31-5-2018

5. BERTSIOA

*GUI eta datu baseak erabilita*

*Oihane Axpe Telleriarte*

*DAM 1-eko ikaslea*

*IES Uni Eibar-Ermua*

*Eibar, Gipuzkoa*

**Denda bat kudeatzeko aplikazioa**

Programazioa

**AURKIBIDEA**

[1. Sarrera 1](#_Toc515501969)

[2. Proiektua 2](#_Toc515501970)

[2.1. Klase diagrama 2](#_Toc515501971)

[2.2. DDBB-aren diagrama 6](#_Toc515501972)

[3. Proiektuaren egitura 7](#_Toc515501973)

[3.1. Klaseak 7](#_Toc515501974)

[3.2. Herentzia 10](#_Toc515501975)

[3.3. Atributuak 11](#_Toc515501976)

[3.4. Eraikitzaileak 12](#_Toc515501977)

[3.5. Metodoak 13](#_Toc515501978)

[3.5.1. Metodo estatikoak 15](#_Toc515501979)

[3.6. Getter & Setter 21](#_Toc515501980)

[3.7. Array-ak 23](#_Toc515501981)

[3.8. Fitxategiak 24](#_Toc515501982)

[3.9. Datu basea 26](#_Toc515501983)

[3.10. Interfaze grafikoa 27](#_Toc515501984)

[3.10.1. View - bistak 27](#_Toc515501985)

[3.10.2. Controller 28](#_Toc515501986)

[3.10.3. Main 31](#_Toc515501987)

[3.11. Erroreen kontrola 32](#_Toc515501988)

[4. Aplikazioa 34](#_Toc515501989)

**IRUDIEN TAULA**

[1. Irudia: Proiektuaren diagrama 2](#_Toc515505347)

[2. Irudia: Herentzia - Pertsona 3](#_Toc515505348)

[3. Irudia: Herentzia - Produktuak 4](#_Toc515505349)

[4. Irudia: Klase diagrama - MVC 5](#_Toc515505350)

[5. Irudia: DDBB diagrama 6](#_Toc515505351)

[6. Irudia: Paketeak 7](#_Toc515505352)

[7. Irudia: Argazkiak paketea 7](#_Toc515505353)

[8. Irudia: Controller paketea 8](#_Toc515505354)

[9. Irudia: Gestioa paketea 8](#_Toc515505355)

[10. Irudia: Model paketea 9](#_Toc515505356)

[11. Irudia: Main paketea 9](#_Toc515505357)

[12. Irudia: View paketea 9](#_Toc515505358)

[13. Irudia: Klase abstraktuak 9](#_Toc515505359)

[14. Irudia: Herentzia 10](#_Toc515505360)

[15. Irudia: Adibidea - Herentzia Pertsona 10](#_Toc515505361)

[16. Irudia: Adibidea - Herentzia Langilea 11](#_Toc515505362)

[17. Irudia: Adibidea - Herentzia Bezeroa 11](#_Toc515505363)

[18. Irudia: Atributuak 11](#_Toc515505364)

[19. Irudia: Atributuak (herentziarekin) 11](#_Toc515505365)

[20. Irudia: Adibidea – Instantzia berria sortu 12](#_Toc515505366)

[21. Irudia: Eraikitzaileetan gainkarga 12](#_Toc515505367)

[22. Irudia: Adibidea - Eraikitzaileak 12](#_Toc515505368)

[23. Irudia: Adibidea - Objektu berria 12](#_Toc515505369)

[23. Irudia: Adibidea - printDatuak() metodoa (Kamiseta) 13](#_Toc515505370)

[25. Irudia: Adibidea - printDatuak() metodoa (Produktua) 13](#_Toc515505371)

[25. Irudia: Adibidea - printDatuak() 13](#_Toc515505372)

[27. Irudia: Adibidea - Metodoak 14](#_Toc515505373)

[28. Irudia: Adibidea - prodKontsultatu() metodoa (Kamiseta) 14](#_Toc515505374)

[29. Irudia: Adibidea - prodKontsultatu() metodoa (Produktua) 14](#_Toc515505375)

[30. Irudia: Adibidea- prodKontsultatu() 15](#_Toc515505376)

[31. Irudia: Adibidea - Metodo estatiko bat definitu 15](#_Toc515505377)

[32. Irudia: Metodo estatikoak erabiltzen 15](#_Toc515505378)

[33. Irudia: Metodo estatikoak 16](#_Toc515505379)

[34. Irudia: Adibidea - Metodo estatikoa 16](#_Toc515505380)

[35. Irudia: Getter & Setter 21](#_Toc515505381)

[36. Irudia: Adibidea - Set gainkarga 21](#_Toc515505382)

[37. Irudia: Adibidea - Getter 22](#_Toc515505383)

[38. Irudia: Adibidea - Setter 22](#_Toc515505384)

[39. Irudia: Adibidea - Getter & Setter (Date mota) 22](#_Toc515505385)

[40. Irudia: Adibidea - Array 23](#_Toc515505386)

[41. Irudia: Array-ak 23](#_Toc515505387)

[42. Irudia: kodeakAldatuEtagorde(String hasiera): String metodoa 24](#_Toc515505388)

[43. Irudia: Adibidea - Getter & Setter (kodea automatikoki gehitzen) 25](#_Toc515505389)

[44. Irudia: kodeak.txt fitxategiko balioak 25](#_Toc515505390)

[45. Irudia: mysql-connector-java-5.1.46-bin.jar 26](#_Toc515505391)

[46. Irudia: Datu basera konexioa 26](#_Toc515505392)

[47. Irudia: MVC 27](#_Toc515505393)

[48. Irudia: Adibidea - KamisetaGehitu bista 27](#_Toc515505394)

[49. Irudia: Adibidea - ProdAukeratu bista 28](#_Toc515505395)

[50. Irudia: Controller 28](#_Toc515505396)

[51. Irudia: Adibidea – Akzioak gehitzen 29](#_Toc515505397)

[52. Irudia: Akzioak gehitzen 29](#_Toc515505398)

[53. Irudia: Akzioak inplementatu Controller-ean 30](#_Toc515505399)

[54. Irudia: Adibidea - KamiGehituController 30](#_Toc515505400)

[55. Irudia: Adibidea - Main 31](#_Toc515505401)

[56. Irudia: Adibidea - Errorea 32](#_Toc515505402)

[57. Irudia: Adibidea - try/catch 32](#_Toc515505403)

[58. Irudia: Adibidea - try/catch mezuak 32](#_Toc515505404)

[59. Irudia: Adibidea – Erabiltzaileak sartuko dituen datuak baliozkotzen 33](#_Toc515505405)

[60. Irudia: Main - Menu nagusia 34](#_Toc515505406)

[61. Irudia: Aplikazioa – denda kudeatu 34](#_Toc515505407)

[62. Irudia: Aplikazioa – denda ezabatu 35](#_Toc515505408)

[63. Irudia: Aplikazioa – denda gehitu 35](#_Toc515505409)

[64. Irudia: Aplikazioa – bezeroak kudeatu 36](#_Toc515505410)

[65. Irudia: Aplikazioa – denda ezabatu 36](#_Toc515505411)

[66. Irudia: Aplikazioa – bezeroa gehitu 37](#_Toc515505412)

[67. Irudia: Aplikazioa – langileak kudeatu 37](#_Toc515505413)

[68. Irudia: Aplikazioa – denda ezabatu 38](#_Toc515505414)

[69. Irudia: Aplikazioa – langilea gehitu 38](#_Toc515505415)

[70. Irudia: Aplikazioa – hornitzaileak kudeatu 39](#_Toc515505416)

[71. Irudia: Aplikazioa – denda ezabatu 39](#_Toc515505417)

[72. Irudia: Aplikazioa – hornitzailea gehitu 40](#_Toc515505418)

[73. Irudia: Aplikazioa – eskaerak kudeatu 40](#_Toc515505419)

[74. Irudia: Aplikazioa – denda ezabatu 40](#_Toc515505420)

[75. Irudia: Aplikazioa – eskaera gehitu 41](#_Toc515505421)

[76. Irudia: Aplikazioa – produktuak kudeatu 41](#_Toc515505422)

[77. Irudia: Aplikazioa – denda ezabatu 42](#_Toc515505423)

[78. Irudia: Aplikazioa – kamiseta gehitu 42](#_Toc515505424)

[79. Irudia: Aplikazioa – inbentarioa 43](#_Toc515505425)

[80. Irudia: Aplikazioa – inbentarioa 44](#_Toc515505426)

[81. Irudia: Aplikazioa – kontsulta 45](#_Toc515505427)

[82. Irudia: Aplikazioa – salmenta 45](#_Toc515505428)

**TAULAK**

[1. Taula: Modifikatzaileen desberdintasunak 4](#_Toc515502109)

[2. Taula: DDBB-ko erlazioak 6](#_Toc515502110)

[3. Taula: Erroreen kontrola 32](#_Toc515502111)

# **SARRERA**

Sortutako aplikazioa denda bat kudeatzeko sortu da.

Proiektu hau, Netbeans 8.2-rekin sortuta dago, java lengoaian idatzita.

Hau, bigarren bertsioa izango da eta dokumentu honek aldaketak jasan dezake edozein puntutan, proiektua aurrera doan heinean.

Dokumentu honek, honako estruktura hau dauka: Lehenengo, proiektuaren diagrama klasea nolakoa izango den azaltzen da. Ondoren, proiektuaren egitura nolakoa izango den azalduko da, hau da, klaseak, atributuak, herentzia, metodoak eta metodo estatikoak, getter eta setter-ak, array-ak, fitxategiak, interfaze grafikoa eta erroreen kontrola nola kudeatuko den. Gainera, kontsolaren nondik norakoak ere azalduko dira.

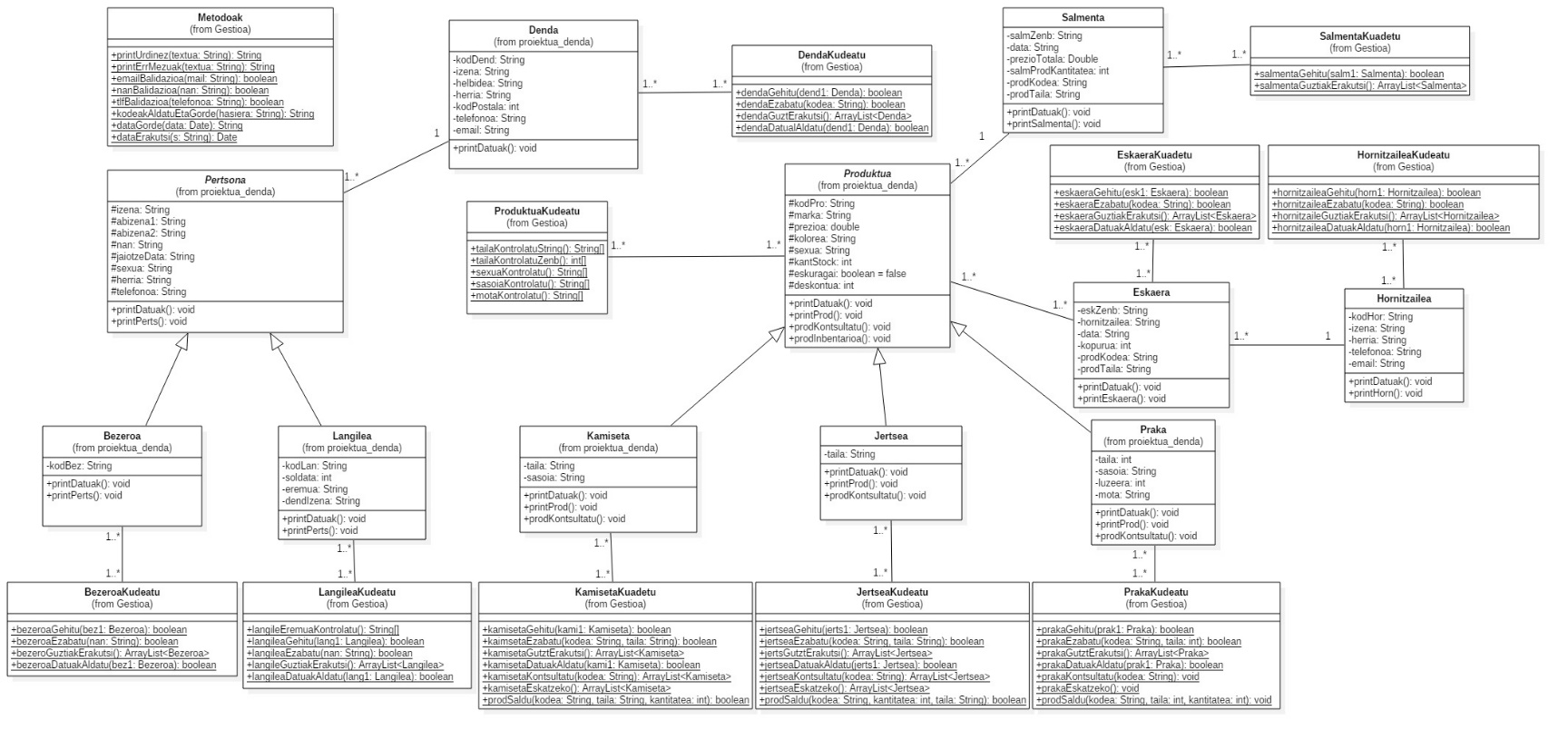
# **PROIEKTUA**

Aplikazio honek, denda bat gestionatzeko balio du. Dendan lan egingo duten langileak gestionatzeko aukera emango du. Bezeroak ere kudeatzeko aukera egongo da, eta salduko diren produktuak kudeatzeko ere erabiliko da, baita hornitzaileei produktuak eskatzeko ere.

## Klase diagrama

Proiektua kodifikatzen hasi aurretik, diagrama bat egin da, edukiko dituen klaseak, herentzia eta klaseen arteko erlazioak kontuan hartuta.

Ondorengo diagraman, proiektuak izango dituen klase guztiak ikusi daitezke.

Klase bakoitzak, atributuak eta metodoak dituzte, ondorengo irudian ikusten den bezala.

