DÜZCE ÜNİVERSİTESİ
TEKNOLOJİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ

WEB API PLATFORMU TASARIMI

BMT332-BULUT BİLİŞİM
DR ÖĞR. ÜYESİ AHMET ALBAYRAK

NÖ 18742967422 ÖMÜRCAN SERDAR

İÇİNDEKİLER DİZİNİ

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

ŞEKİLLER

ÖZET

Bu makalede API tasarımı ve kullanımı hedeflenmiştir. Kullanıcı isteği (GET, POST, PUT, DELETE vb.) doğrultusunda, API sayfasında http durum kodları (200 OK, 201 CREATED, 400 BAD REQUEST vb.) tutulmuş, veritabanında işlemler yapılmış, veriler JSON veri formatına dönüştürülerek, isteğin geldiği kod bloğuna JSON formatında veri iletimi gerçekleştirilmiştir. Son olarak API sayfasında header ayarlanmıştır. İsteğin geldiği ilgili kod bloğunda, API sayfasından iletilen verilerden elde edilen silindi, güncellendi veya eklendi mesajları sonucunda aktif pencerede işlemler gerçekleştirilmiştir. Makalede, örnekler üzerinden anlatım gerçekleştirilmiş ve makalenin "Sonuçlar" kısmında çalışmanın sonucu ifade edilmiş, önerilerde bulunulmuştur.

Anahtar Kelimeler: api, api tasarım, api kullanım

ABSTRACT

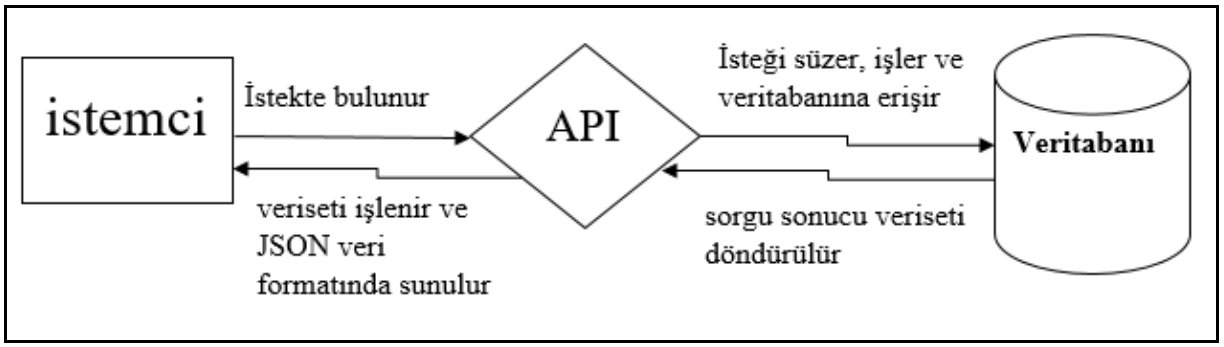
In this article, API design and usage is aimed. In accordance with the user request (GET, POST, PUT, DELETE etc.), http status codes (200 OK, 201 CREATED, 400 BAD REQUEST etc.) are kept in the API page, transactions are made in the database, the data is converted to JSON data format, JSON is sent to the code block where the request comes. Data transmission was performed in the format. Finally, the header has been set on the API page. In the relevant code block where the request came from, operations were performed in the active window as a result of the deleted, updated or added messages obtained from the data transmitted from the API page. In the article, narration was made through examples, and the result of the study was expressed in the "Results" section of the article and suggestions were made.

Keywords: api, api design, api use

API (Application Programming Interface)

Uygulama Geliştirme Arayüzü anlamına gelen API sayesinde yazılım geliştiricileri, ellerindeki verileri istedikleri sınırlılıkta dış dünyayla paylaşabilmekte ve bu paylaşım sürecinde tüm kontrolleri ellerinde tutabilmektedir.

Son yıllarda internete erişim olanağı çok farklı platformlar tarafından sağlandığı için kullanıcıların ihtiyaçlarını sadece web siteleri karşılayamaz olmuştur. Bu platformlara bilgisayarlar, tabletler ve özellikle birbirinden farklı yetenekler sergileyen mobil telefonlar örnek olarak verilebilir. Haliyle uygulama geliştiricileri platformdan bağımsız bir şekilde kullanıcılara ulaşabilmek için uygulamadan da bağımsız API geliştirmeye yönelmektedirler. Geliştirilen bu API'ler tüm platform ve uygulamaların okuyup, anlamlandırabileceği XML ya da JSON veri tipini kullanarak hizmet sunmakta ve bu şekilde tüm bağımlılıklardan kendilerini arındırarak geliştiricilerin amaçları doğrultusunda son kullanıcılara hizmet verebilmektedirler. Bu süreç Sekil 1'de gösterilmiştir.



