



Sarrera

Proiektu honen helburua, hiru mailako software arkitektura batean diseinatutako **apustuak** kudeatzen dituen informazio sistema baten garapena izango da.

Orain arte (0 iterazioa) egindako proiektuaren iturburu kodea <https://github.com/jononekin/Bets21> estekan aurkitu dezakezu.

Dokumentu honetan orain arte garatutako informazioa eskaintzen da (0 iterazioa):

1. Informazio sistemaren deskribapena.
2. Domeinuaren eredua.
3. Erabilpen kasuen eredua.
4. "Create question" erabilpen kasuaren sekuentzia diagrama.
5. Iturburu kodea eta eclipse instalazioa.
6. Aplikazioaren egitura.
7. Aplikazioa martxan jartzeko argibideak.
8. Iturburu kodea.
9. Iterazio bakoitzean aurkeztu behar den gutxienezko dokumentazioa.

1. Informazio sistemaren deskribapena

Kirol apustuak kudeatzeko aplikazio bat garatu nahi dugu.

Sistemaren administratzaileak sistemaren gertaerak (adibidez, Real Madrid/Barcelona partidua), gertaeren galderak (adibidez, nork irabaziko du partidua? , edo nork sartuko du lehendabiziko gola?) eta galdera bakoitzaren pronostikoak bere irabaziarekin sortuko ditu. Adibidez, aurreko galdererako, 2 pronostiko irabazi desberdinarekin sortu daitezke:

Gertaera: Real Madrid vs. Barcelona
Galdera: Nork irabaziko du? Ohiko 1-X-2
Apustu minimoa: 2 euro
Pronostikoa: Real Madrid
Irabazia: 1,20 €

Gertaera: Real Madrid vs. Barcelona
Galdera: Nork irabaziko du? Ohiko 1-X-2
Apustu minimoa: 2 euro
Pronostikoa: Barcelona
Irabazia: 2 €

Gertaera baten ezaugarriak bere deskripzioa eta data dira. Galdera bakoitzak hiru ezaugarri dauzka: galderaren deskripzioa, zein gertaeran oinarritzen den eta apustu kantitate minimoa (adb: 2 euro). Bukatzeko, pronostiko bat galdera baten emaitzaz eta apustutako euro bokoitzangatik euro irabazia deskribatuta dago.

Erabiltzaile erregistratuek pronostiko batengan apustu bat egin dezakete hurrengo baldintzak betetzen badira:

- a) Pronostikoa oinarritzen den gertaera oraindik ez da gertatu, b) apustu kantitatea galdera horretan definitutako apustu kantitatea minimoa baino handiagoa da eta c) bere kontuan nahiko diru dago apustu hori egiteko.

Apustu adibide bat hurrengo izan liteke:

Gertaera: Real Madrid vs. Barcelona

Galdera: Nork irabaziko du? Ohiko 1-X-2

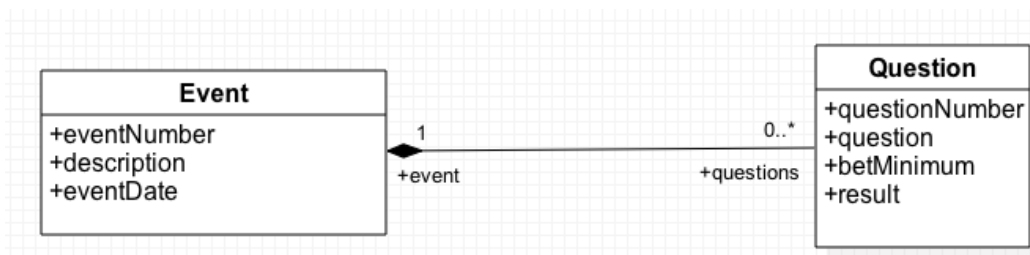
Pronostikoa: Real Madrid

Apustu kantitatea: 30 euro. (asmatzen badu 36 euro irabaziko ditu, bestela, dena galduko du).