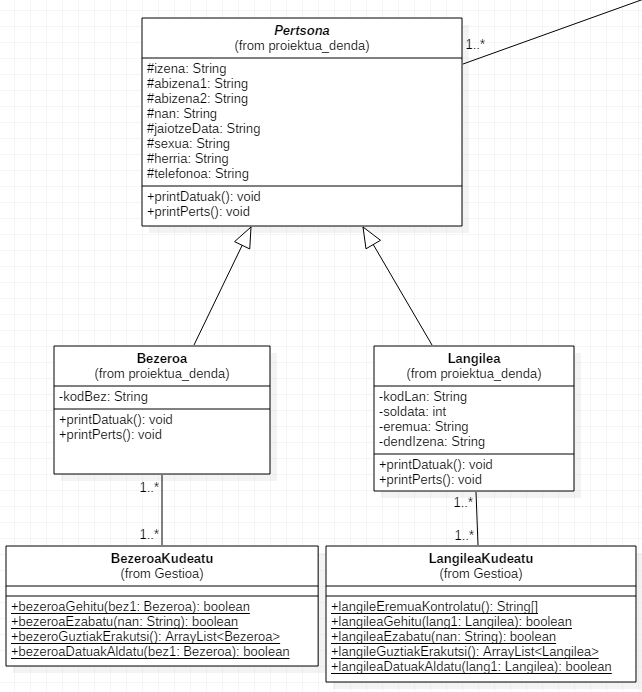
1. Irudia: Proiektuaren diagrama

Horrez gain, getter eta setter-ak ere izango dituzte. Hauek, metodo publikoak dira eta atributuetara heltzeko balio dute. Aurrerago ([Getter & Setter](#_Getter_&_Setter_1) puntuan), adibideekin batera azalduko dira gehiago.

Diagraman ikusten den bezala, proiektuak bi atal nagusi izango ditu. Alde batetik, bezero eta langileen kudeaketa eta bestetik, produktuen kudeaketa.

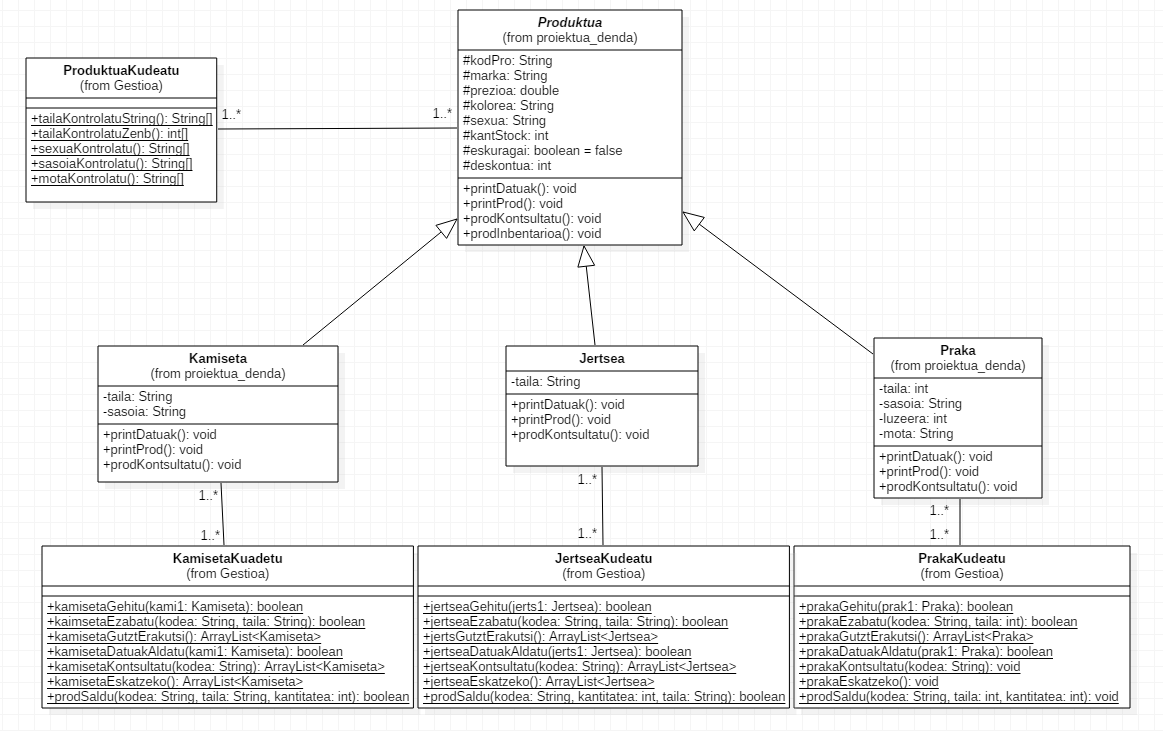
Proiektu honetan, bi herentzia izango ditugu. Bat pertsona eta bezero-langileen artean eta bestea, produktuena.

1. Pertsona klaseak, bezeroa eta langilea azpi klaseak ditu.



2. Irudia: Herentzia - Pertsona

1. Produktuak klaseak, kamiseta, jertseak eta prakak azpi klaseak ditu.



3. Irudia: Herentzia - Produktuak

Klaseetako atributu eta metodoak, publikoak, pribatuak edo babestuak izan daitezke, eta lortu nahi dugunaren araberakoak izango dira.

Ondorengo taulan ikusi daitezke modifikatzaileen desberdintasunak.

|  |  |  |
| --- | --- | --- |
| + | **public** | Edozein klaseetatik ikusi daiteke. |
| - | **private** | Klasetik kanpora ezin da ikusi. |
| # | **protected** | Klasetik bertatik eta azpi klaseetatik bakarrik ikusi daiteke. |

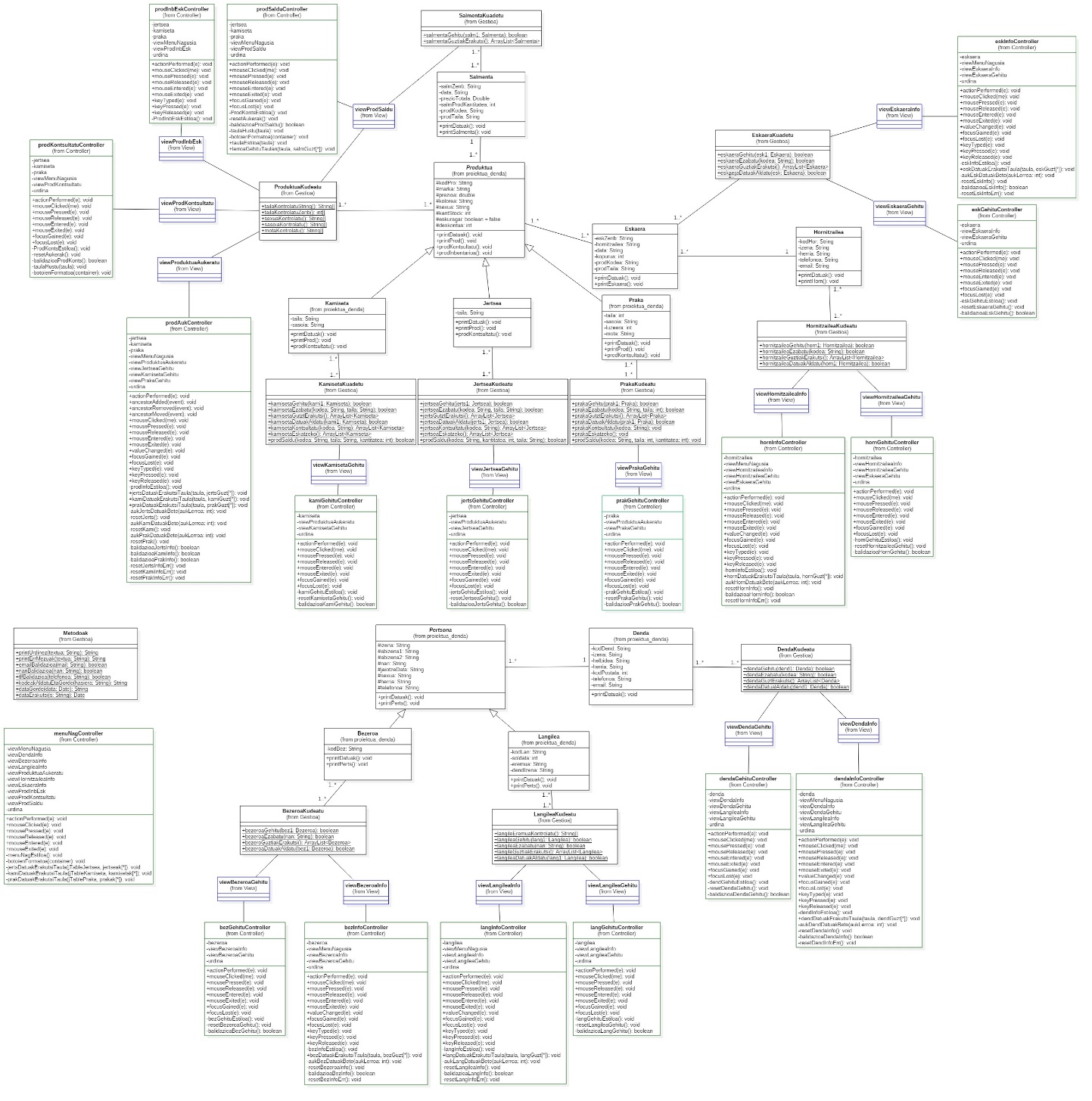
1. Taula: Modifikatzaileen desberdintasunak

Klaseen arteko erlazioa adierazteko:

 Herentzia. Super klase eta azpi klaseen arteko erlazioa.

 Klaseen arteko erlazioa adierazten du.

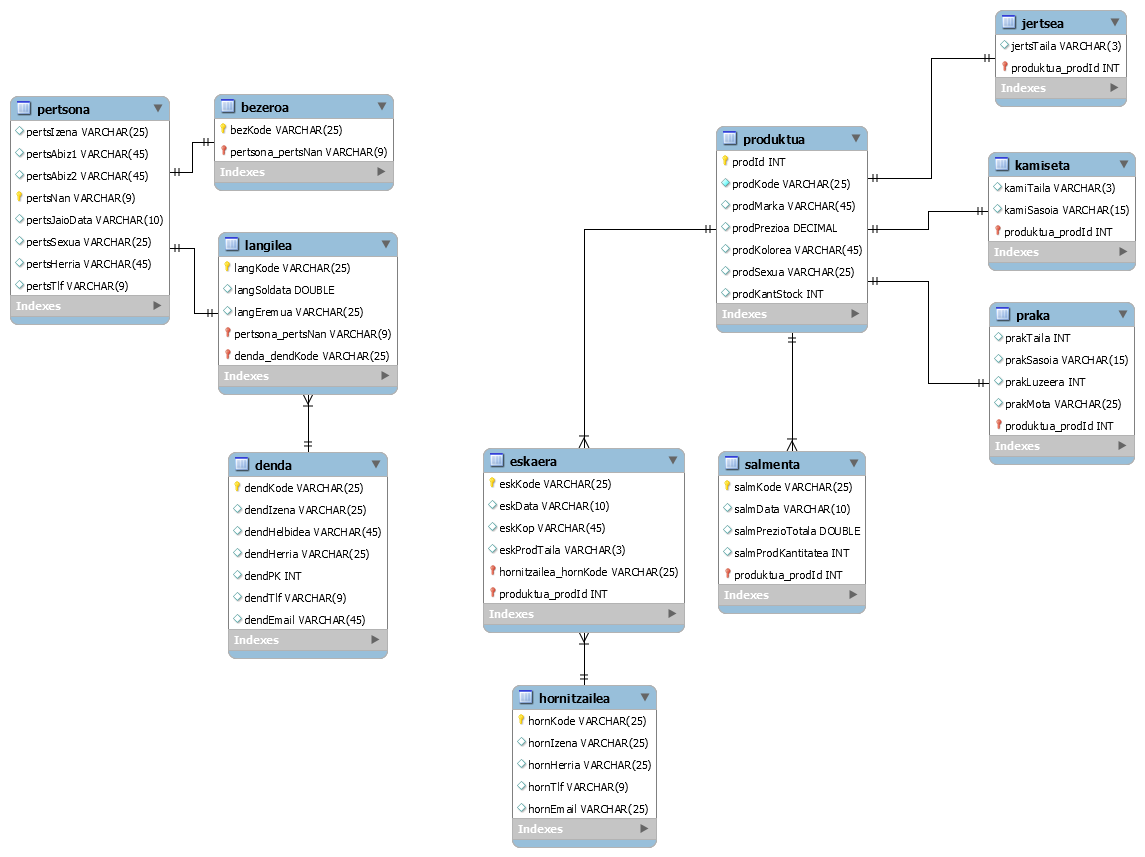
Ondorengo irudian ikusi daiteke klase diagrama, bista eta Controllerrekin batera. Kolore urdina duten klaseak, bistak dira eta berde kolorekoak berriz, Controllerrak.



4. Irudia: Klase diagrama - MVC

## DDBB-aren diagrama

Aurreko puntuko klase diagrama kontuan hartuta, datu baseen taulak eta beraien arteko erlazioa sortu dira.



5. Irudia: DDBB diagrama

Ondorengo taulan, diagraman agertzen diren gezien azalpena ikusi daiteke, bakoitzaren erlazioa jakinez.

|  |  |
| --- | --- |
| **GEZIA** | **ERLAZIOA** |
|  | 1:1 erlazioa |
|  | 1:N erlazioa. |

2. Taula: DDBB-ko erlazioak

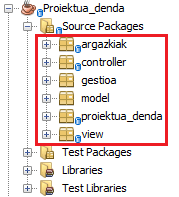
# **PROIEKTUAREN EGITURA**

Aurreko puntuan ikusitako klase diagrama kontuan hartuta, klaseak sortu dira. Atal honetan, klaseak, atributuak, metodoak… zeintzuk diren eta nola dauden sortuta azaltzen da.