Şekil 1: API Blok Şema

Örnek olarak bilgim sayfası üzerinde "bilgiGoster" isminde bir buton olduğu ve butona basıldığında 61 id parametresiyle api/user sayfasına erişip kullanıcıya ad, soyad ve adres gibi bilgilerin kullanıcıya sunulduğu düşünülsün. Bu senaryo için sözde kod blokları aşağıdaki gibidir.

api/user sayfası için;

```
if ( REQUEST_METHOD ==”GET” ){
        db_bilgiler=db->query('select * from user where id=GET[“id”]')->fetch()
        jsonKume=array();
        jsonKume[“bilgiler”]=db_bilgiler
        echo json_encode(jsonKume);
}
```

bilgim sayfası için ;

<button> bilgiGoster </button>

if(bilgiGoster.click()) {

GET : api/user?id=61,

success:function (donenveri) {

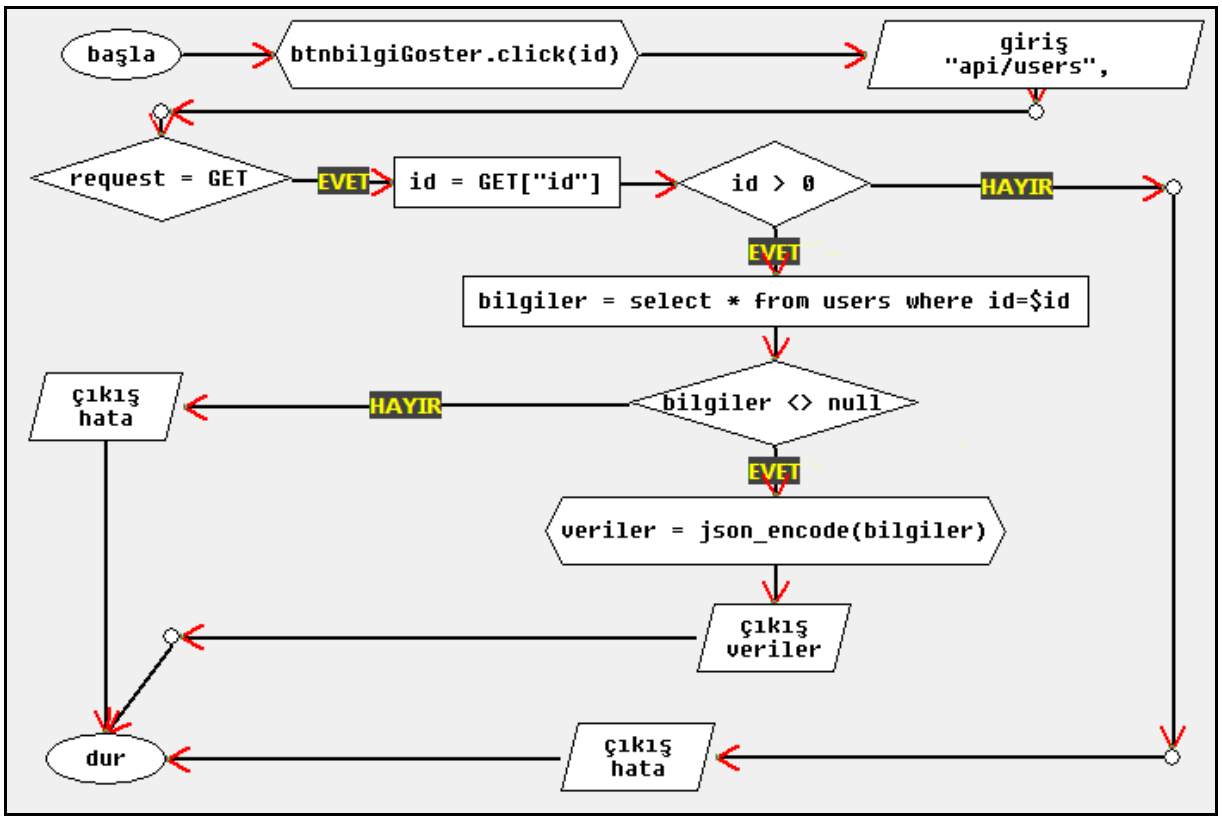
alert(donenveri.bilgiler.ad) },

error:function (donenveri){

alert("hata")}

}

Bu senaryo için akış diyagramı Şekil 2'de gösterilmiştir.



