

PROJE KONUSU

Bu projede toptan giyim kumaş işi,satış ve stok operasyonları ile birlikte konu olarak alınmıştır.

PROJE AMACI

Projede ana amaç bir işletmenin stok ve satış operasyonlarını sade ve anlaşılır bir şekilde gerçekleştirmek ve bu işlemleri yaparken karmaşıklık,zorluk ve zaman kaybını minimuma indirmektir.

PROJE KAPSAMI VE SINIRLARI

Kullanıcı satışları ve stokları iki ayrı pencere üzerinden görebilecek,bu pencerelerde satışlar için; satan personel adı,satıldığı müşteri adı,ürün adı,adet bilgisi ve satış tarihi bilgilerini görebilecek.Stoklar penceresinde ise; stoğa eklenen ürün adı,ekleyen personel adı,adet bilgisi ve tarih bilgilerini görebilecektir.Kullanıcı dilerse aynı pencereler üzerinden satış veya stok işlemi yapabilir,toplam satış ve toplam stok rakamlarında ulaşabilir.

Uygulama üzerinde yer alan uygulama işlemleri,ürünler,personel işlemleri,müşteriler ve stok ve satış işlemleri alt menüleri ile farklı özelliklere erişilebilir.Uygulama işlemleri menüsü altında uygulama koyu temaya çevrilebilir veya çıkış yapılabilir.Ürünler menüsü altında ürün ve kategori bilgileri eklenip yine bu bilgiler düzenlenebilir.Personel işlemleri menüsü altında yeni personel,yeni yetki ve yeni hesap eklenip yine bu eklemeler üzerinde değişiklikler yapılabilir.Müşteriler menüsü altında yeni müşteri bilgileri eklenip bu bilgilerde değişiklikler yapılabilir.Stok ve satış işlemleri menüsü altında ise daha önceden yapılmış stok ve satış işlemleri üzerinde güncellemeler yapılabilir.

Projede yetkilendirme özelliği mevcuttur.Her hesaba farklı yetkiler atanabilir ve atanan bu yetkiler doğrultusunda erişebildikleri bilgiler,menüler ve yapabildikleri işlemler farklı olacaktır.Yapılan tüm işlemler ve uygulama üzerinden girilen tüm veriler veritabanında kayıt altına alınacaktır.Ayrıca projede katmanlı mimari esaslarına uyulmaya çalışılmıştır.

PROJEDE YARARLANILAN MATERYALLER

Projede standart Java kütüphaneleri,Junit kütüphaneleri ve swing kütüphanelerinden yararlanılmış ve phpMyAdmin üzerinden MySql veri tabanı kullanılmıştır.

TEKNİK ŞARTNAME

Tanım

- İŞİN ADI:GİYİMTEK
- İŞİN TANIMI:Toptan kumaş veya tekstil ürünlerinin alım satımlarını yapan firmaların depo operasyonlarının yönetilmesi.
- Uygulama bir masaüstü uygulaması olacak olup Java dilinde kodlanacaktır.Projemizde MySQL Database’i kullanılacaktır