## Klaseak

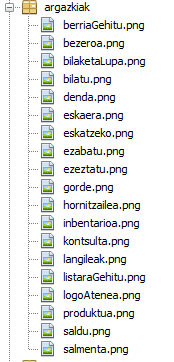
Honako hauek dira proiektu honetan sortutako klaseak.

Sei pakete sortu dira, ***argazkiak***, ***controller***, ***gestioa, model,*** ***proiektua\_denda eta view***, ordena mantentzeko.



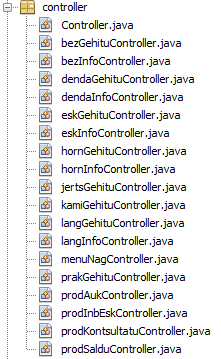
6. Irudia: Paketeak

1. ***Argazkiak*** paketean, honako irudi hauek daude:



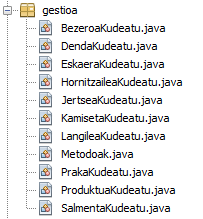
7. Irudia: Argazkiak paketea

1. ***Controller*** paketean, bistak kontrolatzeko fitxategiak daude. Honek, bistak egingo duen guztia kontrolatuko du, bai diseinu aldetik, baita funtzionalitate aldetik ere.



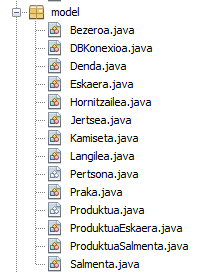
8. Irudia: Controller paketea

1. ***Gestioa*** paketean berriz, beste paketeko klaseak kudeatzeko sortu diren klaseak daude, non objektu bakoitza kudeatzeko metodo estatikoak dauden.



9. Irudia: Gestioa paketea

1. ***Model*** paketean, modeloak daude. Baita, datu basera konektatu eta deskonektatzeko klasea ere (DBKonexioa klasea).



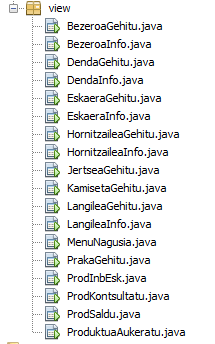
10. Irudia: Model paketea

1. ***proiektua\_denda*** paketen, proiektuko klasea dago, hau da, main-a.



11. Irudia: Main paketea

1. ***view*** paketean, aplikazioak dituen bista guztiak daude.



12. Irudia: View paketea

Kolore desberdinez agertzen direnak ( adibidez), **klase abstraktuak** dira. Klase hauen instantzia berririk ezingo da sortu. Hau lortzeko, kodean “*abstract*” hitza erabili behar da. Hona hemen adibide bat.



13. Irudia: Klase abstraktuak

## Herentzia

Herentzia, super klase bat sortzea da, honen azpi klaseak, bere atributu bai metodoak heredatzeko asmoz. Horrela, azpi klaseetan, ez lirateke egongo atributu eta metodoak errepikatuta.

Esan bezala, klase bat heredatzerakoan, klase horrek duen atributu bai metodoak heredatzen dira. Hau da, superklaseak dituen atributu eta metodoak, azpi klaseek ere izango dute.

Hau lortzeko, kodean “*extends*” hitza erabili behar da. Hona hemen adibide bat.



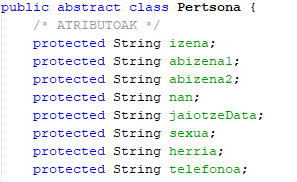
14. Irudia: Herentzia

Adibide horretan, Kamiseta, Produktuak klasearen azpi klasea da. Produktuak dituen atributu eta metodoak heredatuko ditu.

Honekin, kodea klaseetan behin eta berriz errepikatzea saihestuko dugu, amankomunean daukaten atributu eta metodoak aita izango den klasean jarriz.

Proiektu honetan, herentzia behin baino gehiagotan aplikatzen da. Honen adibide bat, pertsona, langile eta bezeroen artekoa da.

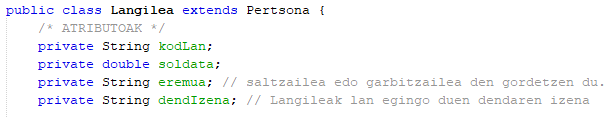
Langile eta bezero guztiek, izena, abizenak, NAN-a, jaiotze data… izango dute. Atributu guzti horiek behin eta berriz errepikatu ordez, **Pertsona** izeneko klase bat sortu eta atributu honek bertan jarri daitezke, ondorengo argazkian ikusten den bezala.



15. Irudia: Adibidea - Herentzia Pertsona

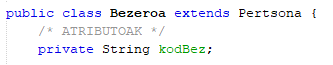
Pertsona klaseak atributu guzti horiek izanik, eta Langilea eta Bezeroa klaseetan herentzia aplikatuz, hauetan desberdinak direnak bakarrik jarri beharko lirateke. Hona hemen bi adibideak.

1. Langilea klasean, kodLan (langilearen kodea), soldata eta eremua daude.



16. Irudia: Adibidea - Herentzia Langilea

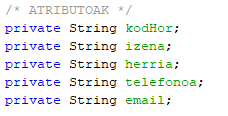
1. Bezeroa klasean, berriz, kodBez (bezeroaren kodea).



17. Irudia: Adibidea - Herentzia Bezeroa

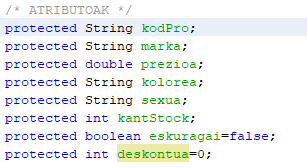
## Atributuak

Klase bakoitzak, beharrezkoak dituen atributuak ditu. Atributuak definitzerakoan, zein modifikatzaile izango duen (public, private, protected edo default/package), zein datu mota izango den (int, String, double…) eta edukiko duen izena jarri behar dira.



18. Irudia: Atributuak

Atributu guztiak, pribatuak (private) izango dira, salbuespen batekin. Herentzia aplikatuta dauden azpiklaseetan, babestuak (protected) izango dira.



19. Irudia: Atributuak (herentziarekin)

## Eraikitzaileak

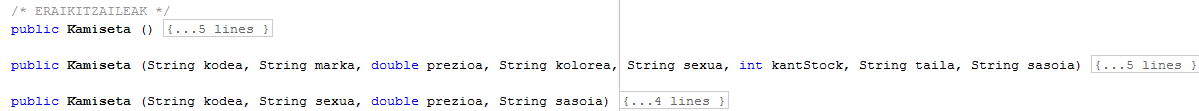
Eraikitzaileak, objektu berri bat sortzeko balio dute. Hauen izena, klasearen berdina izan behar da eta ez dute ezer bueltatzen.

Objektu baten instantzia berri bat egiten denean, eraikitzaileari deitzen zaio. Instantzia berri bat egiteko, *“new”* hitza erabiltzen da.



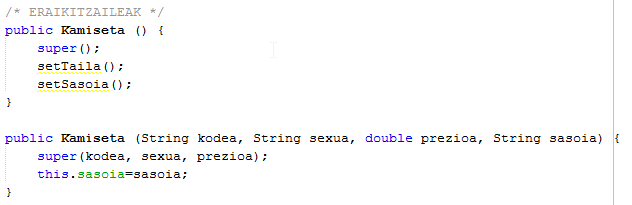
20. Irudia: Adibidea – Instantzia berria sortu

Klase berdin bakoitzean, eraikitzaileen gainkarga egon daiteke. Guztiek izen berdina izan behar dute, baina, parametro desberdinak. Hau da, parametro kopurua eta ordena desberdina izan behar da.



21. Irudia: Eraikitzaileetan gainkarga

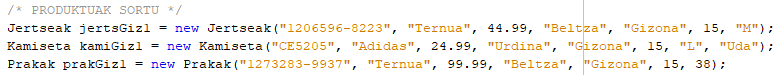
Eraikitzaileen adibide bat, Kamiseta klaseko hau izan daiteke. Hemen, aurretik esan bezala, gainkarga daukagu.



22. Irudia: Adibidea - Eraikitzaileak

***OHARRA: super();*** edo ***super(arg1, arg2…);*** erabiltzen dira, herentzia dutenen kasuan, super klaseko eraikitzaileari deitzeko.

Objektu berri bat sortzeko, instantzia berri bat egiten da, eraikitzaile bati deituz. Parametro kopuru eta ordenaren arabera, eraikitzaile bati edo beste bati deitzen zaio.



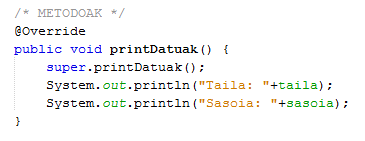
23. Irudia: Adibidea - Objektu berria

## Metodoak

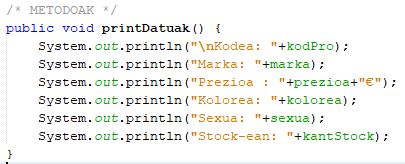
Metodoek, definituta daukaten zeregin jakin bat egiten dute eta izenaren bidez deitzen dira.

Herentziadun klaseetan, azpi klaseetan, ***super.<MetodoIzena>()*** erabili beharko da superrari deitu eta datu guztiak hartzeko, bai azpi klaseak duena eta baita superklaseak duenak ere.

Ondorengo irudian ikusi daiteke adibide argi bat. Lehenengo irudiko metodoa, **Kamiseta** klasean definituta dago. Honek, **Produktua** klaseko atributuak hartzen dituenez, *super.printDatuak()* jarri behar da. *super.printDatuak()*, superklasearen *printDatuak()* metodoari deituko dio (ikusi gezi gorria) eta bertakoa inprimatu, bestearekin batera.

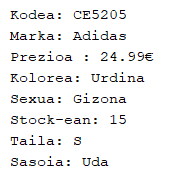


23. Irudia: Adibidea - printDatuak() metodoa (Kamiseta)



25. Irudia: Adibidea - printDatuak() metodoa (Produktua)

Metodo horrek, honako hau bueltatuko luke:



25. Irudia: Adibidea - printDatuak()

**printDatuak()** metodoa klase guztietan sortu da. Honek, klaseko atributu guztien datuak erakusten ditu.

**printDatuak()** metodoaz gain, beste hainbat sortu dira. Ondorengo irudian ikusi daitezke adibide batzuk. Metodo hauek, Kamiseta klasean definituta daudenak dira. Hauek, ondoren, Kamiseta kudeatzeko sortuta dauden beste klasean erabiliko dira, datu zehatz batzuk bistaratzeko.

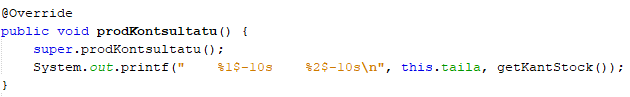


27. Irudia: Adibidea - Metodoak

Datuak pantailan inprimitzerakoan, zutabetan edo errenkadetan ikusi nahi denaren arabera, informazioa nahasi zamar agertu daiteke.

Datuak errenkadetan bistaratzeko, **System.out.printf()** metodoa erabili da, datuen arteko espazioa kontrolatzeko.

Ondoko irudian, horren adibide bat.



28. Irudia: Adibidea - prodKontsultatu() metodoa (Kamiseta)

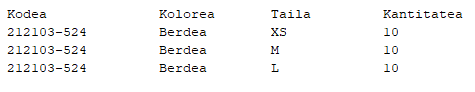


29. Irudia: Adibidea - prodKontsultatu() metodoa (Produktua)

**%1$-15s** bezalako parametroak erabiltzen dira. “%1$” zatiak, zenbatgarren aldagaiari egiten dion erreferentzia markatzen du. “15s” zatiak berriz, datuen arteko hutsune kopurua markatzen du.



Espazioak kontrolatuta, ondorengo irudian ikusten den emaitza lortuko da.



30. Irudia: Adibidea- prodKontsultatu()

### ***Metodo estatikoak***

Kamisetak, prakak, jertseak, bezeroak… hauetako klase kudeatzeko, beste klase bat sortu da, **KamisetaKudeatu.java** bezalakoak. Kudeatzeko klase hauetako bakoitzean, fitxategi bat sortu da objektuak bertan gordetzeko eta metodo estatikoak, objektuak kudeatzeko. Kudeaketa hori, fitxategien gainean egiten da, aurrerago ikusiko den bezala.

Metodo estatikok definitzeko, *“static”* hitza erabiltzen da.



31. Irudia: Adibidea - Metodo estatiko bat definitu

Metodo hauek, instantzia berririk egin gabe erabili daitezke.



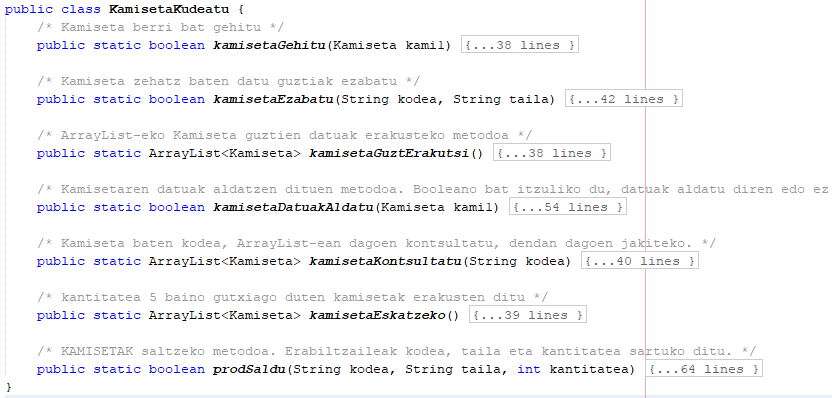
32. Irudia: Metodo estatikoak erabiltzen

Irudian ikusten den bezala, kamisetaGehitu(arg) metodo estatikoa da, eta metodo hori erabiltzeko, ez dago instantziatu beharrik. Nahikoa da metodoa definituta dagoen klasearen izena aurretik jartzea.

**<Klase\_izena>.<metodo\_estatiko\_izena>;**

Metodo estatikoen adibide batzuk, hondoko irudian ikusten da. Aurretik komentatu bezala, objektuak gordetzeko fitxategi bat sortu da eta objektu horiek kudeatzeko (gehitu, ezabatu, erakutsi…) metodo estatikoak.

