# PROJE 3

Egemen Çakır

*Kocaeli Bilgisayar Mühendisliği*
*Email: cakiregemen0@gmail.com*

## I. GIRIS

Bu projenin amacı,Bu proje kapsamında sanal ortamda arsa ve market ile ilgili (alım-satım-kiralama-işletme vb.) ticari aksiyonları yerine getiren bir platformun veritabanı sistemini tasarlamanız ve bu veritabanı üzerinde gerekli işlemleri gerçekleyen bir oyun geliştirmeniz amaçlanmaktadır. Tasarlanacak veritabanı, metaland platformu üzerindeki kullanıcılar için varlıklara yönelik hesaplara ve tüm işlemlere ilişkin bilgileri organize bir şekilde yönetilmesine yardımcı olacaktır. Bu sayede kullanıcıların ihtiyacı olan bilgilere daha kolay ulaşabilmesi sağlanmış olacaktır. Bir veritabanı tasarımının ilk aşamasında sistemin ihtiyaçlarının belirlenmesi ve depolanacak bilgi türlerinin tanımlanması için Varlık-İlişki (ER) diyagramı oluşturulmalıdır. ER diyagramı sistem içerisinde var olabilecek varlıkların ve aralarındaki ilişkilerin görsel olarak ifade edilmesi için kullanılır. Geliştirme sırasında, ER diyagramı gereksinimlerin daha açık ve özlü bir şekilde haritalanmasına yardımcı olmaktadır. .Ayrıca Oyunun kuralları:

- Oyun karesel alanlara bölünmüş bir grid alan veya harita üzerinden görselleştirilerek oynanacaktır.Bu karesel alanlar oyun içerisindeki arsalara ve üzerinde kurulacak işletme alanlarına karşılık gelecektir. Bu alanın kaç kareden oluşacağını yönetici belirleyecektir. (Örneğin 3x3 veya 4x5 vb.).
- Sistem oyuna dahil olan yeni oyunculara başlangıçta belirli bir miktar para, yiyecek ve eşya verecektir (Bu üç özellik sayısal değerler ile gösterilecektir).
- Günlük olarak her oyuncudan sahip olduğu yiyecek, eşya ve para miktarından sabit miktarda eksiltme yapılacaktır. Yiyecek veya eşya miktarlarından biri tamamen tükenen oyuncu oyunu kaybetmiş olacaktır.
- Her kullanıcı bir işletmede çalışma hakkına sahip olacaktır. Çalışma saatleri ve günlük kazanacağı para miktarı mağaza sahibiyle yaptığı anlaşmaya göre belirlenecektir. Çalışma saatleri içerisinde oyuncunun başka bir ticari işlem yapması mümkün olmayacaktır.
- Oyun ilk başlatıldığında tüm alanlar (arsalar) yöneticiye ait olacaktır. Ayrıca yöneticiye ait sınırsız oyuncu kapasitesine sahip birer market, mağaza ve emlak bulunacaktır.
- Markette çalışan bir oyuncunun günlük yiyecek miktarı, mağazada çalışan bir oyuncunun günlük eşya miktarı ve emlakçıda çalışan bir oyuncunun ise günlük para miktarında çalıştığı süre boyunca sabit eksiltme yapılmayacaktır.
- Oyuncular sahip oldukları parayla işletme sahiplerinin verdiği fiyata göre marketlerden yiyecek,mağazalardan eşya ya da emlak noktalarından başka bir oyuncuya ait arsa veya işletme (market,mağaza, emlak) satın alabilecek veya işletmeyi kiralayabilecektir. Emlakçı kendi belirlediği bir miktar üzerinden alıcı-kiralayıcı ve satıcı-kiraya verenden bir komisyon ücreti alacaktır.
- Bir oyuncunun sıfırdan işletme kurabilmesi için öncelikle bir arsa sahibi olması gerekmektedir.Sonrasında bu arsa üzerine market, mağaza ya da emlak işletmelerinden herhangi birisini yöneticinin belirlediği para değeri karşılığında yönetici emlak noktasından işlem yaparak inşa edebilecektir.
- Oyuncu bir emlak aracılığıyla başka bir girişimcinin sattığı arsayı alabileceği gibi yönetici emlak noktası üzerinden de bedeli karşılığında satın alabilir.
- Oyuncu üzerine işletme kurulmamış en fazla 2 arsaya aynı anda sahip olabilir. Sistem böyle bir oyuncuya daha fazla boş arsa satışına izin vermeyecektir.
- Her işletme ilk kurulduğunda 1. seviye olarak başlayacak ve bünyesinde en fazla 3 oyuncu çalıştırma kapasitesine sahip olacaktır. Sonraki her seviye için işletmenin oyuncu çalıştırma kapasitesi ikiye katlanarak artacaktır. (1. seviye - 3 oyuncu; 2. seviye-6 oyuncu; 3. seviye 12 oyuncu).
- 1. seviyeye sahip bir işletme müşteri oyuncularla yaptığı ticari işlemlerden elde ettiği gelirler dışında yöneticinin belirlediği sabit bir gelire sahip olacaktır. Bu sabit gelir işletmenin her bir üst seviye geçişinde yine yöneticinin belirlediği bir oran üzerinden artırılacaktır.
- Bir işletme bir hafta boyunca tam kapasite çalışması halinde bir sonraki seviyeye geçecektir.
- Demo sırasında oyundaki diğer tüm kullanıcıların işlem yapmadığı ve tüm parametrelerin sabit kaldığı varsayılarak istenilen gün sayısı kadar tarih ileriye alınmalı ve böylece oyun hızlandırılarak ileri bir tarihteki sonuçlar görüntülenebilmelidir.
- Oyuncu oyun sırasında kendisine ait oyun bilgilerini ve detaylarını (giderler, satın almalar,harcamalar, varlıklar, bütçe durumu, tüm geçmiş tercihler ve aksiyonlar) açılır bir pencere üzerinden görüntüleyebilmeli, benzer şekilde yönetici ise tüm oyuncuların bilgilerini izleyebilmelidir.
- Tüm bu veriler ve veri hareketleri veritabanında tutulmalı ve oyunun ilerleyişine göre güncellenmelidir. Arayüzde gösterilen bilgilerin tamamı yine bu veritabanını üzerinden yapılan gerekli sorgularla elde edilmelidir.