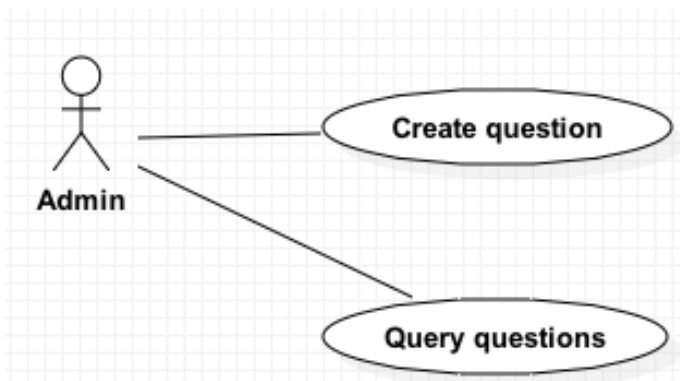
Gertaera bukatzen denean, administratzaileak galderen emaitzak txertatzen ditu (adib. Real Madrid-ek eta Barcelona-k berdindu egin dute), eta sistemak bezeroek egin dituzten apustuen arabera bere kontuak eguneratzen ditu. Emaitza apustutako pronostikoarekin bat badator, erabiltzailearen kontua apustutako kantitatea bider irabazi kantitateaz gehituko da, beste kasuan, apustutako dena galduko du.

Edozein momentuan, erabiltzaileek egindako apustuak, beren egoera (bukatuta edo ez), eta bere etekina (bukatu bada) kontsultatzeko aukera izan beharko dute.

2. Domeinuaren eredua



Erabilpen Kasuen eredua



Gertaera fluxuak

Query question Flow of events

Basic Flow

1. *System* shows a Calendar where days with events are highlighted
2. *Admin* selects a **Date** in a Calendar
3. *System* displays the **events** of this **date**
4. *Admin* selects an **event**
5. *System* displays the **questions** associated to the selected **event**

Alternative flow

1. There are no events on this date. Events cannot be shown.
2. There are no questions associated to the event. Questions cannot be shown.

Create question Flow of events

Basic Flow

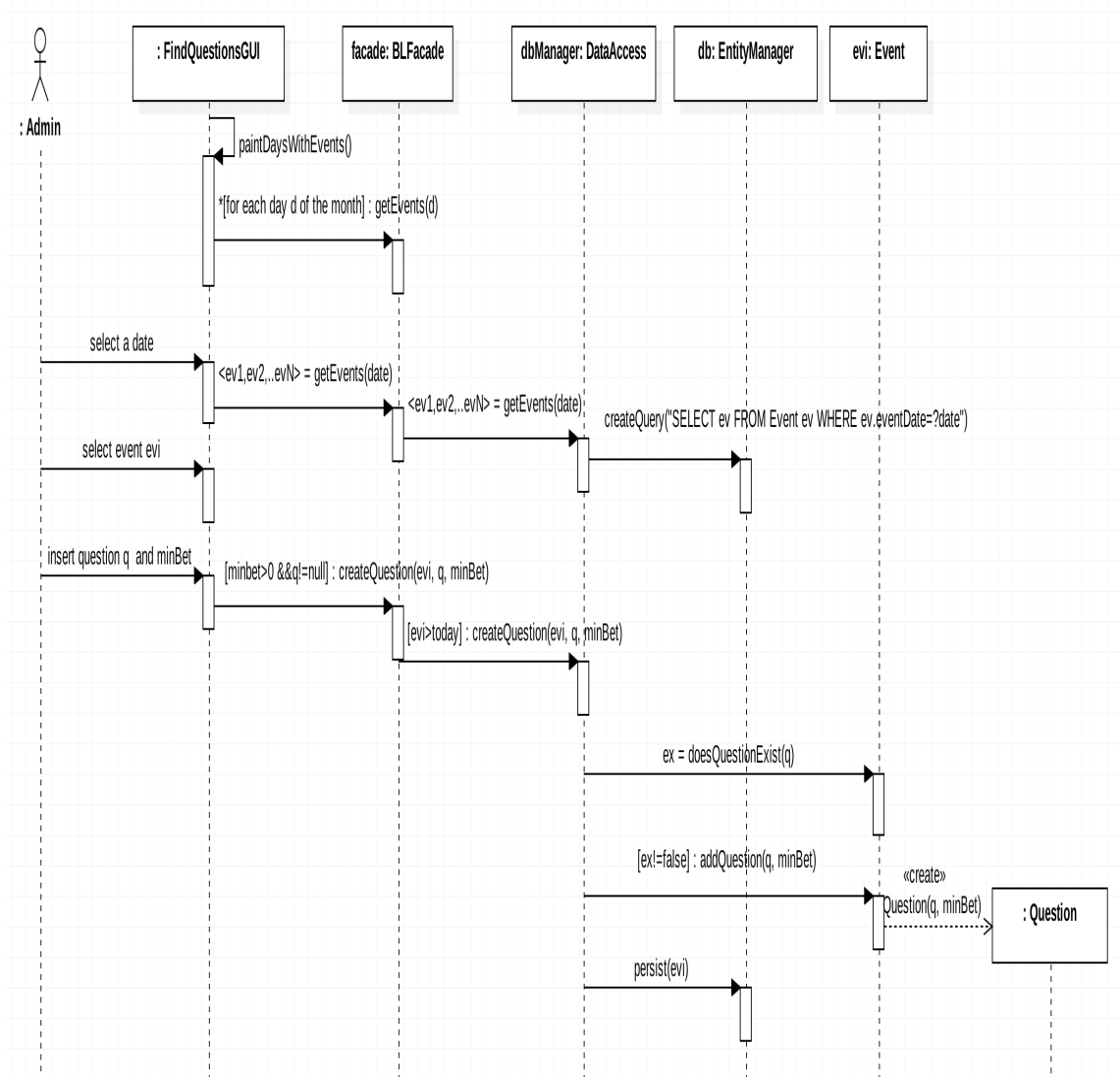
1. *System* shows a Calendar where days with events are highlighted
2. *Admin* selects a **Date** in a Calendar
3. *System* displays the **events** of this **date**
4. *Admin* selects an **event**
5. *Admin* introduces a **question** and a minimum **betting price**
6. *System* adds the new **question** with a minimum **betting price** to the selected **event**