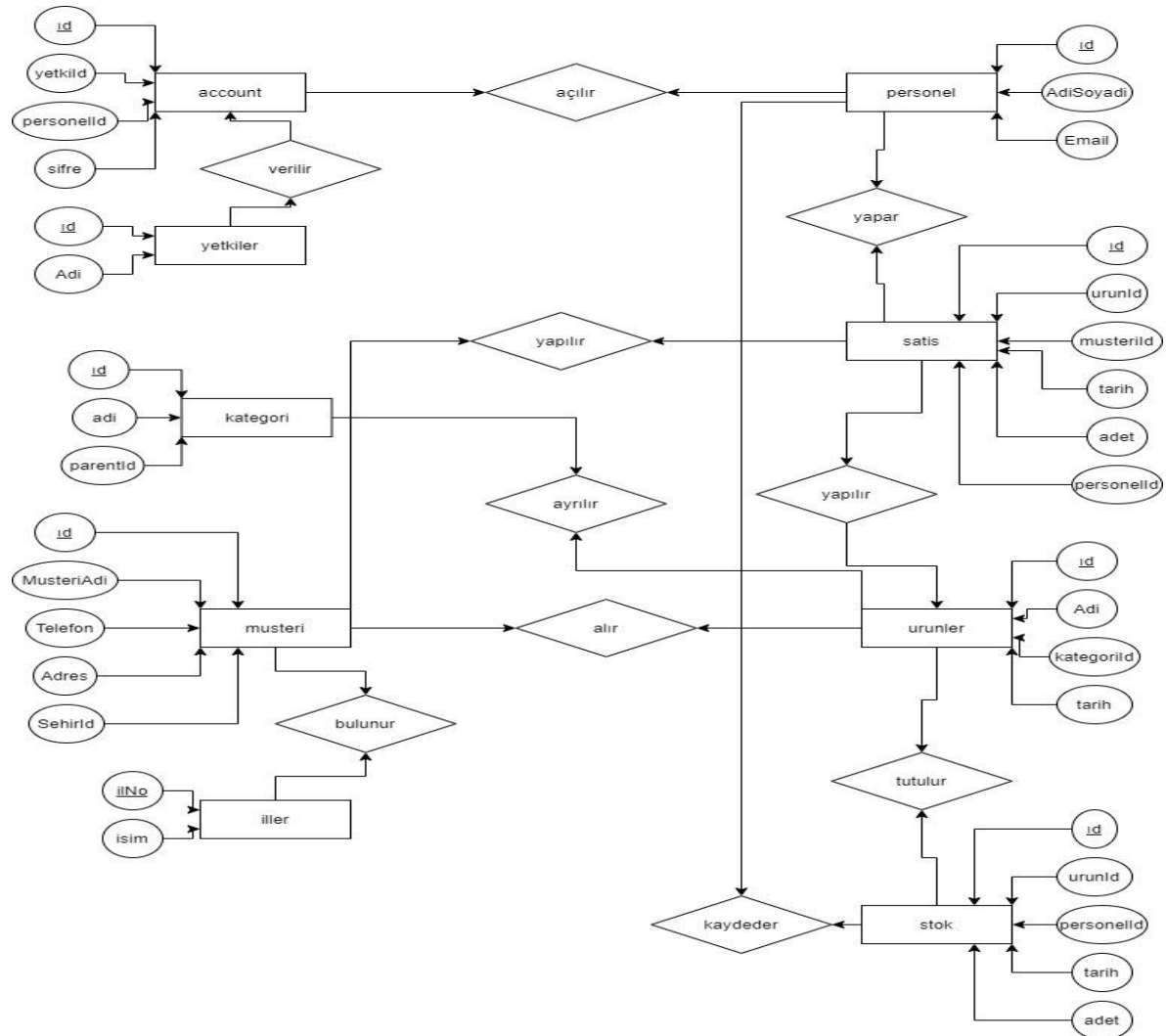
Masaüstü Uygulaması

- Uygulamada kullanıcı girişi olacaktır
- Kullanıcılara yönelik yetkilendirme yapılabilecektir
- Yetkilendirmeler ile verilmiş şifreler farklı menülere girişi engellemelidir
- Yeni kullanıcı kayıt olabilecektir
- Stokta kaydı bulunan ürünlerin satışı veya alımı yapıldığı taktirde stokta bulunan mevcut adetleri yapılan işlemle birlikte güncellenebilmelidir.
- Yeni stok carisi oluşturulabilecektir
- Kayıt edilen ürünlerin cinsi ifade edilmelidir
- Ürün satışı yapacak personelin bilgileri alınıp saklanabilmelidir ve güncellenebilmelidir.
- Ürünlere kategoriler halinde ayrılabilmeli ve yapılan satışlar kategori şeklinde görülebilmelidir

Modüller

- Kullanıcı Girişi modülü
- Ürünler modülü
- Personel İşlemleri modülü
- Müşteri modülü
- Stok İşlemleri Modülü
- Satış İşlemleri Modülü