- Tasarlanan arayüzde oyuncunun kendi yaptığı tüm işlemler gerek oyuncuya göre gerekse işletmeye göre filtrelenip ve kronolojik olarak gösterilmelidir.

Bu proje veritabanı ve veritabanı normalleştirmeyi ögrenme acısından oldukca önemlidir. Bu projeyi java dilinde yazdım ve arayüzü kodlamak için swingi kullandım.

## II. Adimlar

*Adım 1*

İlk adım olarak main classını actım ve JFrame'i extends ettim daha sonra detaylı olarak anlatacagım veritabanını javaya baglamak için Database sınıfını oluşturdum.Bu sınıfta metotlar,constructor ve özellikler içeriyor.Bu özellikler static Connection con,static Statement stat , ResultSet rs özellikleri var.Metotlar ve constructor ise:

- Database constructor:İçindeki kodlar ile veritabanı ile baglantıyı saglıyor.Kodu:con=DriverManager.getConnection (”jdbc:mysql://localhost:3306/proje6”,”root”,”*****”); stat=(Statement) con.createStatement();
- add metodu:Parametre String türünde bir sorgu alıyor ve stat.executeUpdate(sql); kodu ile gelen sorgu içindeki tabloya veri ekliyor.
- update metot:Parametre String türünde bir sorgu alıyor ve stat.executeUpdate(sql); ile gelen sorgu ile tablodan silme veya update işlemleri gerçekleştiriyor.
- Sorgu metod:Dönüş tipi olarak ResultSet olarak bir deger dönderir ve parametre olarak String türünde bir sorgu alıyor ve rs=stat.executeQuery(sql); kodu ile rs ye sorgu sonucları aktarılır ve return rs; ile döndürülür.
- dAte metodu: Bu metod String dönüş tipine sahip ve içindeki kod ile mevcut tarihi String olarak döndürür.

Bu sınıfı yazarak veritabanı baglantısını ve veritabanı işlemlerini metotlara yazdım.