Alternative flow

1. There are no events on this date. Question cannot be added.
2. The question field is empty. Question cannot be added.
3. The minimum betting price is empty or it is not a number. Question cannot be added.
4. Event date has already finished (event day is before current day). Question cannot be added.


4. "Create question" erabilpen kasuaren sekuentzia diagrama

Aplikazioa hiru mailako arkitektura batean diseinatuta dago. AWT/SWING lengoaia erabili da erabiltzaileraren interfaze grafikoak definitzeko, negozio logikaren atzipena Web Zerbitzuren bitartez garatu da, eta persistentzia mailarako, objektuetan oinarritutako objectdb datu basea erabili da. Negozio Logikaren eragiketa guztiak biltzeko, eta GRASP patroia gomendioak jarraituz, Facade klase bat definitu da eragiketa guztiekin. Jarraian sistemaren erabilpen kasu baten sekuentzia diagrama aurkezten da:




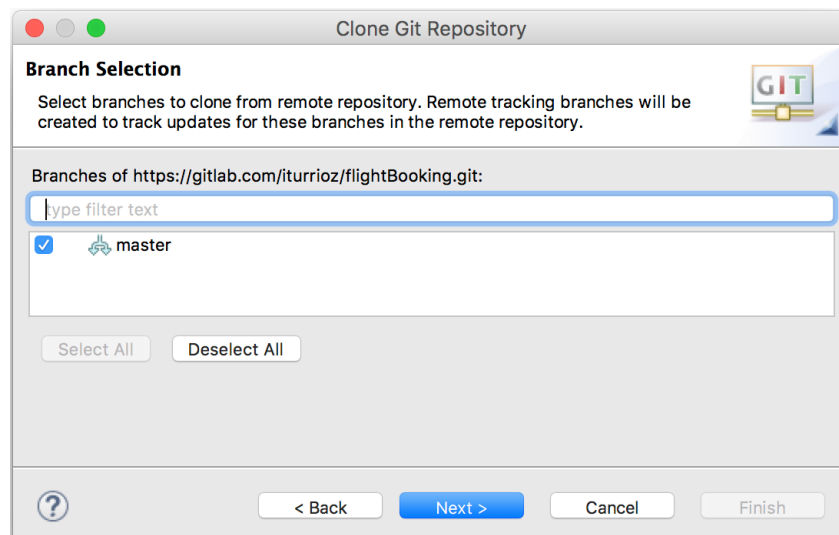
5. Iturburu kodea eta eclipse instalazioa.