Şekil 2: API Senaryo Akış Diyagramı

Bu diyagram dahilinde; kullanıcı istekte bulunacak, API bu isteğin istek modu (GET, POST vb.) durumuna ve gelen/istenen parametre dahilinde işlemler yapacak,yapılan işlemler sonrasında veritabanında sorgulamalar yapacak, sorgu sonucunda bir veri grubu elde edecek, daha sonra bu veri grubunu JSON veri formatında kullanıcı isteğinin geldiği yere yazdırılacak, döndürülecek . Yazdırılan / döndürülen JSON veri formatı ile kullanıcıya istenilen veri grubu ya da hata mesajı gösterilebilecek. Sonuç olarak farklı kaynaklar kullanılarak özel uygulamalar ve paylaşım sağlanmış olacak.

API KULLANIMI (api.omurserdar.com)

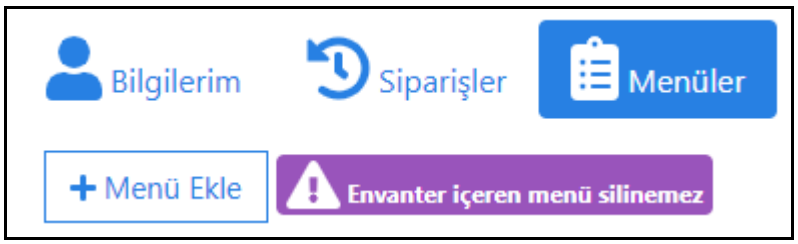
Geniş bir kitle tarafından kullanılan, HTML içine gömülebilen bir betik dil olan PHP ve açık kaynaklı bir ilişkisel veritabanı yönetim sistemi olan MySQL kullanılarak api.omurserdar.com adresinde API tasarımı gerçekleştirilmiştir. Tasarımın amacı, yiyecek/içecek siparişi verebilmek ya da yiyecek/içecek satışı yapabilmektir. Bu amaç doğrultusunda iki adet üyelik/giriş tipi mevcuttur; bireysel ve kurumsal. Bireysel kullanıcının yapabilecekleri;

- hesap bilgilerini görüntüleyebilecek,
- sepet sistemi ile sepetini görüntüleyebilecek, sepetine envanter (yiyecek ya da içecek) ekleyebilecek ya da kaldırabilecek, sepetini onaylayıp sipariş verebilecek,
- sipariş detaylarını ve hangi durumda olduğu bilgisini görüntüleyebilecektir.

Kurumsal kullanıcının yapabilecekleri;

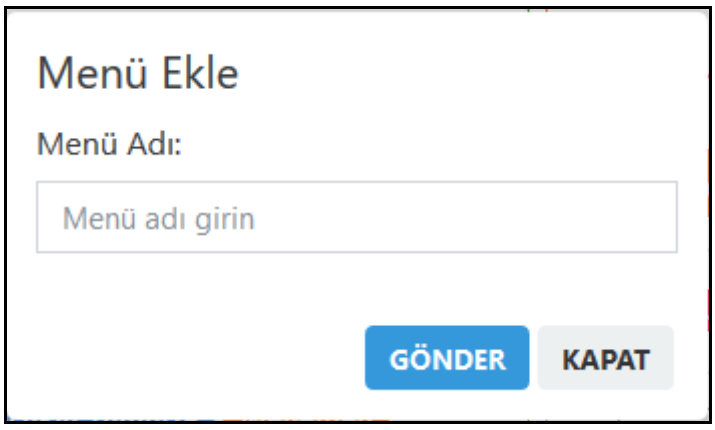
- hesap bilgilerini görüntüleyebilecek, düzenleyebilecek,
- siparişlerin detaylarını görüntüleyebilecek ve durum güncelleyebilecek,
- envanter ekleyebilecek, görüntüleyebilecek, düzenleyebilecek ve silebilecek,
- menüler (kategori) ile, menüleri ve menülere bağlı envanterleri görüntüleyebilece, ve düzenleyebilecek, eğer menüde envanter bulunmuyorsa menüyü silebilecektir

Menü eklemek için kurumsal kullanıcı sisteme giriş yaptıktan sonra Şekil 3'de gösterilen "Menüler" tab pencereside bulunan "Menü Ekle" butonuna basması gerekmektedir.



Şekil 3 : Menüler Tab Panel : Menü Ekleme

Butona basıldıktan sonra açılacak pencere (confirm) Şekil 4'de gösterilmiştir.