*Adım 2*

Main classının ilk dizaynı olarak 4 JLabel ve 2 JButton kullanıdım.Bu jLabellardan birini başlık olarak kullandım ve diger JLabellaı fotograf ekledim.İki butonun birini giris için digerini kayıt için kullanmayı planladım.Daha sonra pencereyi kapatmak için setVis() adında bir metot yazdım.Daha sonra kayıt butonuna action ekledim.Burada setVis() metodunu ve kayıt paneli için Kayıt classını oluşturdum.Bu class ın dizaynında 3 JButton ,JLabeller ve JTextFieldler kullandım.Bu Jlabeller ve JTextFieldleı kullanıcı adını , kullanıcı soyadını ve sifresini almak için kullandım.3 Butondan birini Main classına dönmek için diger ikisini yönetici kayıt ve oyuncu kayıtını yapmak için kullandım.Yönetici kayıt butonunun actionını yazdım.Burada öncelikle bir sql sorgusu ile yönetici sayısını aldım eger yönetici sayısı 0 ise yönetici kaydı için JTextFieldlerden kullanıcı adını,soyadını ve sifresini adım ve kullanıcı tablosuna bir kayıt eklemek için aldıgım bilgiler ile gerekli tablo özelliklerinide sorguya ekleyerek kayıtı ekledim.Oyunda 1 tane yönetici olacagı için yönetici sayısı 1 olsaydı bu kayıt gercekleştirilmiyecti.Burada oyunun temel özelliklerini belirledigi için YönetticiSettings adında bir class yazdım.Bu classta 17 tane JLabel,17 tane JTextField ve 2 buton ekledim.Bu 17 labeller ve textfieldler ile özelliklerin degerlerini adım.Bir buton ile Kayıt sınıfına dönmek için digerinide kayıt için kullanıdm.Sınıfta defualtSettings metodunu yazdım.Bu metot öncelikle GenelSistem tablosundan id sayısını aldım ve eger sayı 1 ise bu özelliklerin bulundugu tablolardan degerleri aldım ve JTextFieldlara yazdım.Eger 0 olsaydı varsayılan şekilde bulunan degerleri yazdırdım.Kayıt butonunun actionında textfieldlardan aldıgım degerleri eger GenelSistem id sayısna göre özelliklerin bulundugu tablolara kayıt olarak ekledim ya da update olarak degerleri degiştirdim.Şimdi oyuncu kayıt butonu için action yazdım.Burada da öncelikle yönetici sayısını aldım ve sayı 1 degil ise yönetici kaydı yapılana kadar oyuncu kaydı yapılmıyor fakat sayı 1 ise kullanıcı ad,soyad ve sifresi ile ve gerekli özellikler ile Oyuncu tablosuna ve Kullanıcı tablosuna yeni kayıt olarak ekledim.Kayıt sınıfını yaptıktan sonra giris sınıfını yapmak için Main classında Giris butonuna actionında Giris sınıfından bir nesne ürettim ve setVis() ile Main classını kapattım.Giris sınıfında 3 JLabel ve JTextField ve 2 buton kullandım.Burada Textfieldlar ve labeller ile Kullanıcıların ad , soyad ve sifresini aldım , bir butonu Maine classa dönmek için ve digerinide Giris yapmak için kullandım bu Giris butonunun action bölümünde girilen degerle ile kullanıcı tablosundan kullanıcının türü aldım ve türü Oyuncu ise OyuncuOyun sınıfından nesne olusturdum eger kullanıcı türü Yönetici ise YöneticiOyun sınıfından nesneye oluşturup giris kısmınıda bitirmiş oldum.

*Adım 3*

Bu adımda OyuncuOyun sınıfnı yazmaya başladım.Bu sınıf JFrame'i extends ediyor ve ActionListener'ı implements ediyor.Bir adet JPanel'i oyun alanı olarak kullanmak üzere contentpane'e ekledim daha sonra JLabeller ile giris yapan kullancının ad,soyadını ve güncel para,esya ve yiyecek miktarını göstermek üzere yerleştirdim.Daha sonra 4 JButton kullandım bunlardan birini Main classa dönmek için kullandım digerleri sırasıyla,kiralık liste,satılık liste ve kişisel bilgiler olarak kullandım.Bu butonların action kısımlarını yazmadan önçe contenetPane'e ekledigim panale alanların eklenmesi için Alan adında bir sınıf yazdım.Bu sınıfının içinde build1 ve build2 adında iki metod yazdım bunlar:

- build1 metod:Alan türünde bir nesne ve JPanel türünde bir nesneyi parametre olarak alıyor.Daha sonra sql sorguları ile AlanSistem tablosundan oyun boyutunu alıp boyuta göre Alan sınıfının içindeki Static 2 boyutlu matrisin her elemanını parametere olarak gelen panele ekliyor.
- build2 metod: Alan türünde bir nesne , JPanel türünde bir nesne ve 2 adet alan boyutun degerlerini içeren int degerinde degerler alıyor.Bu fonksiyonda parametre olarak gelen panele boyut sayısına göre Alan sınıfının içindeki static 2 boyutlu dizinin her bir elemanını yerleştiriyor.

