
public Mezua getMezuaErabiltzaile(String u2) {
    Mezua ema = null;
    for (Mezua m:Mezuak) {
        if ((m.User1.getUsername().equals(this.getUsername())&&m.User2.getUsername().equals(u2)) || (m.User1.getUsername
            ema = m;
        }
    }
    return ema;
}
```

Horain:

```
public Mezua getMezuaErabiltzaile(String u2) {
    Mezua ema = null;
    ema = extracted(u2, ema);
    return ema;
}

private Mezua extracted(String u2, Mezua ema) {
    for (Mezua m:Mezuak) {
        if ((m.User1.getUsername().equals(this.getUsername())&&m.User2.ge
            ema = m;
        }
    }
    return ema;
}
```

4.- DataAccess klasko getAllBets metodoan **change method signature** egin List<Question> bat itzuli ordez ArrayList<Question> bat itzultzeko amaieran cast bat eginez

Lehen:

```
public List<Apustua> getAllBets(){
    TypedQuery<Apustua> query = db.createQuery("SELECT a FROM Apustua a", Apustua.class);
    return query.getResultList();
}
```

Horain:

```
public ArrayList<Apustua> getAllBets(){
    TypedQuery<Apustua> query = db.createQuery("SELECT a FROM Apustua a", Apustua.class);
    return (ArrayList<Apustua>) query.getResultList();
}
```