Orain arte (0 iterazioa) egindako proiektuaren iturburu kodea <https://github.com/jononekin/Bets21> estekan aurkitu dezakezu.

Lehendabizi <https://github.com/jononekin/lab2GUI-NL> joan eta goiko partean dagoen  Fork botoia sakatu eta eskatzen diren pausoak jarraitu proiektua kopiatzeko. Honen ondorioz, proiektuaren kopia bat izango duzue zuen Github kontuan.

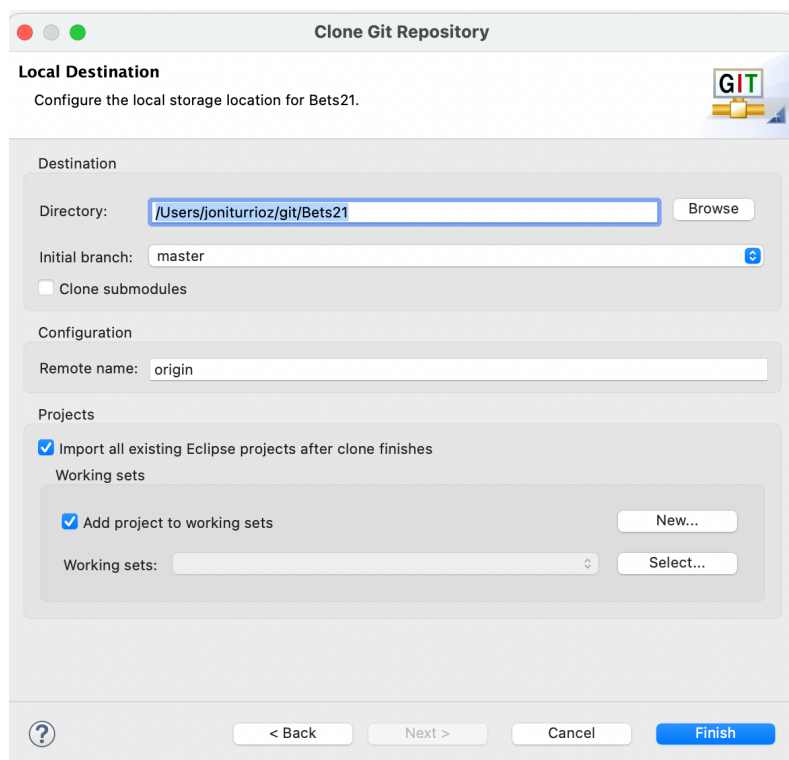
Jarraian, Eclipse-ko **Git Repositories** leihoan, **Clone a Git repository and add the**

 **clone to this view** botoia aukeratu. Kopiatu zuen urruneko errepositoriaren URL-a (hau da, zuen kontuan sortu duzuen errespositorioaren kopiaren helbidea), zure kredentzialak eta **Next** sakatu. Hurrengo **Branch Selection window** irudian berriz **Next** egin.



Irudia: **Branch Selection** leihoa, urruneko errepositorioaren zein zati gure errespositorio lokalean kopiatu eta sinkronizatu nahi dugun adierazteko.

Eclipse-k zure lokal errepositorio non utzi jakin behar du (Urruneko errepositorioarekin sinkronizatzen den kopia), hau **Local Destination** leihoan agertzen da. **Oso garrantzitsua da** "Add project to working sets" aukeratzea Finish sakatu baino lehen.



Irudia: **Local Destination** leihoa, klonatutako urruneko errepositorioaren helbidea adierazteko.

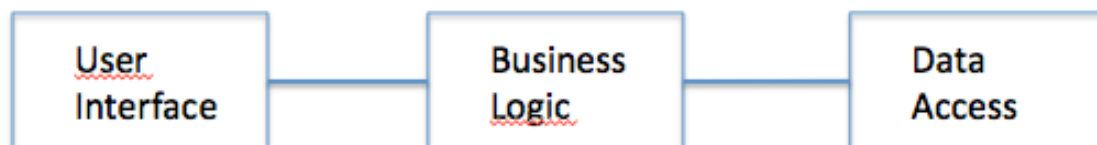
Honen ondorioz, proiektua eclipseko proiektuen artean agertuko da. Github-eri buruz gehiago jakin nahi baduzu (adibidez, eclipsen egin dituzun aldaketak Github-era igo), egelan dagoen 8. Laborategia kontsultatu dezakezu.