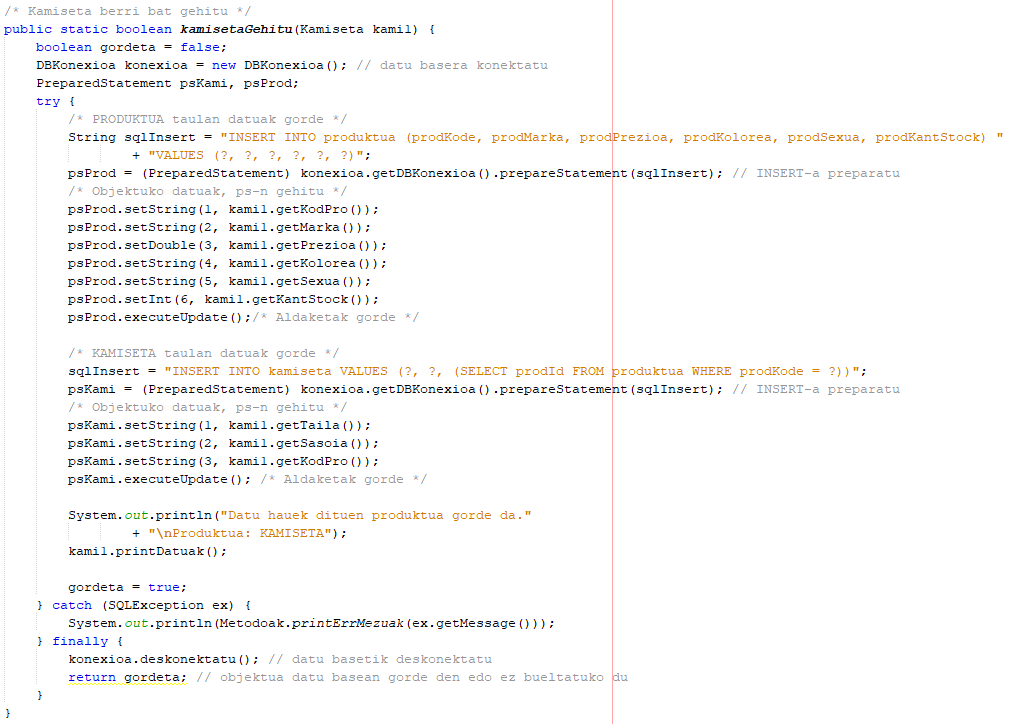
Fitxategiak, Objektuak karpeta barruan sortzen dira eta **kamiseta.obj** bezalakoak izango dira.



33. Irudia: Metodo estatikoak

Metodo estatiko baten adibidea ondoko irudian ikusi daiteke. Metodo hau, objektu bat (kamiseta kasu honetan) datu basera gehitzeko sortu da.

1. Objektuaren instantzia berria sortzen da, erabiltzaileak bistan sartu dituen datuekin.
2. Sortutako instantzia berri hori datu basean erregistratzen da. Honetarako, kasu honetan, INSERT bat egiten da datu baseko kamiseta kamiseta eta produktua tauletan (bi tauletan egin behar da, herentzia dagoelako eta produktua taulan, kamisetaren datuak erregistratzen direlako).



34. Irudia: Adibidea - Metodo estatikoa

Sortu diren metodo estatikoak honako hauek dira:

*Metodoak klasea*()

* **printUrdinez(textua: String): String 🡪** Textua urdinez bueltatzen duen metodoa. Mezuak urdinez kontsolan inprimitzeko (db-ra konektatzen denean adibidez).
* **printErrMezuak(textua: String): String 🡪** Textua larroxaz bueltatzen duen metodoa. Errore mezuak kontsolan inprimitzeko.
* **emailBalidazioa(mail: String): boolean 🡪** Email-a ondo estrukturatuta dagoen konprobatzen duen metodoa. Expresio erregularrak erabiltzen dira (adibidea@adibidea.com)
* **nanBalidazioa(nan: String): boolean 🡪** NAN zenbakia egokia den edo ez konprobatzen duen metodoa.
* **tlfBalidazioa(tlf: String): boolean 🡪** Telefono zenbakia egokia den edo ez konprobatzen duen metodoa.
* **kodeakAldatuEtaGorde(hasiera: String): String 🡪** kodeak.txt fitxategian gordetako kodeak (bezero, langile…) aldatzen dituen metodoa.
* **dataGorde(data: String): String 🡪** data uuuu/hh/ee formatura bihurtzen duen metodoa.
* **dataErakutsi(s: String): Date 🡪** string bat jasotzen du parametro bezala eta Date formatura bihurtzen du.

*DendaKudeatu klasea*()

* **dendaGehitu(dend1: Denda): boolean 🡪** denda berri bat gehitu.
* **dendaEzabatu(kodea: String): boolean 🡪** Denda bat ezabatu, kodearen arabera.
* **datuakErakutsi(): ArrayList<Denda>** 🡪 Dendaren datuak erakutsi.
* **dendaDatuakAldatu(dend1: Denda): boolean 🡪** Dendaren datuak aldatu.

*BezeroaKudeatu klasea* ()

* **bezeroaGehitu(bez1: Bezeroa): boolean** 🡪 bezero berri bat gehitu.
* **bezeroaEzabatu(nan: String): boolean** 🡪 bezero bat ezabatu, NAN-aren arabera.
* **bezeroGuztiakErakutsi(): ArrayList<Bezeroa>** 🡪 erregistratutako bezero guztiak erakutsi.
* **bezeroaDatuakAldatu(bez1: Bezeroa): boolean** 🡪 bezero baten datuak aldatu.

*LangileaKudeatu klasea* ()

* **langileEremuaKontrolatu(): String[]** 🡪 langileen lan eremua kontrolatzeko metodoa (saltzailea edo garbitzailea izan daitezke).
* **langileaGehitu(lang1: Langilea): boolean** 🡪 langile berri bat gehitu.
* **langileaEzabatu(nan: String): boolean** 🡪 langile bat ezabatu, NAN-aren arabera.
* **langileGuztiakErakutsi(): ArrayList<Langilea>** 🡪 erregistratutako langile guztiak erakutsi.
* **langileaDatuakAldatu(lang1: Langilea): boolean** 🡪 langile baten datuak aldatu.

*ProduktuaKudeatu klasea* ()

* **tailaKontrolatu(): String[]** eta **tailaKontrolatu(taila: int): String[]** 🡪 produktuen taila kontrolatzeko metodoa. Bistetako comboBox-ak kargatzeko erabiltzen dira.
* **sexuaKontrolatu(): String[] 🡪** produktuen edo pertsonan sexua (emakumea edo gizona) kontrolatzeko metodoa. Bistetako comboBox-ak kargatzeko erabiltzen dira.
* **sasoiaKontrolatu(): String[] 🡪** produktuen sasoia (udaberria, uda, udazkena edo negua) kontrolatzeko metodoa. Bistetako comboBox-ak kargatzeko erabiltzen dira.
* **motaKontrolatu(): String[] 🡪** produktuen mota kontrolatzeko metodoa. Bistetako comboBox-ak kargatzeko erabiltzen dira.

*JertseaKudeatu klasea* ()

* **jertseaGehitu(jerts1: Jertsea): boolean** 🡪 jertse berri bat gehitu.
* **jertseaEzabatu(kodea: String, taila: String): boolean** 🡪 jertse bat ezabatu, kodea eta tailaren arabera.
* **jertsGuztErakutsi(): ArrayList<Jertsea>** 🡪 erregistratutako jertse guztiak erakutsi.
* **jertseaDatuakAldatu(jerts1: Jertsea): boolean 🡪** jertsearen datuak aldatu.
* **jertseaKontsultatu(kodea: String): ArrayList<Jertsea>** 🡪 jertsea, dendan dagoen kontsultatu, kodearen arabera.
* **jertseaEskatzeko(): ArrayList<Jertsea>** 🡪 5 baino gutxiago dauden jertseak erakutsi.
* **prodSaldu(kodea: String, taila: String , kantitatea: int): boolean** 🡪 produktuaren salmenta. Erabiltzaileak, kodea, erosi nahi duen kantitatea eta taila sartu beharko ditu.

*KamisetaKudeatu klasea* ()

* **kamisetaGehitu(kami1: Kamiseta): boolean** 🡪 kamiseta berria gehitu.
* **kamisetaEzabatu(kodea: String): boolean** 🡪 kamiseta bat ezabatu, kodea eta tailaren arabera.
* **kamisetaGuztErakutsi(): ArrayList<Kamiseta>** 🡪 erregistratutako kamiseta guztiak erakutsi.
* **kamisetaDatuakAldatu(kami1: Kamiseta): boolean 🡪** kamisetaren datuak aldatu.
* **kamisetaKontsultatu(kodea: String): ArrayList<Kamiseta>** 🡪 kamiseta, dendan dagoen kontsultatu, kodearen arabera.
* **kamisetaEskatzeko(): ArrayList<Kamiseta>** 🡪 5 baino gutxiago dauden kamisetak erakutsi.
* **prodSaldu(kodea: String, taila: String, kantitatea: int): boolean** 🡪 produktuaren salmenta. Erabiltzaileak, kodea, taila eta erosi nahi duen kantitatea sartu beharko ditu.

*PrakaKudeatu klasea* ()

* **prakaGehitu(prak1: Praka): boolean** 🡪 praka berri bat gehitu.
* **prakaEzabatu(kodea: String, taila: int): boolean** 🡪 praka bat ezabatu, kodea eta tailaren arabera.
* **prakaGuztErakutsi(): ArrayList<Praka>** 🡪 erregistratuta dauden praka guztiak erakutsi.
* **prakaDatuakAldatu(prak1: Praka): boolean 🡪** prakaren datuak aldatu.
* **prakaKontsultatu(kodea: String): ArrayList<Praka>** 🡪 praka, dendan dagoen kontsultatu, kodearen arabera.
* **prakaEskatzeko(): ArrayList<Praka>** 🡪 5 baino gutxiago dauden prakak erakutsi.
* **prodSaldu(kodea: String, taila: int, kantitatea: int): boolean** 🡪 produktuaren salmenta. Erabiltzaileak, kodea, taila eta erosi nahi duen kantitatea sartu beharko ditu.

*HornitzaileaKudeatu klasea* ()

* **hornitzaileaGehitu(horn1: Hornitzailea): boolean** 🡪 hornitzaile berri bat gehitu.
* **hornitzaileaEzabatu(kodea: String): boolean** 🡪 hornitzaile bat ezabatu, kodearen arabera.
* **hornitzaileGuztiakErakutsi(): ArrayList<Hornitzailea>** 🡪 erregistratuta dauden hornitzaile guztiak erakutsi.
* **hornitzaileaDatuakAldatu(horn1: Hornitzailea): boolean 🡪** Hornitzailearen datuak aldatu.

*EskaeraKudeatu klasea* ()

* **eskaeraGehitu(esk1: Eskaera): void** 🡪 eskaera berri bat gehitu.
* **eskaeraEzabatu(kodea: String): void 🡪** eskaera bat ezabatu, kodearen arabera.
* **eskaeraGuztiakErakutsi(): ArrayList<Eskaera>** 🡪 erregistratuta dauden eskaera guztiak erakutsi.
* **eskaeraDatuakAldatu(esk1: Eskaera): boolean 🡪** Eskaeraren datuak aldatu.

*SalmentaKudeatu klasea* ()

* **salmentaGehitu(salm1: Salmenta): boolean** 🡪 salmenta berri bat gehitu.
* **salmentaGuztiakErakutsi(): ArrayList<Salmenta>** 🡪 erregistratuta dauden salmenta guztiak erakutsi.

## Getter & Setter

Aurretik aipatutako metodoez gain, get eta set metodoak ere sortu dira klase guztietan.

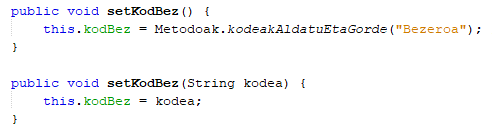
* **Getter**-ak, atributu batek duen balioa lortu eta erabiltzeko balio du.
* **Setter**-ak, atributuei balio bat emateko balio dute. Metodo honek, ez du ezer bueltatzen.

Atributu bakoitzeko, set bat eta get bat sortu dira klase bakoitzean. Get-ek, atributuaren balioa bueltatuko du eta set-ek berriz, parametro bezela jartzen den balioa gordeko da.



35. Irudia: Getter & Setter

**OHARRA**: Momentuz, set eta get metodoetan dagoen gainkarga bakarra, objektuen kodeetan da. Adibidez, ondorengo irudian ikusten den bezala, Bezeroa klasean.



36. Irudia: Adibidea - Set gainkarga

Metodo guzti hauek, antzerakoak izango dira.

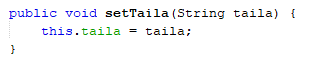
1. Get metodoak:



37. Irudia: Adibidea - Getter

1. Set metodoak:

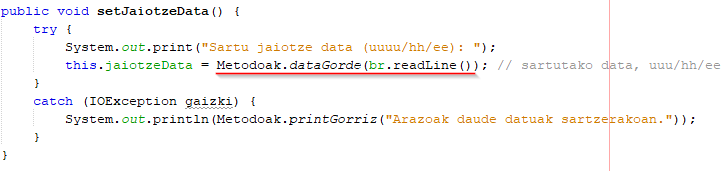
Datua parametro bezala pasatuko zaio, balioa objektuan gorde dezan.

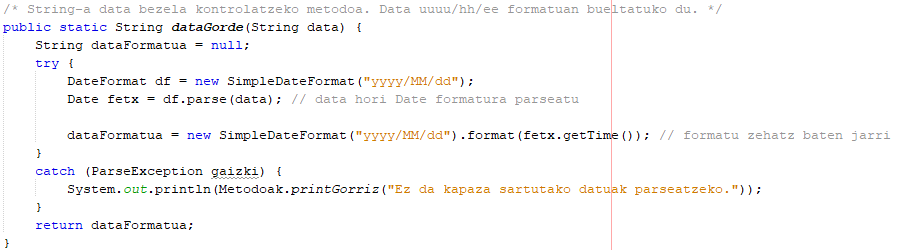


38. Irudia: Adibidea - Setter

Datak jasotzerako orduan, honako forma honetan egin da. Metodo bat sortu da, data parseatzeko eta *uuuu/hh/ee* formatuan gorde ahal izateko. Metodo honek, **dataGorde** izena dauka.