Veritabanı ER Modeli



MALİYET RAPORU

HAFTA	PROJE FAZ	SÜREÇ	AÇIKLAMA	PERSONEL	GÜN	ÜCRET	MALİYET
1	Analiz	İhtiyaç Belirleme	Projede yapılması istenen özellikleri belirleme.	2	7	15	200
2	Analiz	İhtiyaç Belirleme	Yapılacak uygulama için teorik ve pratik eksiklerin giderilmesi	2	7	15	200
3	Geliştirme	Veri Tabanı	MySql üzerinde projeye uygun bir veri tabanı oluşturma	2	7	15	300
4	Geliştirme	Uygulama Geliştirme	Kullanıcı Girişi Modülünün oluşturulması	2	7	15	300
5	Geliştirme	Uygulama Geliştirme	Ürün modülünün oluşturulması	2	7	15	300
6	Geliştirme	Uygulama Geliştirme	Personel İşlemleri modülünün oluşturulması	2	7	15	300
7	Geliştirme	Uygulama Geliştirme	Stok İşlemleri modülünün oluşturulması	2	7	15	300
8	Geliştirme	Uygulama Geliştirme	Satış işlemleri modülünün oluşturulması	2	7	15	300

PROJENİN KOD VE UYGULAMA KISIMLARINDAN BAZI EKРАН GÖRÜNTÜLERİ

Contract(Types) Sınıflarına Ait Örnekler;

```
J KategoriContract.java ❸
3 public class KategoriContract {
4     private int id;
5     private String adi;
6     private int parentId;
7
8     public int getId() {
9         return id;
10    }
11
12    public void setId(int id) {
13        this.id = id;
14    }
15
16    public String getAdi() {
17        return adi;
18    }
19
20    public void setAdi(String adi) {
21        this.adi = adi;
22    }
23
24    public int getParentId() {
25        return parentId;
26    }
27
28    public void setParentId(int parentId) {
29        this.parentId = parentId;
30    }
31
32    @Override
33    public String toString() {
34        // TODO Auto-generated method stub
35        return adi;
36    }
37
38    public class SatisContract {
39        private int id;
40        private int musteriId;
41        private int personelId;
42        private int urunId;
43        private int adet;
44        private String tarih;
45
46        public int getId() {
47            return id;
48        }
49
50        public void setId(int id) {
51            this.id = id;
52        }
53
54        public int getMusteriId() {
55            return musteriId;
56        }
57
58        public void setMusteriId(int musteriId) {
59            this.musteriId = musteriId;
60        }
61
62        public int getPersonelId() {
63            return personelId;
64        }
65
66        public void setPersonelId(int personelId) {
67            this.personelId = personelId;
68        }
69
70        public int getUrunId() {
```

Complex Contract Types Sınıflarına Ait Örnekler;

```
5 public class SatisContractComplex {
6     private int id;
7     private String musteriAdi;
8     private String personelAdi;
9     private String urunAdi;
10    private int adet;
11    private String tarih;
12
13    public int getId() {
14        return id;
15    }
16
17    public void setId(int id) {
18        this.id = id;
19    }
20
21    public String getMusteriAdi() {
22        return musteriAdi;
23    }
24
25    public void setMusteriAdi(String musteriAdi) {
26        this.musteriAdi = musteriAdi;
27    }
28
29    public String getPersonelAdi() {
30        return personelAdi;
31    }
32
33    public void setPersonelAdi(String personelAdi) {
34        this.personelAdi = personelAdi;
35    }
36
37    public String getUrunAdi() {
```

```
5 public class StokContractComplex {
6     private int id;
7     private String personelAdi;
8     private String urunAdi;
9     private String tarih;
10    private int adet;
11
12    public int getId() {
13        return id;
14    }
15
16    public void setId(int id) {
17        this.id = id;
18    }
19
20    public String getPersonelAdi() {
21        return personelAdi;
22    }
23
24    public void setPersonelAdi(String personelAdi) {
25        this.personelAdi = personelAdi;
26    }
27
28    public String getUrunAdi() {
29        return urunAdi;
30    }
31
32    public void setUrunAdi(String urunAdi) {
33        this.urunAdi = urunAdi;
34    }
35
36    public String getTarih() {
37        return tarih;
```