Bu metotlardan build1 i kullanarak OyuncuOyun classında oyun alnaımı oluşturmuş oldum.Daha sonra kiralık liste ve satılık liste butonlarına action verdim.Bunun için KiralıkSatılıkTabloPane adında bir class oluşturdum ve parametre olarak Oject dizisi ve iki tane tam sayı deger alıyor.Bu sınıfta JTable ve JScrollPane'i ekledim ve bir adet metod yazdım.Bu metod:

- buildTablo metot:Bu metot bir adet Objecet dizisi ve bir adet int deger alıyor.Bu int degere göre kiralık alanları ya da satılık alanların listelerini tabloya ekledim.Örnegin int degeri 1 ise bu da kullanıcının kiralık listesini görmek istedigi anlamına geliyor onun için öncelikle bir arrayliste kiralıkList tablosundaki id leri ekledim eger arrayListin eleman sayısı 0'dan kucukse bir şey yapmadım fakat 0'dan buyukse bir for içinde her bir id için o id satırındaki emlak için id , kiralık isletme id sini aldım ve emlak ve kiralık isletme id kullananrak tabloda kullanılacak bilgiler için gerekli degerleri almak için sorgularda bu emlak ve isletme id lerini kullanarak tabloda gösterdim.

Bu KiralıkSatılıkTabloPane sınıfını yazdıktan sonra her iki butonun action kısmında kullandım fakat parametrelinde Object dizisini ve int degerlerini kiralık ve satılık olamsına göre farlı degerler atadım.Kisisel bilgiler butonunun action kısmını yazmak için KullanıcıBilgiPane adında bir sınıf oluşturdum.Bu sınıf parametre olarak iki adet parametre deger alıyor ve bu sınıfta 3 adet JTable ve JScrollPane kullandım.Bu tabloları doldurmak için ise bir metot yazdım.Bu metot:

- creatTables metot:bir adet int deger alıyor bu degerde kişisel bilgilerini ögrenek kişinin id'si.Bu metotta öncelikle gelen id ile kullanıcı tablosundaki kullanıcı ad,soyad,tarih ve tür bilgileri alınıyor.Daha sonra eger kullanıcı oyuncu ise Oyuncu tablosundan mevcut para,yemek ve esya degerleri alınıp tablo bire ekleniyor.Daha sonra kullanıcının id'si kullanılarak Alanlar tablosundan kişiye ait alan bilgileri alınıp eger işletme iseler İsletmeler tablosundan isletmeye ait nilgiler alınıp tablo2'ye ekleniyor.Son olarak kullanıcı id ile kullanıcını yapmıs oldugu islemleri İslemler tablosundan alıp bazı düzenlemeler ile tablo3'e ekledim.

Bu metodu yazdıktan sonra en son olarak eger kullanıcı yaptıgı islemleri silmek isterse diye tablo3 için bir MouseListener ekledim ve eger kullanıcı tablo3 ün herhangibir satırına tıkladıgında JOptionPane ile öncelikle silmek ister misiniz ? Sorusunu sorup alınan cevap ile sql sorgusu ile İslemler tablodan secilen islemin id'sini kullanarak islemi sildirdim.Bu sınıfı bitirdikten sonra kisisel bilgiler butonunda nesnesini oluşturup parametresini verdim ve OyuncuOyun Sınıfının son adımı olarak implement ettigim actionListenerın override edilmesi gereken metodu olan actionPerformend metodunu yazdım.Burada öncelikle kullanıcının sectigi alanın alan no'sunu aldım ve daha sonra İslemPanel adında bir sınıf olusturdum ve parametre olarak bu alan no'yu verdim ve OyuncuOyun sınıfnı bitirdim.