Ondorengo adibidean ikusten den bezala, data jasotzerako orduan (**setJaiotzeData** metodoan), **dataGorde** metodoari deitzen dio, honek *uuuu/hh/ee* formatuan parseatu eta gordetzeko.





39. Irudia: Adibidea - Getter & Setter (Date mota)

**dataGorde** metodoan, data jasotzeko formatua definitzen da lehenengo (urtea/hilabetea/eguna) eta ondoren, erabiltzaileak sartu duen string-a ***df.parse(<data>)*** erabilita, datua parseatuko du.

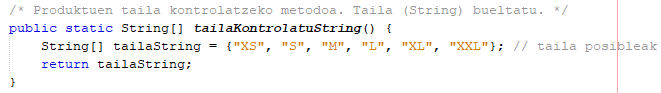
## Array-ak

Array-etan, mota berdineko datuak gordetzen dira eta luzeera aurretik zehazten da.

***<arrayMota>[] <arrayIzena> = {<definitutakoDatuak>…}***

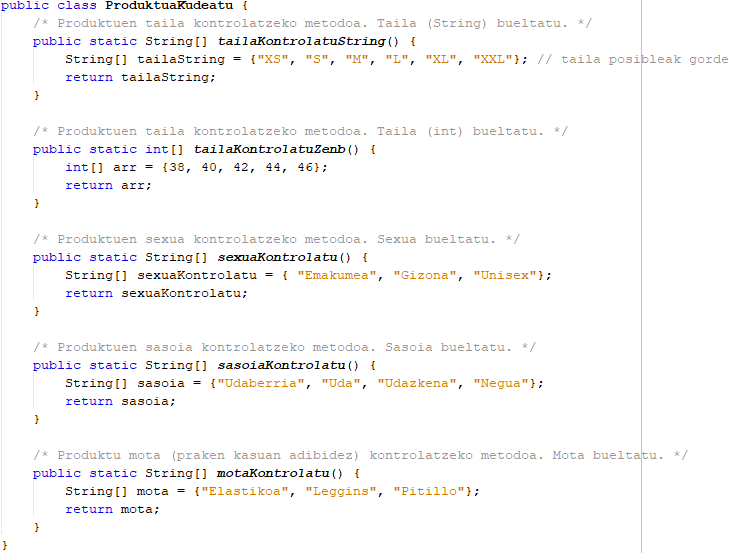
Hiru metodo sortu dira datu zehatz batzuk kontrolatzeko, taila eta sexua hain zuzen. Metodo estatiko hauetako bakoitzean, Array bat definitzen da, eta bertan, datuak zehaztuta egongo dira.

Adibidez, erabiltzaileak produktu baten taila (String motakoa) aukeratzerakoan kontrol bat eramateko.



40. Irudia: Adibidea - Array

Array-ak, honako kasuetan erabili dira. Guztietan, datuak kontrolatzeko.



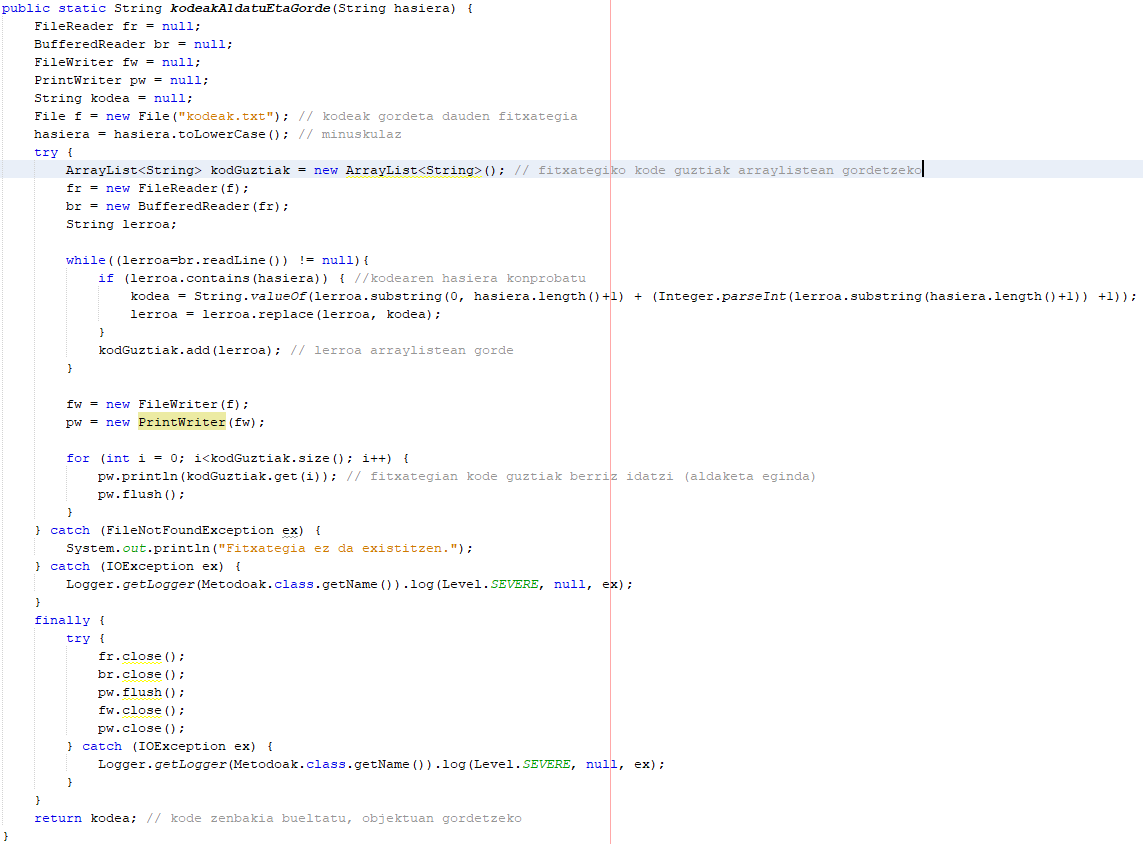
41. Irudia: Array-ak

## Fitxategiak

Fitxategietan, mota desberdinetako datuak gorde daitezke.

Testu fitxategi batean, kodeak.txt izeneko fitxategian, bezero, langile, hornitzaile, eskaera eta salmenta berrien kodeak gordetzen dira. Hau lortzeko, metodo bat sortu da, metodo honi deitzean, automatikoki erregistratzeko.

Ondorengo irudian ikusi daiteke hauen kontrola eramateko adibidea. Kontrol hori eramateko, metodo estatiko bat sortu da.



42. Irudia: kodeakAldatuEtagorde(String hasiera): String metodoa

Fitxategian dagoena irakurtzen du eta bere lerroa bilatzen duenean, kodearen azkenengo lau karaktereak hartzen ditu, hauek zenbaki bihurtu eta +1 egin. Ondoren, berriz eraikitzen da String-a eta fitxategian gorde.

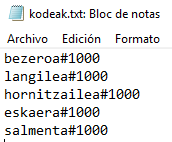
Ukitzen ez diren kodeak ez galtzeko, kodeak aldatzen hasi aurretik, arraylist batean gordetzen dira.

Metodo estatiko hau, bezero, langile, eskaera, hornitzaile eta salmenta klaseetan erabiliko da, klase bakoitzeko kodearen set metodoan hain zuzen ere.



43. Irudia: Adibidea - Getter & Setter (kodea automatikoki gehitzen)

Hauen defektuzko balioak honako hauek dira:

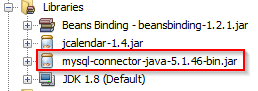


44. Irudia: kodeak.txt fitxategiko balioak

## Datu basea

Aplikazioko datuak, datu base zehatz batean gordeko dira. Horretarako, aplikazioa gai izan behar da datu basera konektatzeko.

Hau lortzeko, mysql-connector-java-5.1.46-bin.jar liburutegia gehitu da.



45. Irudia: mysql-connector-java-5.1.46-bin.jar

Ondorengo irudian, datu basera konektatu eta deskonektatzeko klasea ikusi daiteke.

Atributu bezala, datu basearen izena, url-a, erabiltzailea eta pasahitza definitzen dira, ondoren konektatzeko.

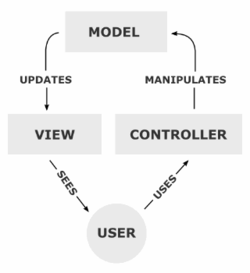


46. Irudia: Datu basera konexioa

Datuak gehitu, kontsultatu edo manipulatzen diren bakoitzean, aplikazioa, datu basera konektatuko da eta hau bukatzean, deskonektatu.

## Interfaze grafikoa

MVC (Modelo Vista Controlador) erabili da. Aplikazioko datuak, erabiltzailearen interfazea eta kontrola hiru osagai desberdinetan banatzen ditu.



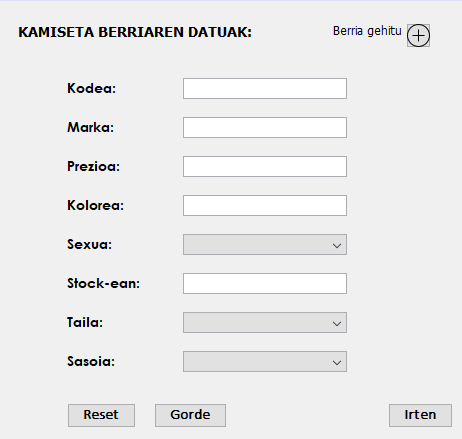
47. Irudia: MVC

### ***View - bistak***

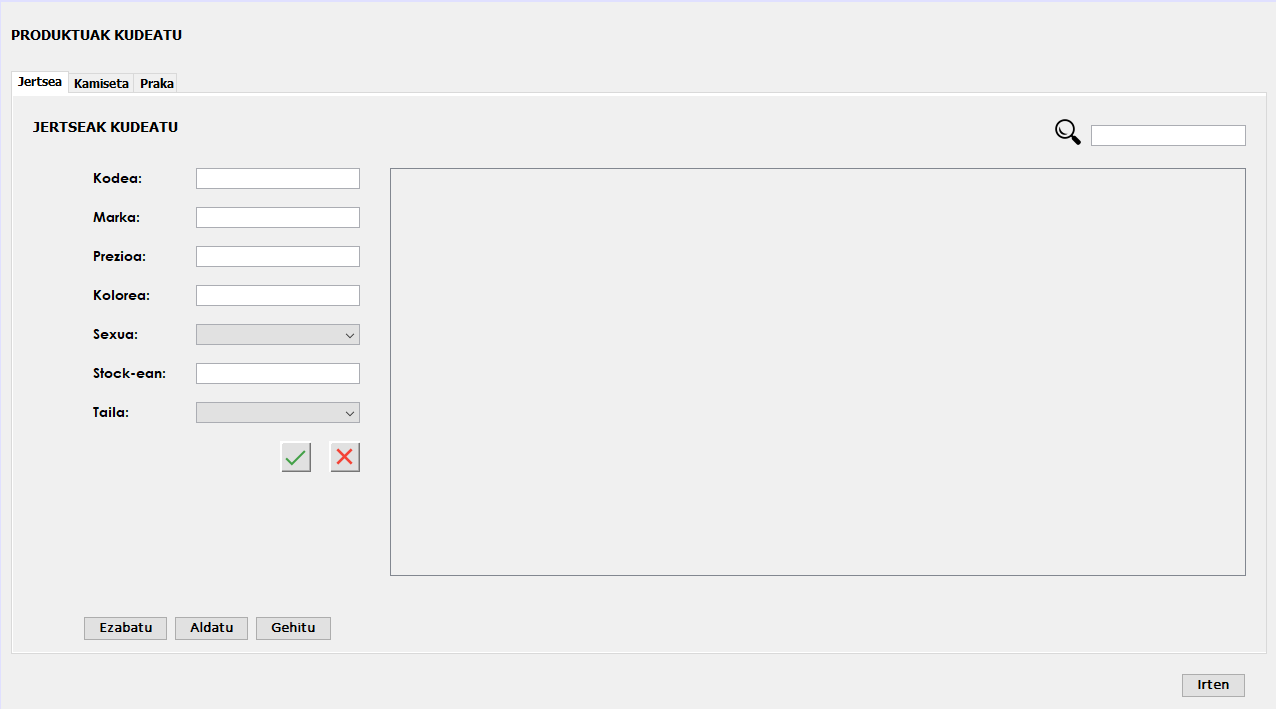
Bista bakoitzeko, JFrameForm klase bat sortu behar da. Bertan, nahi den osagaiak jarri behar dira.

Osagaiak, jLabel, jTextField, jButton, jComboBox… izango dira.

Ondorengo irudietan adibide batzuk ikusi daitezke. Ikusten den bezala, osagaiak definiturik bakarrik daude. Ez dago ez estilorik ez kolorerik definituta (hau Controller atalean egingo da).



48. Irudia: Adibidea - KamisetaGehitu bista

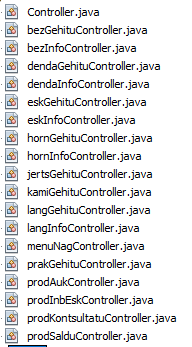


49. Irudia: Adibidea - ProdAukeratu bista

### ***Controller***

Controller-ak, aplikazioan gertatzen diren gertaerak kudeatuko ditu.

Controller bakarra egon daiteke edo bat baino gehiago. Kasu honetan, bista bakoitzeko bat sortu da.



50. Irudia: Controller

**Controller.java**: Bista guztietako osagaiek non entzungo duten definituta egongo da, hau da, osagaiak gertaera bat duenean, nora jo behar duen.

Controller-ean, bista bakoitzak izango duen portaera definitzen da. Hau da, botoi bat sakatzean, botoiaren gainetik pasatzean, zerbait idaztean… gertatuko dena definituta egongo da bertan.