D.A.L Sınıflarına Ait Örnekler;

```
8 public class KategoriDal extends ObjectHelper implements DALInterfaces<KategoriContract> {
9
10     @Override
11     public void Insert(KategoriContract entity) {
12         Connection connection = getConnection();
13         try {
14             Statement statement = connection.createStatement();
15             statement.executeUpdate("INSERT INTO Kategori (Adi, ParentId) VALUES ('" + entity.getAdi() + "','"
16                                     + entity.getParentId() + "')");
17             statement.close();
18             connection.close();
19         } catch (SQLException e) {
20             // TODO Auto-generated catch block
21             e.printStackTrace();
22         }
23     }
24
25     @Override
26     public List<KategoriContract> GetAll() {
27         List<KategoriContract> dataContract = new ArrayList<KategoriContract>();
28         Connection connection = getConnection();
29         KategoriContract contract;
30         try {
31             Statement statement = connection.createStatement();
32             ResultSet resultSet = statement.executeQuery("SELECT * FROM Kategori");
33             while (resultSet.next()) {
34                 contract = new KategoriContract();
35                 contract.setId(resultSet.getInt("Id"));
36                 contract.setAdi(resultSet.getString("Adi"));
37                 contract.setParentId(resultSet.getInt("ParentId"));
38
39                 dataContract.add(contract);
40             }
41         }
42     }
43 }
```

```
18 public class StokDal extends ObjectHelper implements DALInterfaces<StokContract> {
19
20     @Override
21     public void Insert(StokContract entity) {
22
23         Connection connection = getConnection();
24         try {
25             Statement statement = connection.createStatement();
26             statement.executeUpdate(
27                 "INSERT INTO Stok (PersonelId, UrunId, Tarih, Adet) VALUES (" + entity.getPersonelId() + ","
28                     + entity.getUrunId() + "','" + entity.getTarih() + "','" + entity.getAdet() + "')");
29             statement.close();
30             connection.close();
31         } catch (SQLException e) {
32             // TODO Auto-generated catch block
33             e.printStackTrace();
34         }
35     }
36
37     public List<StokContractComplex> GetAllStok() {
38         List<StokContractComplex> dataContract = new ArrayList<StokContractComplex>();
39         Connection connection = getConnection();
40         StokContractComplex contract;
41         try {
42             Statement statement = connection.createStatement();
43             ResultSet resultSet = statement.executeQuery("SELECT stok.Id, AdiSoyadi, Adi, Adet, stok.Tarih FROM stok"
44                                                         + " LEFT JOIN urunler ON stok.UrunId = urunler.Id"
45                                                         + " LEFT JOIN personel ON stok.PersonelId = personel.id ORDER BY stok.Id DESC");
46             while (resultSet.next()) {
47                 contract = new StokContractComplex();
48                 contract.setId(resultSet.getInt("Id"));
49                 contract.setPersonelAdi(resultSet.getString("AdiSoyadi"));
50                 contract.setUrunAdi(resultSet.getString("urunler.Adi"));
51                 contract.setTarih(resultSet.getString("Tarih"));
52                 contract.setAdet(resultSet.getInt("Adet"));
53             }
54         }
55     }
56 }
```