*Adım 4*

Bu adımda islemPanel adındaki sınıfı oluşturmaya başaldım.Burada 6 adet JButton ve JLabeller kullandım.JLabelleri İsletme ya da arsanın sahibinin adını ve soyadını ve isletme ya da arsanın id bilgilerini göstermek için yazdım.Butonları ise sırası ile OyuncuOyun sınıfna dönmek için,satın alma,kiralama, calısma , arsa ya da isletme isleri ve isletme kurmak için kullandım.Bir adet dflt adında bir metod yazdım ve constroctur içinde kullanıdm bu metod temel olrak butonların ve labelların düzenini ve sınıf içinde lazım olacak olan bilgiler için sorular ile özellik bilgilerini alıyor.Butonların action kısımları:

- Satın alım action:Öncelikle parametre olarak gelen alan no ile üzerinde işlem yapılacak alanın arsa ya da isletme oldugunu anlamak için Alanlar tablosundan alan tür özelligini aldım ve eger arsa ise ve arsa satılıkta ise satısta oldugu emlagın id sini JOptionPane ile gösterdim eger isletme ise Alanlar tablosundan aldıgım alan id ile İsletmeler tablosundan isletme tür özelligini aldım.İsletmenin market , magaza ise ürün birim fiyatını gösterip kullanıcıdan adet miktarıı adım ve sql sorguları ile kullanıcının ve isletme sahibinin mevcut para ,esya ve yiyecek miktarlarını güncelleyip islemler tablosuna da bu islemi ekledim.Eger emlak ise isletme TabloPane2 adında bir sınıf oluşturdum ve bu sınıfta temel olarak bir tablo üzerinde bu emlak için izin verilen satıs yetkili isletme ya da arsaları gösterdim ve MouseListener ile kullanıcının sectigi alanın satın alma islemini gerçekleştirdim.
- Kiralama action:Burda secilen alan eger arsa ise kiralama yapılamaz diye bir mesaj cıkardım.Eger isletme bir emlak ise yine TabloPane2 nesnesi olusturup tablo üzerinden bu işletmeye kiralam iszni verilen işletmelerin secilip kiralama işlemlerini yapılmasını sagladım.
- Calısma action:Burada öncelikle kullanıcının mevcut bir işinin olup olamdıgını kontrol ettim ve eger yoksa secilen alanın isletme ise isetmenin calısan sayısını ve kapasitesini alıp yeterli yerinin olup olmadıgına baktım eger yer varsa kullanıcıya maasını gösterdim ve kullanıcıdan calısma saatlerini ve calısma gün sayısını aldım ve sql ifadeleri ile Calısanlar tablosuna kayıt ekleyip secilen isletmenin calısan sayısını bir arttırıdm.
- Arsa/İsletme islem action:İsletmeciSettings adında bir sınıf oluşturdum ve bu sınıftan nesne oluşturup parametrelerini oluşturudum.
- İsletme Kurma action:Burada öncelikle alanın bir emlak ve yöneticiye ait mi olduguna baktım eger degilse ve burası bir arsa ise bu işlemi nerede gerçekleştirebilecegi yani yöneticinin emlagının id'sini JOptionPane ile gösterdim.Eger burası yöneticinin emlagı ise AlanSecimPane adında bir sınıf oluşturudm ve bu sınıfta kısaca kullanıcının mevcut arsalarını tabloda gösterdim ve action ile secilen arsanın hangi isletmeye dönüştürülecegini kullanıcıdan aldım.Örnegin emlak yazdıysa kullanıcıdan komisyon ve maas ücretlerini aldım ve Alanlar , İsletmeler , İslemler ve vb. tablolarda updateler ve kayıt

eklemeler ile işletme kurma işlemini sagladım.

Böylelikle İslemPanel sınıfını tamamlamış oldum.

*Adım 5*

Bu adımda İslemciSettings sınıfını yazdım.Bu sınıfda JButtonlar,JLabellar ve JTextField lar var .Bu labeller ve textfieldlar mevcut maas,satılan şeylerin birim fiyatı , kiralıkta ise kiralık listesindeki fiyatı ve satısta ise satıs fiyatını içeriyor.Bu sınıfta begin adında bir metod yazdım ve bu metot temel olarak bu label ve textfieldları düzenliyor.Butonların actionları:

- kaydet butonu action: Burada eger degeri degişebilecek olan degerler varsa degerleri degiştirilip veritabanına update ile güncellenmesi saglandı.
- kiralık listeden cıkar buton action:Burada öncelikle alanın türü bakılıp eger arsa degilse işletmenin kiralık listesinde olup olmadıgı kontrol edildi eger kiralık listesinde ise listeden cıkarıldı.
- satıs listedene cıkar buton action:Burada öncelikle alanın satıs listesinde olup olmadıgı kontrol edildi eger satıs listesinde ise listeden cıkarıldı.
- satılık ekle buton action: Burada TabloPane2 nin nesnesini oluşturudm ve mevcut emlak listesini tabloda gösterdim ve kullanıcının sectigi emlaga satıs izni verilmiş oldu.
- kiralık ekle buton action:Burada TabloPane2 nin nesnesini oluşturudm ve mevcut emlak listesini tabloda gösterdim ve kullanıcının sectigi emlaga kiralık izni verilmiş oldu.
- Yükselme butonu action:Burada yöneticinin belirledigi yükseltme fiyatını ve isletmenin mevcut seviyesini aldım ve seviyesine göre yükseltme fiyatına ekstra para ekleyip kullanıcaya JOptionPane ile soru sorudum eger onaylarsa bu sefer kullanıcının parasının yetip yetmedigine baktım ve eger yeterse sql ifadeleri ile kullanıcının para mikatarını ve isletmesinin seviye ve kapasite miktarını güncellesim.

Böylelikle bu sınıfıda bitirmiş oldum.

*Adım 6*

Bu adımda Giris kısmında eger giris bilgileri alınan kişi yönetici ise onun için YöneticiOyun sınıfını yaptım fakat OyuncuOyun sınıfı ile tek farkı ekstra Oyuncu BİIgi ve ayar butonları olamsı.Bu butonlar:

- oyuncu bilgi buton action:Burada OyuncuListePane sınıfını yazdım temel olarak oyundaki oyuncu listesini tabloya ekliyor ve yönetici hangi oyuncu satırına tıklarsa o oyuncunun bilgilerini görebiliyor.
- Ayar buton action:Burada yöneticinin oyun ayarlarını dinamik bir şekilde degiştirilmesi için oluşturdum.

Ve son olarak Main kısmında oyuncuların günük giderlerini , calısma ve kiralık bitis ve oyundan silinecek oyuncuların kontrolünü saglayan 3 adet metod yazdım ve demo görünüm için mevcut saat ve tarihi ileri bir tarihe ayarlamak için bir buton koydum böylelikle oyunun tarihi ileri alarak neler olup bittigini oyunun mantıgını iyi bir şekilde gösterebildim.Böylelikle projeyi bitirmiş oldum.

## III. Özet

Bu projenin amacı, veritabanı kullanarak meta-land adında bir oyun geliştirmekti.Bu proje yapımında veritabanını ve veritabanı ile ilgili işlemleri uygulayarak ve test ederek daha iyi ögrendim ve benimsedim.Ayrıca projenin arayüzünü java swing ile yaptım ve swing componentlerini daha detaylı ögrenme vekullanam fırsatı yaşadım.

## IV. Sonuç

Tüm kontroller yapıldıktan sonra artık ana menüde yönetici ya da oyuncu kaydı yapabiliyor,oyuna girebiliyor , alım,satım,yükseltme , inşa etme , kiralama ,vb. işlemleri gerçekleştirip oynumu oynayabildim.Fakat projeyi yaparken bazı problemlerle karşılaştım bunlar:

- Öncelikle sorgu sonuclarının yanlış ya da hata verip programın durdurulaması gibi sorunlarla karşılaştım.Bu sorunlar için, yazdıgım sorguları kontrol ederek ve veritabanını kontrol ederek bu sorunları çözdüm.
- Karşılaştıgım diger sorun ise tablolara yeni kayıt eklerken bazı hatalar almamdı bunun sebebide bazı özelliklerin adını yanlış ya da degerini veri tipini yanlış girmemden kaynaklandıgını anladım ve bunlarıda kontrol ederek düzelttim.

Genel olarak projenin sonuçları böyleydi.