Horretarako, jTextField, jComboBox, jButton… bakoitzari akzioak gehituko zaie (adibidez, ActionListener, MouseListener, FocusListener…). Hauek gehitzerakoan, controllerra zein den edo nork kontrolatu behar duen definitu behar da, irudian ikusten den bezala. Parentesien artean, controllerra jarri beharko da (kontrolatuko duen klase berdinean jartzen bada, ***this*** hitza erabili beharko da).



51. Irudia: Adibidea – Akzioak gehitzen

Ondorengo irudian, hobeto ikusi daiteke:



52. Irudia: Akzioak gehitzen

Controller bakoitzak zein metodo exekutatu behar duen jakin dezan, “Listener”-ak inplementatu behar dira, ondoren, erabili ahal izateko.



53. Irudia: Akzioak inplementatu Controller-ean



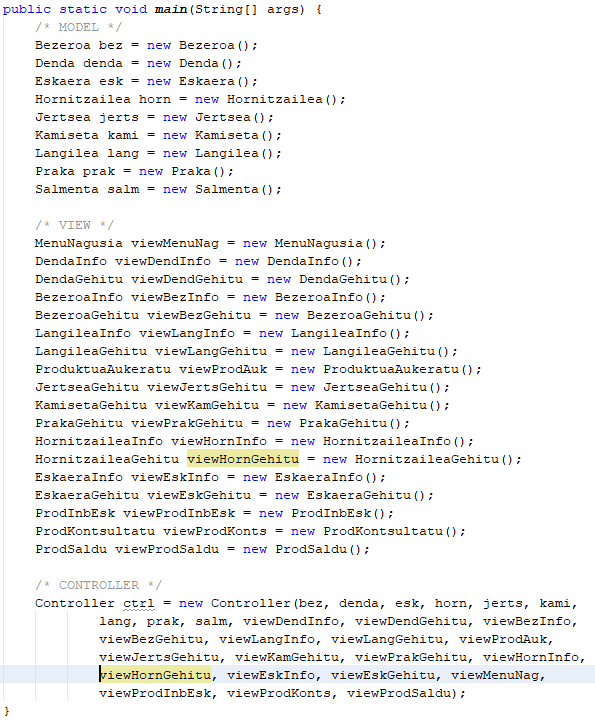
54. Irudia: Adibidea - KamiGehituController

### ***Main***

Aplikazioa exekutatu dadin, main-ean modelo eta bista guztien instantzia berriak sortu behar dira.

Adib: **Bezeroa bez = new Bezeroa();**

Behin instantziak sortuta daudela, Controllerreko eraikitzaileari pasatu behar dira.

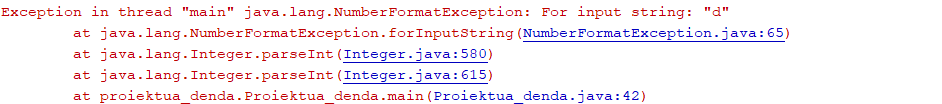


55. Irudia: Adibidea - Main

## Erroreen kontrola

Datuak jasotzerako orduan, errore desberdinak agertu daitezke. Erabiltzaileak datu mota egokia ez jartzea, programa datuak jaso edo bueltatzeko kapaza ez izatea…

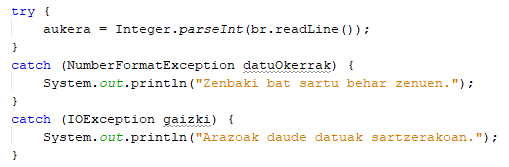
Errore hauek, ondorengo irudikoen antzerakoak izango dira.



56. Irudia: Adibidea - Errorea

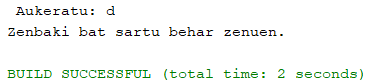
Errore hauek kontrolatzeko, try – cath erabili da. Hauek, errore bat ematen duenean, erabiltzaileak ulertuko ez duen linea gorriak ikusi beharrean, zein izan den arazoa esango dion mezu bat erakutsiko dute.

Hona hemen, erabili diren adibide batzuk:



57. Irudia: Adibidea - try/catch

Try/catch erabilita, honelako mezuak agertuko dira.



58. Irudia: Adibidea - try/catch mezuak

Proiektuan erabilitakoak hauek dira:

|  |  |
| --- | --- |
| NumberFormatException | Karaktere ez numerikoak sartzean ematen duten erroreak kontrolatzeko. |
| IOException | Sarrera/irteerako erroreak kontrolatzeko. |
| ParseException | String bat beste datu mota batera parseatu ezin denean ematen duen errorea kontrolatzeko. |
| SQLException | Datu basearen inguruko erroreak kontrolatzeko. |

3. Taula: Erroreen kontrola

Horrez gain, GUI-ko bistak kontrolatzeko eta erroreen kontrola eramateko, funtzio desberdinak sortu dira bistetan, erabiltzaileak sartuko dituen datuak baliozkotzeko asmoz.

Horren adibide bat, ondoko irudian ikusi daiteke.



59. Irudia: Adibidea – Erabiltzaileak sartuko dituen datuak baliozkotzen

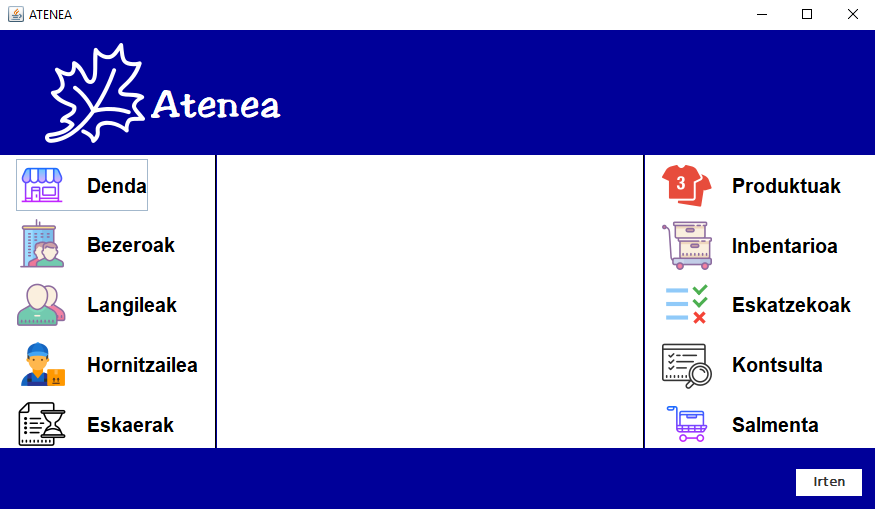
Honelako funtzioek, datuak sartuko dituen eremu bakoitza hutsik dagoen begiratzen du, baita sartutako datua zenbakia izan behar bada edo beste balidazio batzuk (email, telefonoa, NAN zenbakia…).

Datuak okerrak direnean edo eremuren bat hutsik geratzen denean eta erabiltzailea datu horiek gordetzen saiatzen denean, eremua gorriz margotuko da eta mezuren bat ere agertu daiteke.

# **APLIKAZIOA**

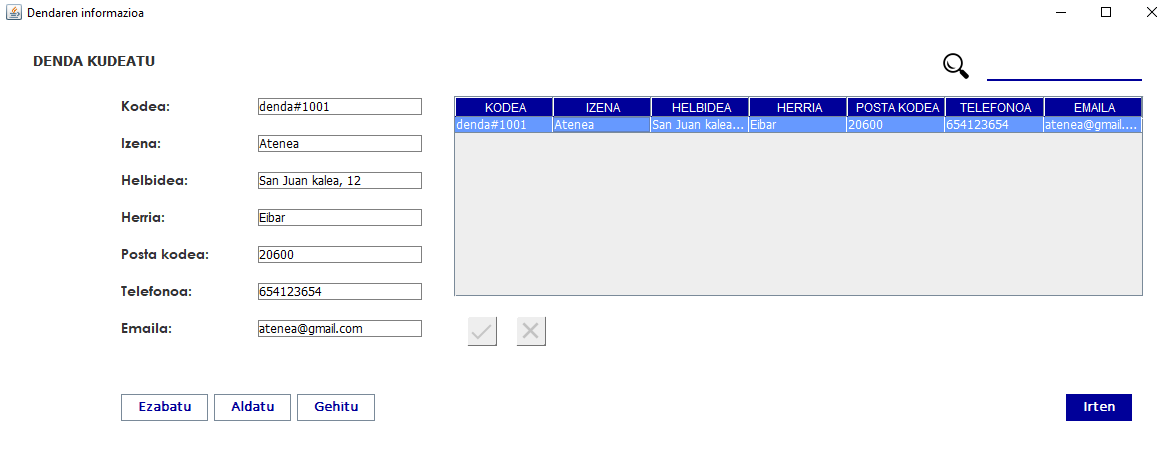
Objektuak sortuko dira eta erabiltzaileak datuak sartuko ditu. Datuak, formulario antzekoetan sartu beharko ditu eta hauek, fitxategietan gordeko dira, horrela, aplikazioa ixten denean, datu horiek mantendu egingo dira.

Menu nagusi bat agertuko da, aukera desberdinak emanez. Sakatzen den botoiaren arabera, gauza bat edo beste agertuko da.



60. Irudia: Main - Menu nagusia

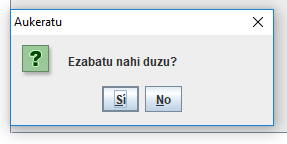
* Menu nagusian **DENDA** aukeratzen bada, denda kudeatzeko leiho bat agertuko da. Taulan, erregistratuta dauden denda guztiak agertuko dira eta hauetako lerro batean sakatzean, eskuineko eremuak beteko dira, lerro horretako datuekin.



61. Irudia: Aplikazioa – denda kudeatu

Leiho honetan, botoi desberdinak daude.

* **Irten** sakatzen bada, leiho hau itxi eta aurrekora joango da.
* **Ezabatu** sakatzen bada, aukeratutako lerro hori ezabatuko da (ez badago ezer aukeratuta, mezu bat agertuko da). Ezabatu ahal izateko, baieztatzeko mezu bat agertuko da.



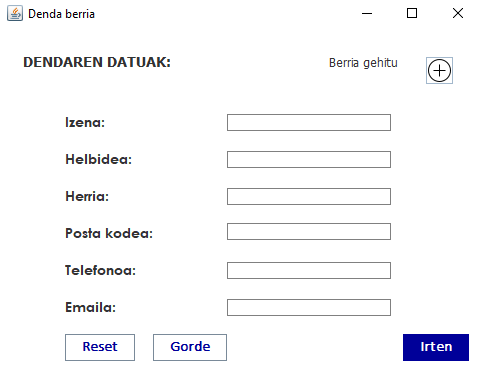
62. Irudia: Aplikazioa – denda ezabatu

* **Aldatu** sakatzen bada, aukeratutako lerro hori aldatzeko aukera ematen du (ez badago ezer aukeratuta, mezu bat agertuko da).

 Aldaketa gordetzeko sakatu behar den botoia.

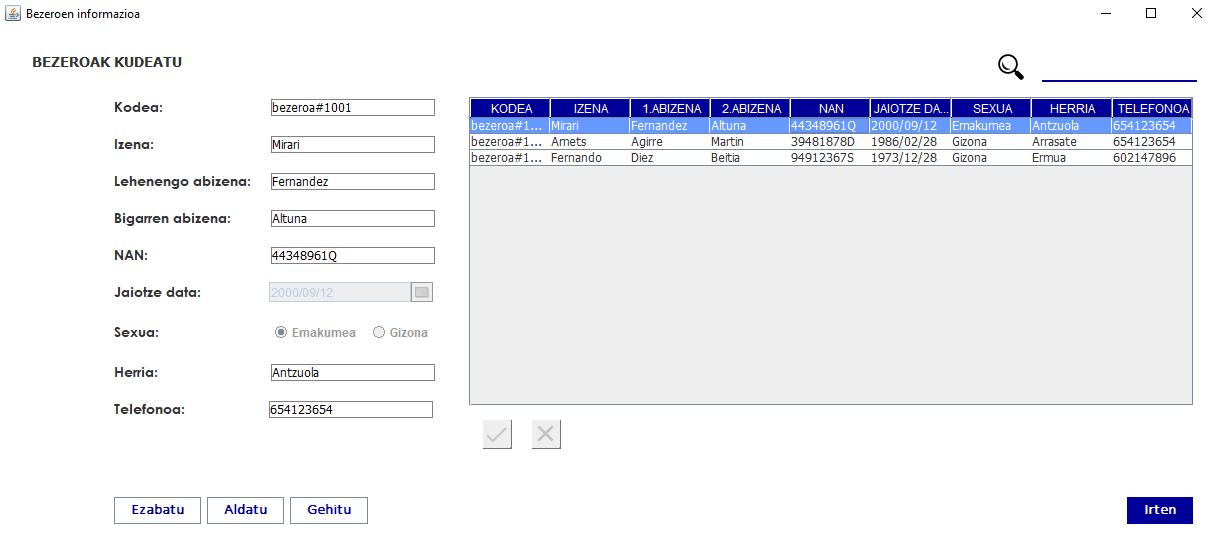
 Aldaketa ezeztatzeko/ezabatzeko sakatu behar den botoia.

* **Gehitu** sakatzen bada, denda berri bat gehitzeko leihoa zabalduko da eta bertan, erabiltzaileak datuak idatzi eta gorde egingo dituen.



63. Irudia: Aplikazioa – denda gehitu

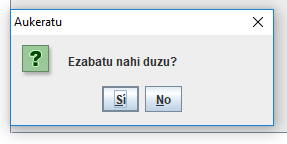
* Menu nagusian **BEZEROA** aukeratzen bada, bezeroa kudeatzeko leiho bat agertuko da. Taulan, erregistratuta dauden bezero guztiak agertuko dira eta hauetako lerro batean sakatzean, eskuineko eremuak beteko dira, lerro horretako datuekin.



64. Irudia: Aplikazioa – bezeroak kudeatu

Leiho honetan, botoi desberdinak daude.