FrontEnd Sınıflarına Ait Örnekler;

```
public class KategoriEkleFE extends JDialog implements FeInterfaces {  
    public KategoriEkleFE() {  
        initPencere();  
    }  
  
    @Override  
    public void initPencere() {  
        JPanel panel = initPanel();  
        panel.setBorder(BorderFactory.createTitledBorder("Kategori Ekle"));  
  
        add(panel);  
        setTitle("Kategori Ekle");  
        pack();  
        setModalityType(DEFAULT_MODALITY_TYPE);  
        setLocationRelativeTo(null);  
        setVisible(true);  
        setDefaultCloseOperation(HIDE_ON_CLOSE);  
    }  
  
    @Override  
    public JPanel initPanel() {  
        JPanel panel = new JPanel(new GridLayout(3, 2));  
  
        JLabel kategoriAdiLabel = new JLabel("Kategori Adı:", JLabel.RIGHT);  
        panel.add(kategoriAdiLabel);  
        JTextField adiField = new JTextField(10);  
        panel.add(adiField);  
        JLabel parentLabel = new JLabel("ParentId:", JLabel.RIGHT);  
        panel.add(parentLabel);  
        JTextField parentField = new JTextField(10);  
        panel.add(parentField);  
        JButton kaydetButton = new JButton("Kaydet");  
        panel.add(kaydetButton);  
        -- .. ..  
    }  
  
    JButton kaydetButton = new JButton("Kaydet");  
    ustPanel.add(kaydetButton);  
    JButton iptalbutton = new JButton("İptal");  
    ustPanel.add(iptalbutton);  
    kaydetButton.addActionListener(new ActionListener() {  
        @Override  
        public void actionPerformed(ActionEvent e) {  
            SatisContract contract = new SatisContract();  
            StokContract stokContract = new StokContract();  
            if (satisBox.getSelectedIndex() == 0 || personelBox.getSelectedIndex() == 0  
                || musteribox.getSelectedIndex() == 0 || satisUrunAdiBox.getSelectedIndex() == 0  
                || stokBox.getSelectedIndex() == 0 || satisAdetField.getText().matches("")) {  
                JOptionPane.showMessageDialog(null, "!!!Lütfen tüm bilgileri giriniz!!!");  
            } else {  
                SatisContractComplex sContract = (SatisContractComplex) satisBox.getSelectedItem();  
                MusteriContract mContract = (MusteriContract) musteribox.getSelectedItem();  
                UrunlerContract uContract = (UrunlerContract) satisUrunAdiBox.getSelectedItem();  
                PersonelContract pContract = (PersonelContract) personelBox.getSelectedItem();  
                StokContractComplex stokContractComplex = (StokContractComplex) stokBox.getSelectedItem();  
  
                SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd");  
                String date = format.format(satisTarihi.getDate());  
  
                stokContract.setId(stokContractComplex.getId());  
                stokContract.setPersonelId(pContract.getId());  
                stokContract.setUrunId(uContract.getId());  
                stokContract.setAdet(Integer.parseInt(satisAdetField.getText()));  
                stokContract.setTarih(date);  
  
                new StokDal().update(stokContract);  
  
                contract.setId(sContract.getId());  
                .. ..  
            }  
        }  
    });  
}
```

Core Sınıflarına Bir Örnek;

```
package tr.com.akarcesme.core;

import java.sql.DriverManager;

public class ObjectHelper extends CoreFields implements CoreInterfaces {

    static {

        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
        } catch (ClassNotFoundException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }

    public Connection getConnection() {

        Connection connection = null;
        try {
            connection= DriverManager.getConnection(getUrl(),getUserName(),getPassword());
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        return connection;
    }

}
```

Kullanılan Interface Örnekleri;

```
1 package tr.com.akarcesme.interfaces;
2
3 import javax.swing.JMenuBar;
4
5 public interface FeInterfaces {
6
7     public void initPencere();
8
9     public JPanel initPanel();
10
11     public JMenuBar initBar();
12
13     public JTabbedPane initTabs();
14 }
15
```

```
1 package tr.com.akarcesme.interfaces;
2
3 import java.util.List;
4
5 public interface DALInterfaces<T> {
6     void Insert(T entity);
7
8     List<T> GetAll();
9
10    T Delete(T entity);
11
12    void update(T entity);
13
14    List<T> GetById(int id);
15 }
16
```