## V. Normalizasyon Adimlari

Normalizasyonun iki temel amacı vardır. Veri tabanında veri tekrarlarını ortadan kaldırmak ve veri tutarlılığını (doğruluğunu) artırmak. Basitçe tanımlamak gerekirse, normal formlar normalizasyon seviyeleridir. Bu seviyeler gereksiz veri tekrarlarını ne derecede engellediği ve tutarlılığı ne kadar sağladığına bağlı olarak derecelendirilir. Seviye yükseldikçe veri tutarlılığı artar, veri tekrarı düşer. Normalizasyon seviyeleri 1NF (Birinci Normal Form), 2NF, 3NF, BCNF(Boyce-Codd Normal Form, 3.5NF'de denir), 4NF şeklinde adlandırılır ve yukarı doğru devam eder. Ancak daha yukarı normalizasyon seviyeleri çok nadiren kullanılır çünkü çoğu zaman uygulanması mümkün olmayabilir.

- 1. NF : Bir veri tabanının 1NF olabilmesi için aşağıdaki özellikleri karşılayabilmesi gerekir: *Aynı tablo içinde tekrarlayan kolonlar bulunamaz, *Her kolonda yalnızca bir değer bulunabilir. *Her satır bir eşsiz anahtarla tanımlanmalıdır (Unique Key - Primary Key). Bu özellikler dolayısıyla verilen özellikler , ekledigim özellikler ve oyunun bileşenlerini hesaba katıp tablolar şeklinde ayırmaya başladım ve her tabloya bir id özellik koydum bu özellik bir primary key ve her yeni kayıtta kendini 1 artırarak devam ediyor.Böylelikle 1 NF yi bitirdim.
- 2. NF:Bir veri tabanının 2NF olabilmesi için aşağıdaki özellikleri karşılayabilmesi gerekir: *Tablo 1NF olmalıdır, *Anahtar olmayan değerler ile kompozit (bileşik) anahtarlar arasında kısmi (partial) bağımlılık durumu oluşmamalıdır. *Kısmi bağımlılık durumu, anahtar olmayan herhangi bir değer kompozit bir anahtarın yalnızca

bir kısmına bağıl ise oluşur. *Herhangi bir veri alt kümesi birden çok satırda tekrarlanmamalıdır. *Bu tür veri alt kümeleri için yeni tablolar oluşturulmalıdır. Ana tablo ile yeni tablolar arasında, dış anahtarlar (foreign key) kullanılarak ilişkiler tanımlanmalıdır. Bu özellikler cercevesinde kısmi bagımlılıklar ve veri tekrarları söz konusuydu.Örnek verecek olursam Oyuncular tablosunun özelliklerine baslangıcyemek,baslangıcesya ve baslangıcpara özelliklerinide eklemistim fakat yönetici bu özellikleri her degiştirdiginde oyuncu sayısı kadar satır güncelleme işlemine tabi tutulacaktı bunun önüne geçmek için bu ve bunlar gibi özellikleri başka bir tablo oluşturup oraya yerleştirdim ve foreign key/keyler ile baglantılarını sagladım.Kısmi bagımlılık sorununda ise her tabloda ekledigim id ler dışında bir anahtar ya da anahtarlar cıkmadı ve böylelikle 2 NF da bitmiş oldu.

- 3. NF:Bir veri tabanının 3NF olabilmesi için aşağıdaki özellikleri karşılayabilmesi gerekir: *Veri tabanı 2NF olmalıdır, *Anahtar olmayan hiç bir kolon bir diğerine (anahtar olmayan başka bir kolona) bağıl olmamalı ya da geçişken fonksiyonel bir bağımlılığı (transitional functional dependency) olmamalıdır. Başka bir deyişle her kolon eşsiz anahtara tam bağımlı olmak zorundadır.Bu özellikler baglamında tablolarımda geçişken bagımlı özellikler vardı.Örnegin İsletmeler tablosundadi seviye ve kapasite özellikleri burada kapasite seviyeye baglı fakat seviye anahtar olmayan bir özellik oldugu için bu ve bunlar gibi özellikleri başka bir tabloya alıp foreign key/keyler ile baglantıyı sagladım.Böylelikle 3 Nf'uda halletmiş oldum.

Bu işlemler sonucunda veritabanımın normalizasyon işlemlerini tamamladıgımı söyleyebilirim.

## VI. KAYNAKCA


- https://www.tabnine.com/code/java/methods/javax.swing.JTable/addMouseListener
- https://www.w3schools.blog/select-record-using-statement-jdbc
- http://beltslib.net/veri-tabanlarinda-normalizasyon.html
- https://www.visual-paradigm.com/guide/data-modeling/what-is-entity-relationship-diagram/
- https://www.guru99.com/database-normalization.html