* **Irten** sakatzen bada, leiho hau itxi eta aurrekora joango da.
* **Ezabatu** sakatzen bada, aukeratutako lerro hori ezabatuko da (ez badago ezer aukeratuta, mezu bat agertuko da). Ezabatu ahal izateko, baieztatzeko mezu bat agertuko da.



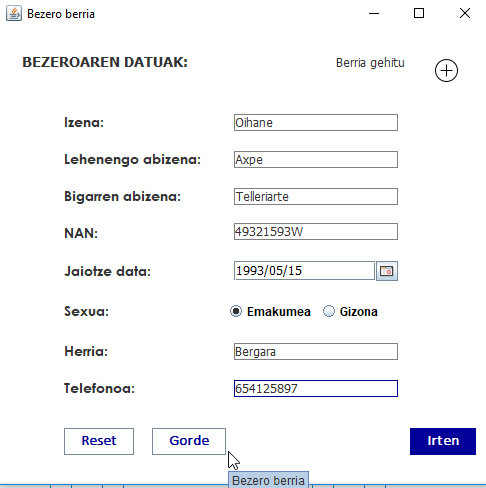
65. Irudia: Aplikazioa – denda ezabatu

* **Aldatu** sakatzen bada, aukeratutako lerro hori aldatzeko aukera ematen du (ez badago ezer aukeratuta, mezu bat agertuko da).

 Aldaketa gordetzeko sakatu behar den botoia.

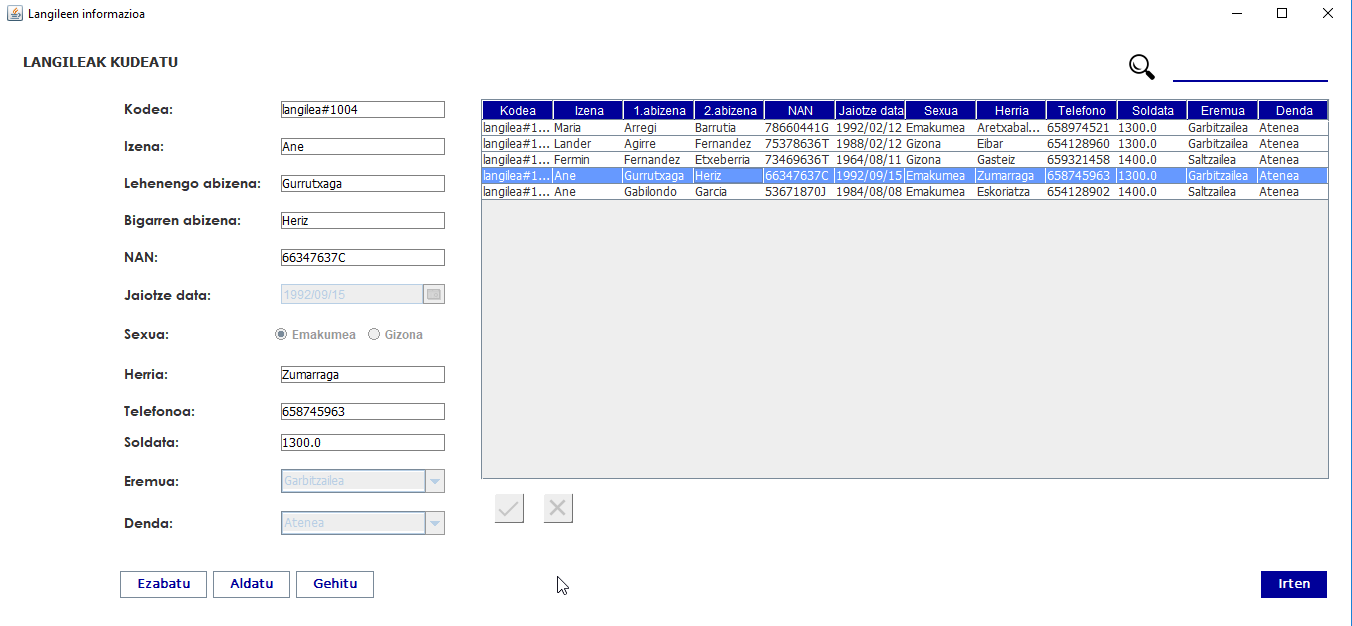
 Aldaketa ezeztatzeko/ezabatzeko sakatu behar den botoia.

* **Gehitu** sakatzen bada, bezero berri bat gehitzeko leihoa zabalduko da eta bertan, erabiltzaileak datuak idatzi eta gorde egingo dituen.



66. Irudia: Aplikazioa – bezeroa gehitu

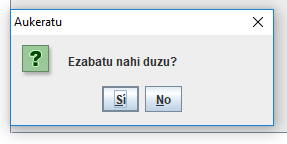
* Menu nagusian **LANGILEA** aukeratzen bada, langilea kudeatzeko leiho bat agertuko da. Taulan, erregistratuta dauden langile guztiak agertuko dira eta hauetako lerro batean sakatzean, eskuineko eremuak beteko dira, lerro horretako datuekin.



67. Irudia: Aplikazioa – langileak kudeatu

Leiho honetan, botoi desberdinak daude.

* **Irten** sakatzen bada, leiho hau itxi eta aurrekora joango da.
* **Ezabatu** sakatzen bada, aukeratutako lerro hori ezabatuko da (ez badago ezer aukeratuta, mezu bat agertuko da). Ezabatu ahal izateko, baieztatzeko mezu bat agertuko da.



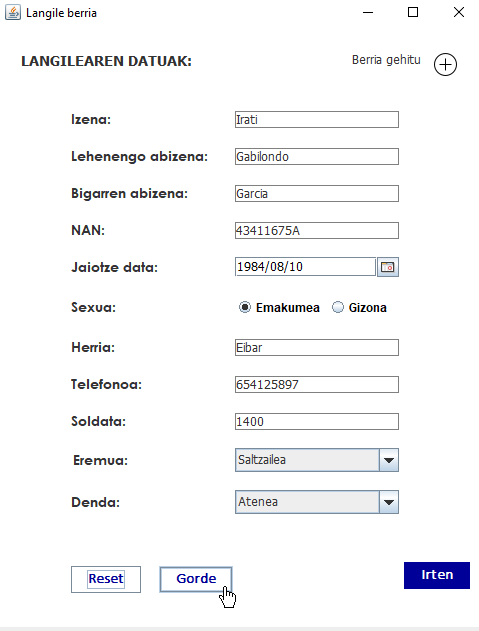
68. Irudia: Aplikazioa – denda ezabatu

* **Aldatu** sakatzen bada, aukeratutako lerro hori aldatzeko aukera ematen du (ez badago ezer aukeratuta, mezu bat agertuko da).

 Aldaketa gordetzeko sakatu behar den botoia.

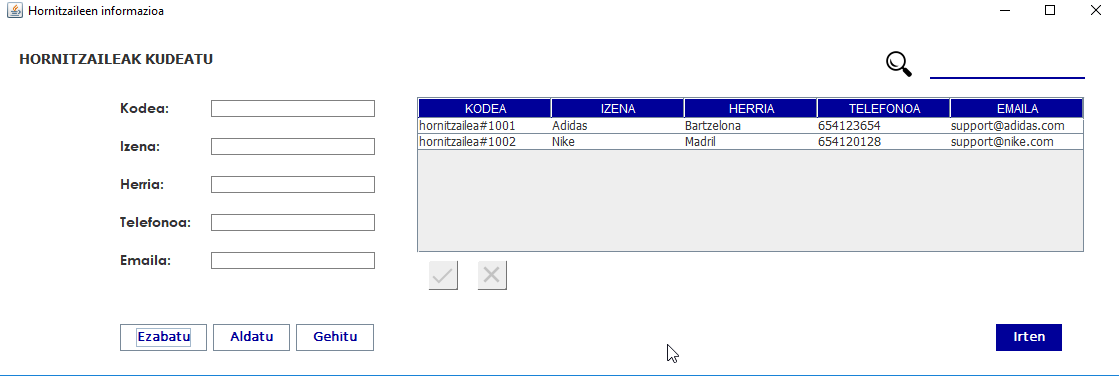
 Aldaketa ezeztatzeko/ezabatzeko sakatu behar den botoia.

* **Gehitu** sakatzen bada, langile berri bat gehitzeko leihoa zabalduko da eta bertan, erabiltzaileak datuak idatzi eta gorde egingo dituen.



69. Irudia: Aplikazioa – langilea gehitu

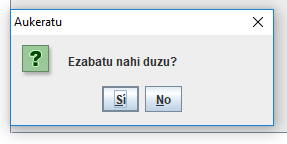
* Menu nagusian **HORNITZAILEA** aukeratzen bada, hornitzailea kudeatzeko leiho bat agertuko da. Taulan, erregistratuta dauden hornitzaile guztiak agertuko dira eta hauetako lerro batean sakatzean, eskuineko eremuak beteko dira, lerro horretako datuekin.



70. Irudia: Aplikazioa – hornitzaileak kudeatu

Leiho honetan, botoi desberdinak daude.

* **Irten** sakatzen bada, leiho hau itxi eta aurrekora joango da.
* **Ezabatu** sakatzen bada, aukeratutako lerro hori ezabatuko da (ez badago ezer aukeratuta, mezu bat agertuko da). Ezabatu ahal izateko, baieztatzeko mezu bat agertuko da.



71. Irudia: Aplikazioa – denda ezabatu

* **Aldatu** sakatzen bada, aukeratutako lerro hori aldatzeko aukera ematen du (ez badago ezer aukeratuta, mezu bat agertuko da).

 Aldaketa gordetzeko sakatu behar den botoia.

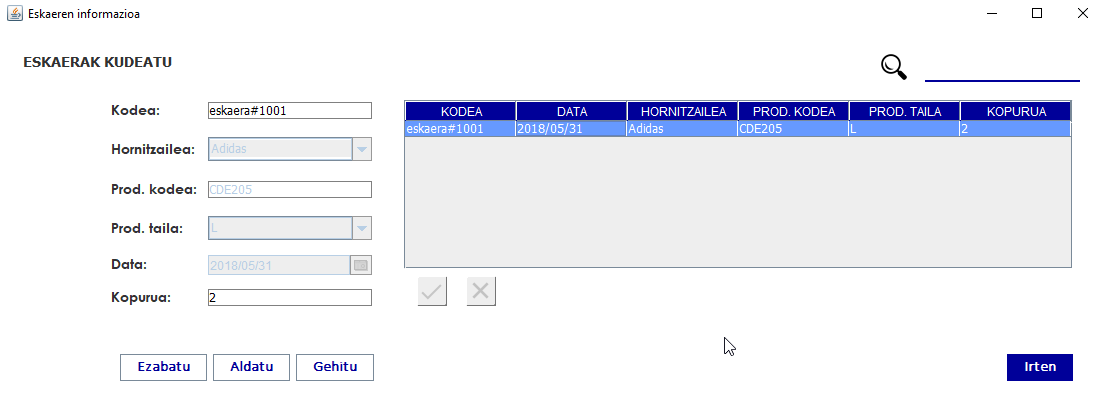
 Aldaketa ezeztatzeko/ezabatzeko sakatu behar den botoia.

* **Gehitu** sakatzen bada, hornitzaile berri bat gehitzeko leihoa zabalduko da eta bertan, erabiltzaileak datuak idatzi eta gorde egingo dituen.



72. Irudia: Aplikazioa – hornitzailea gehitu

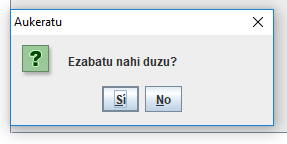
* Menu nagusian **ESKAERA** aukeratzen bada, eskaera kudeatzeko leiho bat agertuko da. Taulan, erregistratuta dauden eskaera guztiak agertuko dira eta hauetako lerro batean sakatzean, eskuineko eremuak beteko dira, lerro horretako datuekin.



73. Irudia: Aplikazioa – eskaerak kudeatu

Leiho honetan, botoi desberdinak daude.

* **Irten** sakatzen bada, leiho hau itxi eta aurrekora joango da.
* **Ezabatu** sakatzen bada, aukeratutako lerro hori ezabatuko da (ez badago ezer aukeratuta, mezu bat agertuko da). Ezabatu ahal izateko, baieztatzeko mezu bat agertuko da.



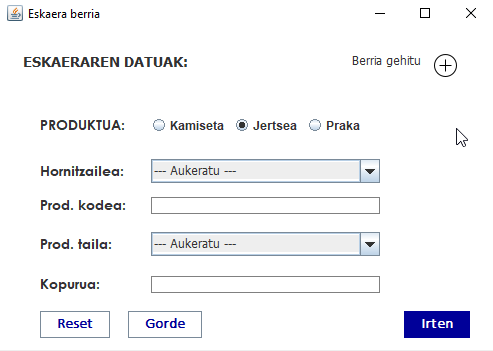
74. Irudia: Aplikazioa – denda ezabatu

* **Aldatu** sakatzen bada, aukeratutako lerro hori aldatzeko aukera ematen du (ez badago ezer aukeratuta, mezu bat agertuko da).

 Aldaketa gordetzeko sakatu behar den botoia.