Proiektuan karpeta desberdinak egongo dira:

- (src/main/java) karpetan, proiektuaren Java iturburu kodea egongo da.
- (src/main/resources) karpetan proiektuak behar dituen baliabide osagarriak kokatuko dira, hau da, datubase eta egutegi batekin lan egiteko JCalendar eta objectDb liburutegiak, datubasea (bets.temp) eta proiektuaren konfigurazio fitxategia (config.xml).

6. Aplikazioaren egitura.

Aplikazioa hiru maila logikoetan banatuta dago: Interfaze grafikoa, Negozio Logika, eta Datu Atzipena.



Aplikazioa 5 java paketetan antolatuta dago:

gui: Interfaze grafikoko klaseak (User Interface tier).

businessLogic: Negozio logikako klaseak (Business Logic tier).

dataAccess: Datu baseko klaseak (Data Access tier).



domain: Domeinu ereduan definitu diren klaseak.

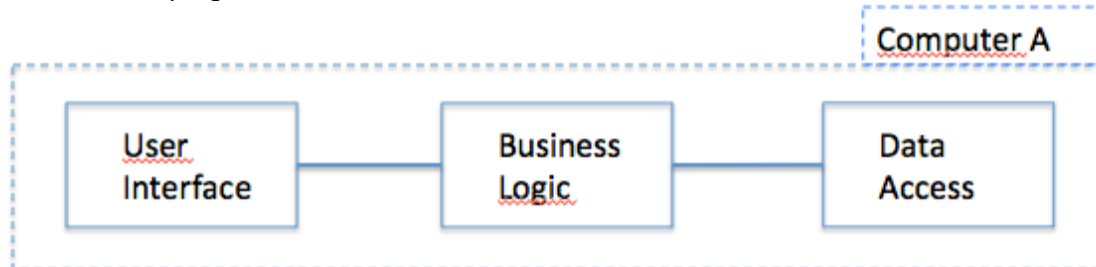
exception: Salbuespenak definitzen dituzten klaseak.

Maila bakoitza, makina berdinean edo beste ordenagailu desberdin batean exekutatu daiteke. Informazio hori *config.xml* fitxategian definitzen da. Ikus dezagun existitzen diren aukerak.

7. Aplikazioa martxan jartzeko argibideak.

Aukera 1: Aplikazioa guztia maila fisiko batean.

Aukera honetan Aurkezpena, Negozio Logika eta Datu Atzipen mailak ordenagailu berdinean despleгатzen dira.



Horretarako, config.xml fitxategia honela konfiguratu behar da:

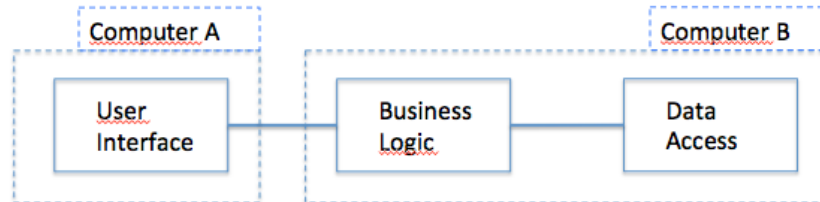
```
<?xml version="1.0" encoding="UTF-8"?>
<config>
  <businessLogic local="true">
    <businessLogicNode>0.0.0.0</businessLogicNode>
    <businessLogicPort>1099</businessLogicPort>
    <businessLogicName>Bets</businessLogicName>
  </businessLogic>
  <database local="true">
    <databaseNode>0.0.0.0</databaseNode>
    <dbFilename>bets.temp</dbFilename>
    .....
  </database>
</config>
```

Egoera honetan, aurkezpenak negozio logika (**<businessLogic>**) modu lokalean atzitzen duela (**local="true"**) eta negozio logikak datu basea (**<database>**) modu lokalean ere atzitzen duela (**local="true"**) erazagutzen dugu.

Ariketa: config.xml fitxategia konfiguratu eta ondoren *gui* paketea dagoen **ApplicationLauncher** klasea exekutatu.

Aukera 2: Aplikazioa bi maila fisikoetan.