Ana Menü Örneği;

```
//Stok/Satış Menü
JMenu stokVeSatisItem = new JMenu("Stok ve Satış İşlemleri");
bar.add(stokVeSatisItem);
JMenuItem stokDuzenleItem = new JMenuItem("Stok Düzenle");
stokVeSatisItem.add(stokDuzenleItem);
JMenuItem satisDuzenleItem = new JMenuItem("Satış Düzenle");
stokVeSatisItem.add(satisDuzenleItem);
stokDuzenleItem.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent e) {
        SwingUtilities.invokeLater(new Runnable() {

            @Override
            public void run() {
                new StokDuzenleFE();
            }
        });
    }
});

satisDuzenleItem.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent e) {
        SwingUtilities.invokeLater(new Runnable() {

            @Override
            public void run() {
                new SatisDuzenleFE();
            }
        });
    }
});
```

Uygulama Stoklar Ekranı;

Satış ve Stok Programı

Uygulama İşlemleri Ürünler Personel İşlemleri Müşteriler Stok ve Satış İşlemleri

Stoklar Satışlar

Stok İşlemleri

Ürün Adı: --Ürün Seçiniz--

Adet:

Stok Tarihi:

Stok Ekle Stok Yenile

Stok Toplam Ürün

Id	Ürün Adı	Personel Adı	Adet	Tarihi	Toplam
53	XPN100	Kutluhan	100	2021-10-18	
52	XCN100	Kutluhan	200	2021-10-18	
51	ACN100	Kutluhan	-5	2021-07-10	
50	BPL100	Kutluhan	-5	2021-07-10	
49	ICN100	Kutluhan	-10	2021-07-10	
48	BPL100	Kutluhan	-5	2021-07-10	
47	BPL100	Kutluhan	-5	2021-06-22	
46	ACN100	Kutluhan	-5	2021-06-22	
45	BPL100	Kutluhan	-10	2021-06-22	
44	BPL100	Kutluhan	20	2021-06-22	
43	BPL100	Kutluhan	-5	2021-06-22	
42	ACN100	Kutluhan	-5	2021-06-22	
41	ACN100	Kutluhan	-10	2021-06-22	
40	ACN100	Kutluhan	15	2021-06-22	
39	ICN100	Kutluhan	100	2021-06-22	
38	ACN100	Kutluhan	-15	2021-06-22	
37	ACN100	Kutluhan	15	2021-06-22	
36	BPL100	Kutluhan	-5	2021-06-20	
35	ICN100	Kutluhan	-5	2021-06-20	
34	BPL100	Kutluhan	-10	2021-06-17	
33	ACN100	Kutluhan	-5	2021-06-17	
32	BPL100	Kutluhan	100	2021-06-17	
31	ACN100	Kutluhan	-10	2021-06-17	
30	ACN100	Kutluhan	100	2021-06-17	

Uygulama Satışlar Ekranı;

Satış ve Stok Programı

Uygulama İşlemleri Ürünler Personel İşlemleri Müşteriler Stok ve Satış İşlemleri

Stoklar Satışlar

Id	Personel Adı	Müşteri Adı	Ürün Adı	Adet	Tarihi	Toplam
38	Kutluhan	Burak Karacan	ACN100	5	2021-07-10	
37	Kutluhan	FabioCassel	BPL100	5	2021-07-10	
36	Kutluhan	Ramzotti	ICN100	10	2021-07-10	
35	Kutluhan	Yakup	BPL100	5	2021-07-10	
34	Kutluhan	Yakup	BPL100	5	2021-06-22	
33	Kutluhan	Burak Karacan	ACN100	5	2021-06-22	
32	Kutluhan	Ramzotti	BPL100	10	2021-06-22	
31	Kutluhan	Yakup	BPL100	5	2021-06-22	
30	Kutluhan	Yakup	ACN100	5	2021-06-22	
29	Kutluhan	Burak Karacan	ACN100	10	2021-06-22	
28	Kutluhan	FabioCassel	ACN100	15	2021-06-22	
27	Kutluhan	Burak Karacan	BPL100	5	2021-06-20	
26	Kutluhan	Burak Karacan	ICN100	5	2021-06-20	
25	Kutluhan	Burak Karacan	BPL100	10	2021-06-17	
24	Kutluhan	FabioCassel	ACN100	5	2021-06-17	
23	Kutluhan	Ramzotti	ACN100	10	2021-06-17	