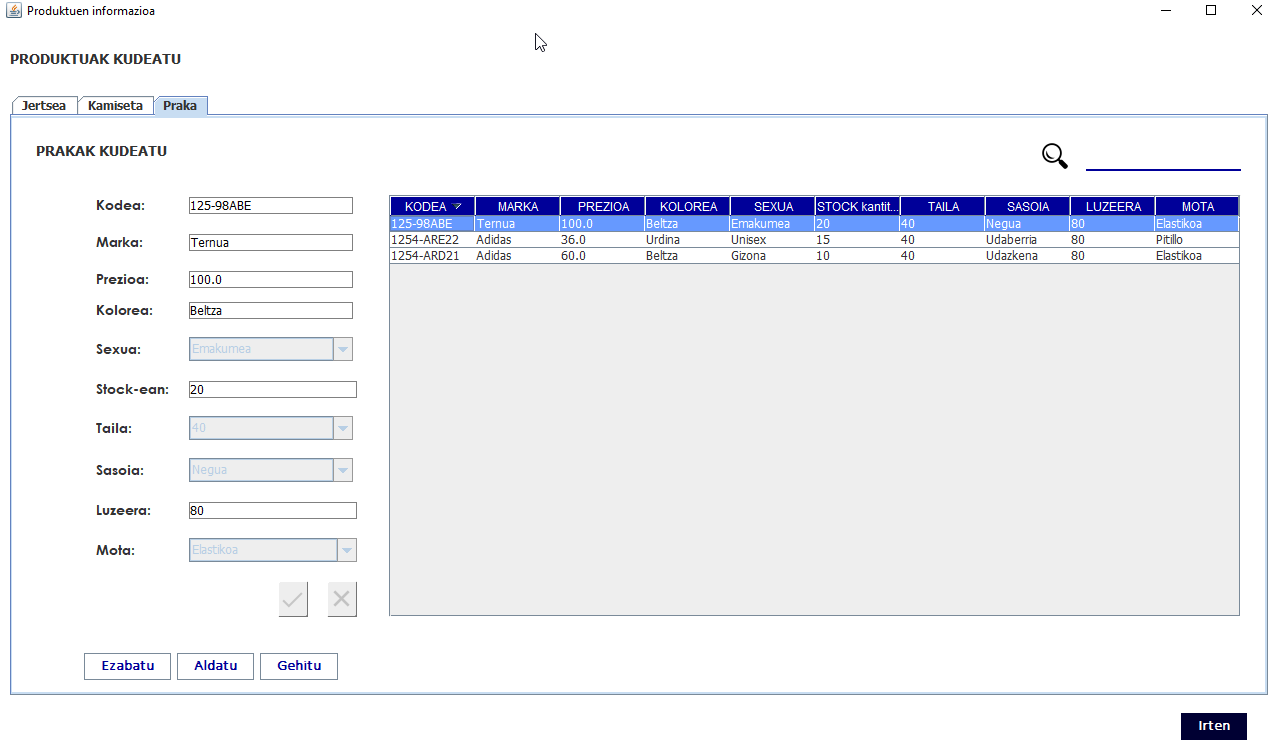
 Aldaketa ezeztatzeko/ezabatzeko sakatu behar den botoia.

* **Gehitu** sakatzen bada, eskaera berri bat gehitzeko leihoa zabalduko da eta bertan, erabiltzaileak datuak idatzi eta gorde egingo dituen.



75. Irudia: Aplikazioa – eskaera gehitu

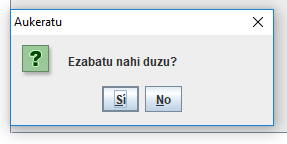
* Menu nagusian **PRODUKTUA** aukeratzen bada, produktuak kudeatzeko leiho bat agertuko da, non hiru panel agertuko diren, produktu bakoitzarentzat bat (kamiseta, jertsea eta praka). Panel bakoitzeko taulan, erregistratuta dauden produktu guztiak agertuko dira eta hauetako lerro batean sakatzean, eskuineko eremuak beteko dira, lerro horretako datuekin.



76. Irudia: Aplikazioa – produktuak kudeatu

Leiho honetan, botoi desberdinak daude.

* **Irten** sakatzen bada, leiho hau itxi eta aurrekora joango da.
* **Ezabatu** sakatzen bada, aukeratutako lerro hori ezabatuko da (ez badago ezer aukeratuta, mezu bat agertuko da). Ezabatu ahal izateko, baieztatzeko mezu bat agertuko da.



77. Irudia: Aplikazioa – denda ezabatu

* **Aldatu** sakatzen bada, aukeratutako lerro hori aldatzeko aukera ematen du (ez badago ezer aukeratuta, mezu bat agertuko da).

 Aldaketa gordetzeko sakatu behar den botoia.

 Aldaketa ezeztatzeko/ezabatzeko sakatu behar den botoia.

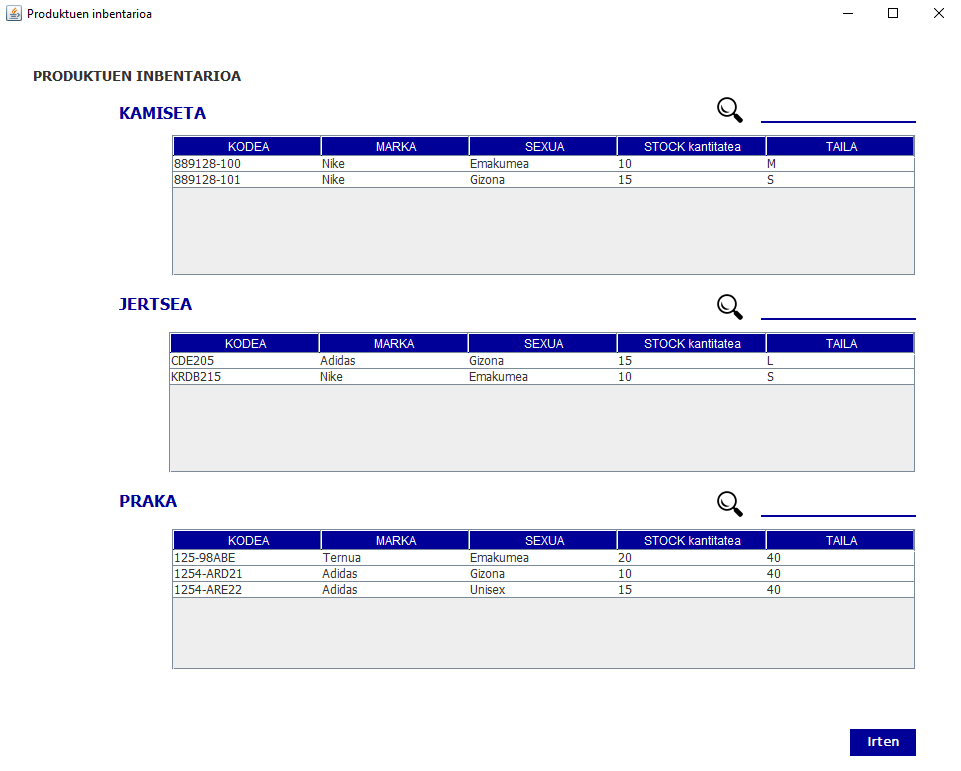
* **Gehitu** sakatzen bada, eskaera berri bat gehitzeko leihoa zabalduko da eta bertan, erabiltzaileak datuak idatzi eta gorde egingo dituen.



78. Irudia: Aplikazioa – kamiseta gehitu

**OHARRA:** Jertsea eta Prakak kudeatzeko, modu berdinean izango litzateke.

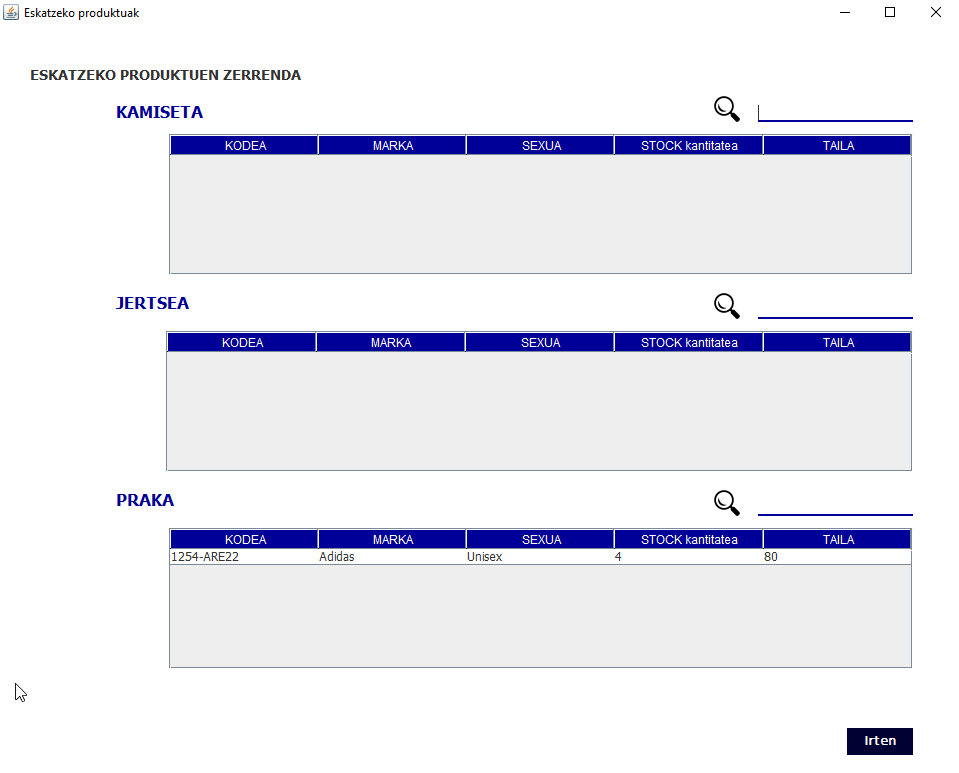
* Menu nagusian **INBENTARIOA** aukeratzen bada, produktu guztien inbentarioa agertuko den leiho bat agertuko da, hiru taula desberdinetan banatuta (kamiseta, jertsea eta praka). Taula bakoitzean, erregistratuta dauden produktu guztiak agertuko dira.



79. Irudia: Aplikazioa – inbentarioa

Leiho honetan, botoi desberdinak daude.

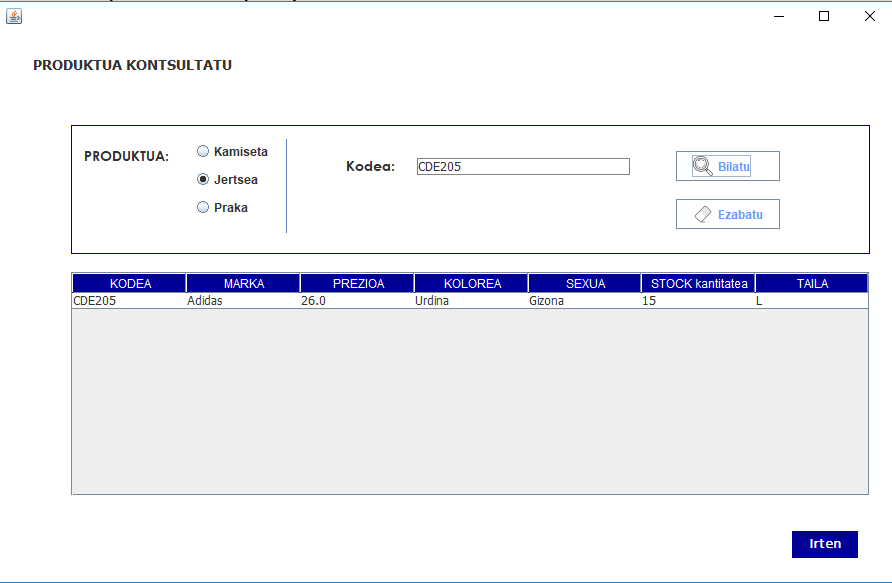
* **Irten** sakatzen bada, leiho hau itxi eta aurrekora joango da.
* Menu nagusian **ESKATZEKOAK** aukeratzen bada, eskatzeko produktuen zerrenda bat agertuko den leiho bat agertuko da, hiru taula desberdinetan banatuta (kamiseta, jertsea eta praka). Taula bakoitzean, erregistratuta dauden produktu guztiak agertuko dira, non stock-ean dauden kantitatea 5 baino gutxiago den.



80. Irudia: Aplikazioa – inbentarioa

Leiho honetan, botoi desberdinak daude.

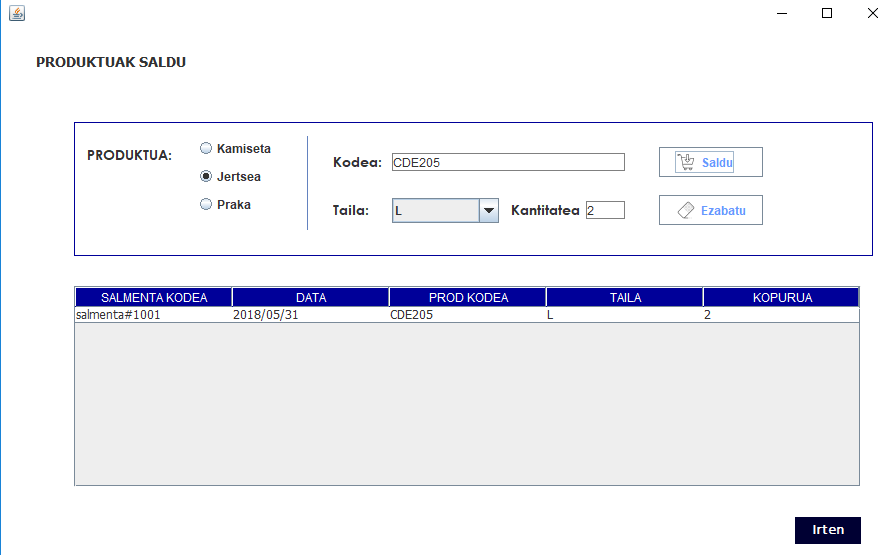
* **Irten** sakatzen bada, leiho hau itxi eta aurrekora joango da.
* Menu nagusian **KONTSULTA** aukeratzen bada, kontsultatu nahi den produktua erregistratuta baldin badago, taulan agertuko da. Ez badago erregistratuta berriz, mezu bat agertuko da.



81. Irudia: Aplikazioa – kontsulta

Leiho honetan, botoi desberdinak daude.

* **Irten** sakatzen bada, leiho hau itxi eta aurrekora joango da.
* Menu nagusian **SALMENTA** aukeratzen bada, saldu nahi den produktua erregistratuta baldin badago, taulan agertuko da eta saldu egingo da. Ez badago erregistratuta berriz, mezu bat agertuko da. Produktua saltzen denean, stock kantitatea murriztuko da.



82. Irudia: Aplikazioa – salmenta

Leiho honetan, botoi desberdinak daude.

* **Irten** sakatzen bada, leiho hau itxi eta aurrekora joango da.