Aukera honetan Aurkezpena maila makina batean, eta Negozio Logika eta Datu Atzipen mailak beste ordenagailu desberdinean desplegatzen dira.



Aplikazioa martxan jartzeko.

1. Aplikazioaren konfigurazioa definitu config.xml fitxategia aldatuz:

```

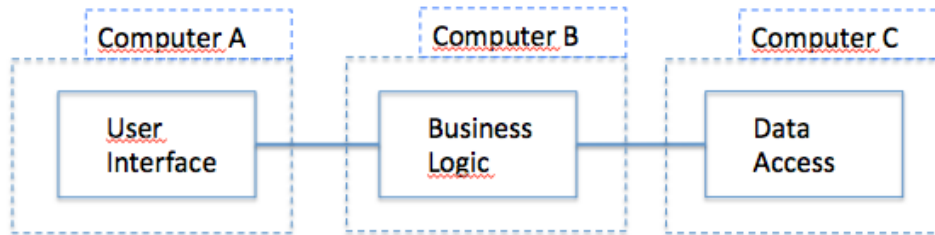
<?xml version="1.0" encoding="UTF-8"?>
<config>
  <businessLogic local="false">
    <businessLogicNode>0.0.0.0</businessLogicNode>
    <businessLogicPort>1099</businessLogicPort>
    <businessLogicName>Bets</businessLogicName>
  </businessLogic>
  <database local="true">
    <databaseNode>0.0.0.0</databaseNode>
    <dbFilename>bets.temp</dbFilename>
  </database>
</config>
  
```

Konfigurazio honetan, bussinessLogic lokalean ez dagoela erazagutzen dugu (**businessLogic local="false"**). Lokalean ez dagoenez, negozio logika non kokatuta dagoen erazagutu behar da, hau da: a) Zein makinatan dagoen negozio logika (**<businessLogicNode>**), (0.0.0.0. bertan gauden konputagailuaren helbidea adierazten du), zein portutan (**<businessLogicPort>**) eta zein da zerbitzuaren izena (**<businessLogicName>**).

2. Negozio logika exekutatu. Horretarako *businessLogic* paketearen dagoen **BusinessLogicServer** klasea exekutatu. Negozio Logikaren zerbitzaria martxan ipiniko da (zerbitzu bat bezala).
3. Aurkezpena exekutatu, hau da, aurreko *gui* paketearen **ApplicationLauncher** klasea exekutatu eta aurreko erabilpen kasuak frogatu.

Ariketa: Nola exekutatuko zenuke beste ordenagailu batean dagoen negozio logika? Bi ordenagailuen artean froga egin.

Aukera 3: Aplikazioa hiru maila fisikoetan.



Aukera honetan Aurkezpena maila ordenagailu batean, Negozio Logika beste batean eta Datu Atzipen maila beste ordenagailu batean desplegatzen dira.

Aplikazioa martxan jartzeko.

1. Aplikazioaren konfigurazioa definitu config.xml fitxategia aldatuz `<database local="true">` jartzen duen lekuan `<database local="false">` ipiniz.