Satış İşlemleri

Müşteri Adı: --Müşteri Seçiniz--

Ürün Adı: --Ürün Seçiniz--

Adet:

Satış Tarihi:

Satış Yap Satış Yenile

Satış Adet Toplamı Satış Fiyat Toplamı

Ürün Ekleme Menüsü;

ACN100 Kutluhan 100

Ürün Ekleyin

Ürün Kayıt Alanı

Ürün Adı:


Kategori Seç: --Kategori Seçiniz--

Tarih Seç:

Fiyat Gir:

Kaydet İptal

Hesap Ekleme Menüsü;



Hesap Kayıt İşlemleri

Hesap İşlemleri

Personel Seç: Kutluhan ▼

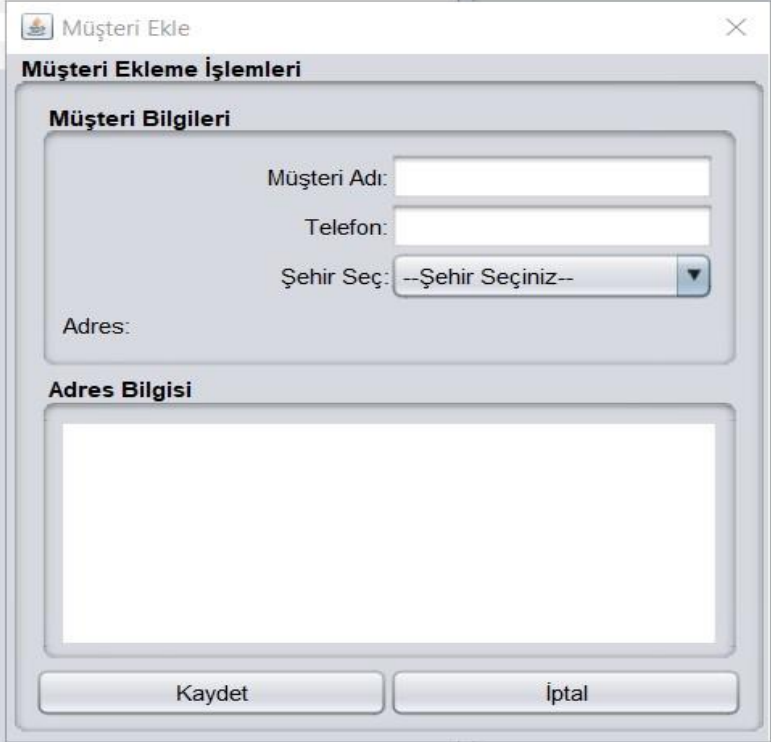
Yetki Seç: --Yetki Seçiniz-- ▼

Şifre Belirle:

Şifre Tekrar:

Kaydet İptal

Müşteri Ekleme Menüsü;



Müşteri Ekle

Müşteri Ekleme İşlemleri

Müşteri Bilgileri

Müşteri Adı:

Telefon:

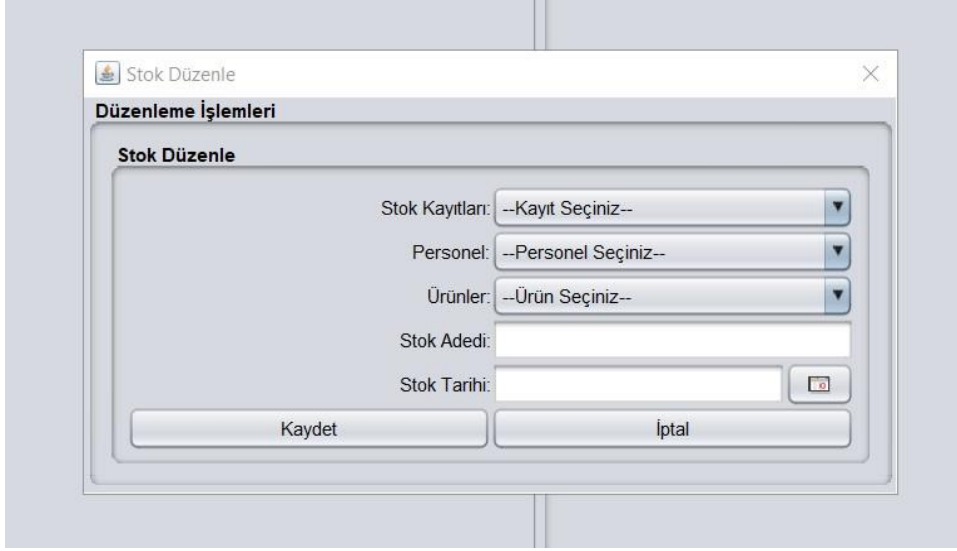
Şehir Seç: --Şehir Seçiniz-- ▼

Adres:

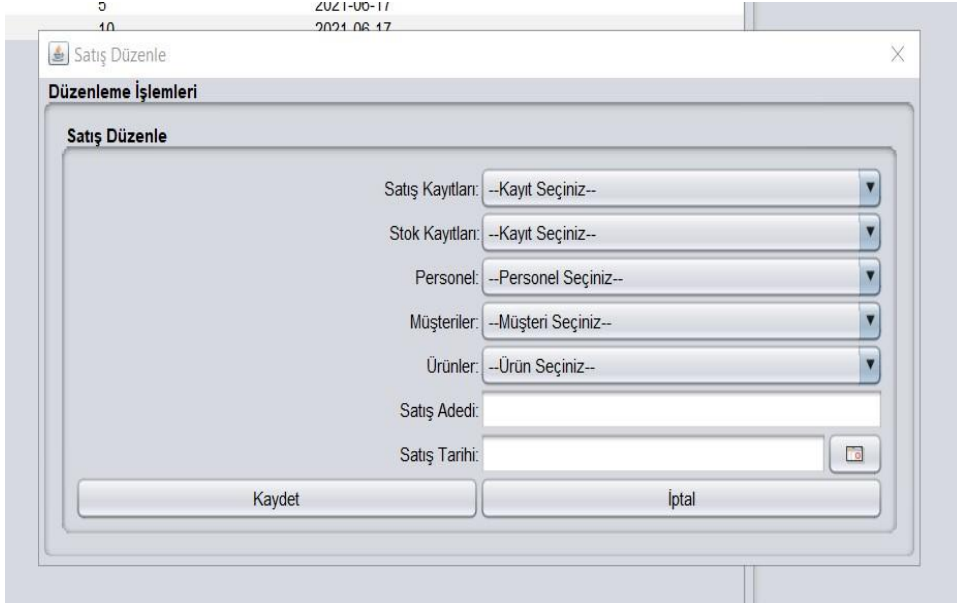
Adres Bilgisi

Kaydet İptal

Stok D zenleme Men s ;



Satı  D zenleme Men s ;



SONU 

Proje sonunda,ama  olarak belirtilen ve teknik  artnamede yer alan hedeflerin tamamı yerine getirilmi tir.Contract type sınıfları ile veritabanı ve kullanıcı aray z  ile ba lantı kurulmu tur.Veritabanında yer alan veriler ba lantı katmanında ki sınıflardan t retilen nesneler vasıtasıyla tutulmu  ve aray zde g sterilmi tir.Aray z  zerinde kullanıcı tarafından girilen veriler ise yine ba lantı katmanından t retilen nesneler ile veritabanına aktarılmı tır.B ylece katmanlı mimari yapısına uyulmaya  alı ılmı tır.Projede kısımda olsa design patternlerden de yararlanılmaya  alı ılmı tır.Maliyet raporu  er evesinde hareket edilmeye  alı ılmı  ve g rev da ılımı yapılarak ekip  alı masından yararlanılmı tır.