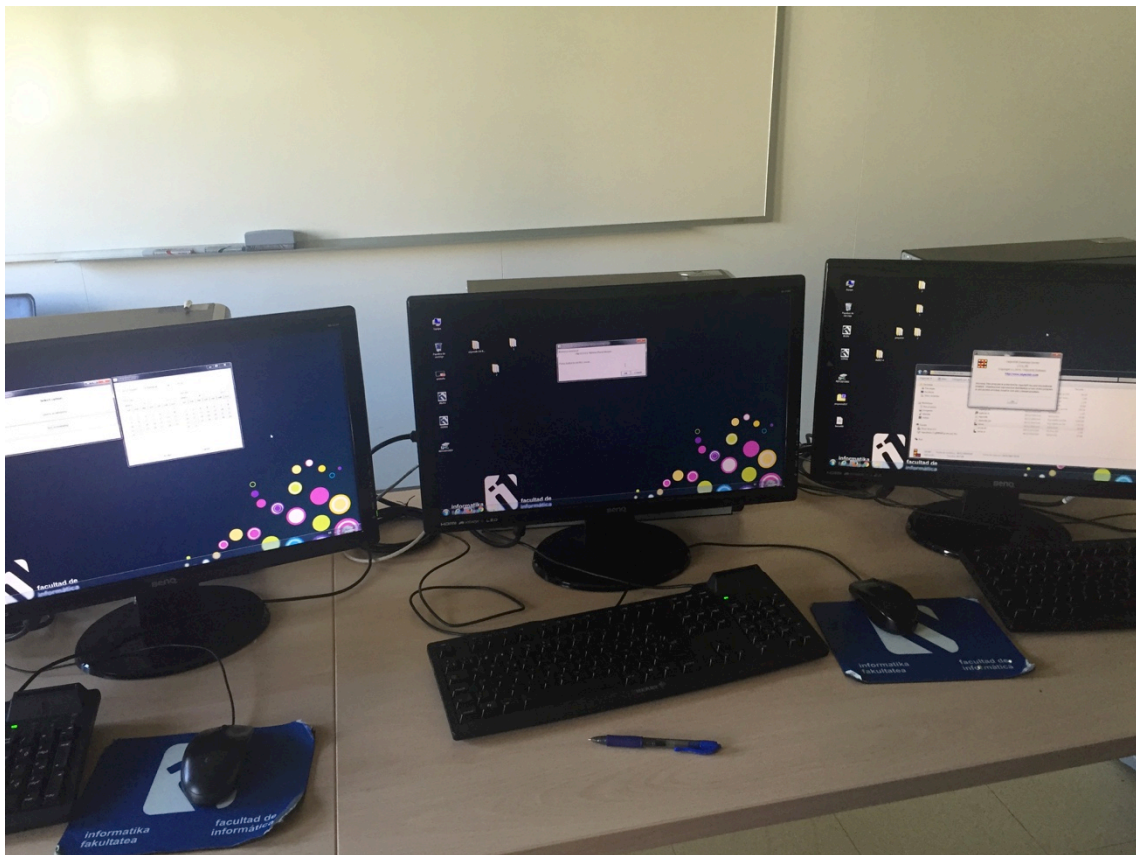
```

<?xml version="1.0" encoding="UTF-8"?>
<config>
  <businessLogic local="false">
    <businessLogicNode>0.0.0.0</businessLogicNode>
    <businessLogicPort>1099</businessLogicPort>
    <businessLogicName>Bets</businessLogicName>
  </businessLogic>
  <database local="false">
    <databaseNode>0.0.0.0</databaseNode>
    <dbFilename>bets.temp</dbFilename>
    .....
  </database>
</config>
  
```

2. Database zerbitzaria exekutatu. Horretarako, *dataAccess* paketea dagoen **ObjectdbManagerServer** exekutatu. Database zerbitzaria martxan ipiniko da (zerbitzu bat bezala).
3. Negozio logikaren zerbitzaria exekutatu, aurreko *businessLogic* paketea **BusinessLogicServer** klasea exekutatu.
4. Bukatzeko, aurkezpena exekutatu, hau da, *gui* paketearen **ApplicationLauncher** fitxategia.

Ariketa:

1. Nola desplegatuko zenuke Aurkezpena eta Negozio Logika ordenagailu batean eta Datu Basea beste makina desberdinean? Bi ordenagailuen artean froga egin.
2. Nola desplegatuko zenuke zure aplikazioa hiru ordenagailu desberdinetan? Azkenean, hurrengo argazkian agertzen den exekuzioa agertu beharko litzateke. Eskuineko ordenagailuan, datu zerbitzaria martxan. Erdiko ordenagailuan, negozio logikako zerbitzaria, eta ezkerrean interfaze grafikoa.



8. Iterazio bakoitzean aurkeztu behar den gutxienezko dokumentazioa.

Proiektua irakasgaian zehar garatuko da 3 iterazioetan. Iterazio bakoitzeko aurkeztu behar den beharrezko dokumentazioa hurrengoa da:

1. Aurkitutako arazoak. Zeintzuk dira eduki dituzuen arazo garrantzitsuenak eta nola bideratu dituzue.
2. Erabilpen kasu hedatuen eredua.
3. Domeinu eredu hedatua.
4. Diseinatutako erabilpen kasuen UML sekuentzia diagramak.
5. Diseinuzko klaseak
6. Implementazioaren iturburu-kodea formatu elektronikoan. Ez da inprimatu behar.