Şekil 4:Menü Ekleme Penceresi

Eğer “GÖNDER” butonuna basıldığında, input boş bırakılırsa ya da boş geçilirse herhangi bir işlem gerçekleşmeyecek ve geriye false değer döndürecek. Buraya kadar olan işlemleri gerçekleyen kod parçacıkları Şekil 5’de verilmiştir.

```
<script type="text/javascript">
 $(document).ready(function(){
//tabmenü ekle js
 $('.btnmenuekle').click(function(){
      $.confirm({
    title: 'Menü Ekle',
    content: '<form action="" class="formName">' +
    '<div class="form-group"><label>Menü Adı: </label>' +
    '<input type="text" placeholder="Menü adı girin" class
        ="ekletabad form-control" required />' +
    '</div></form>',
    buttons: {
        formSubmit: {
            text: 'Gönder',
            btnClass: 'btn-blue',
            action: function () {
                var kurid=$(".kid").attr("data-id");
                var ekletad=$('.ekletabad').val().trim();
                 if(!ekletad){
                    $.dialog('Menü adı boş bırakılamaz! ');
                    return false;
```

Şekil 5: Menü Ekleme Form ve Uyarı Kodları

Eğer “ekletabad” adındaki class seçicisi (input), değer içeriyorsa, bu değeri ve giriş yapan kurumsal kullanıcın $_SESSION[“kullanici_id”] adlı session bilgisinden gelen id değerini api/tabmenu sayfasına jquery ajax fonksiyonu ile POST edecek. Bu işlem için kod parçacığı Şekil 6’da verilmiştir.

```
 $.ajax({
method: 'POST',
url : "/api/tabmenu/",
data : {kid:kurid,ad:ekletad},
```

Şekil 6: Menü Ekleme Ajax ile API sayfasına POST isteği gönderme

Bu kod parçacığı ile api/tabmenu/ sayfasına POST isteğinde bulunulmuş ve parametre olarak id ile ad (kullanıcı session id ve ekletabad input) değerleri gönderilmiş olunur. api/tabmenu/ sayfasında ise gelen istek moduna göre işlemler yapılır. POST isteği geldiğinde, veritabanının tabMenu tablosuna, gelen id ve ad değerleri ile yeni kayıt eklenmeye çalışılacak, eğer eklenen bir kayıt olursa created durum kodu (201) olarak tutulacak ve bilgiler diziye eklenecek, ekleme olmazsa bad request durum kodu (400) olarak tutulacak ve bilgiler bir diziye eklenecek. Sonrasında header, tutulan durum kodu ile ayarlanacak durum kodu diziye eklenecek. Son olarak dizi json veri formatına dönüştürülerek yazdırılacak. Header ayarlama, durum kodunun diziye eklenmesi ve dizinin yazdırılması ortak işlemler olduğu için bundan sonraki örneklerde açıklanmayacaktır. İşlemleri gerçekleyen kod parçacıkları Şekil 7’de verilmiştir.

```php
$jsonArray = array();
$jsonArray["hata"] = FALSE;
$httpKOD = 200;
$istekMOD = $_SERVER["REQUEST_METHOD"];

if($istekMOD=="POST") {
    if(isset($_POST["kid"])&&isset($_POST["ad"])){
 $tabekle=$db->prepare("INSERT INTO tabMenu set kurumsal_id=:pkid,ad=:pad");
 $tabekle->execute(array('pkid' => $_POST["kid"],"pad" => $_POST["ad"]));
 $tabsay=$tabekle->rowCount();
         if($tabsay==1){
             $httpKOD = 201; //CREATED
             $jsonArray["eklenen_tab_id"]=$db->lastInsertId();
             $jsonArray["mesaj"] = "eklendi";
         }
         else{
             $httpKOD = 400; //BAD Request
             $jsonArray["hata"] = TRUE;
             $jsonArray["mesaj"] = "eklenmedi";
         }
    }
}
SetHeader($httpKOD);
$jsonArray[$httpKOD] = HttpStatus($httpKOD);
echo json_encode($jsonArray);
```

Şekil 7: Menü Ekleme - api/tabmenu/ sayfası POST istek bloğu kod parçacığı

Şekil 7'deki işlemler sonrasında isteğin geldiği yere (Şekil 6) json formatına çevrilmiş bilgiler iletilecektir. İletilen bu bilgiler dahilinde aktif olan ekrana bilgi mesajları gösterilebilecek ve eklenen kayıt, aktif sayfaya dahil edilebilecektir.

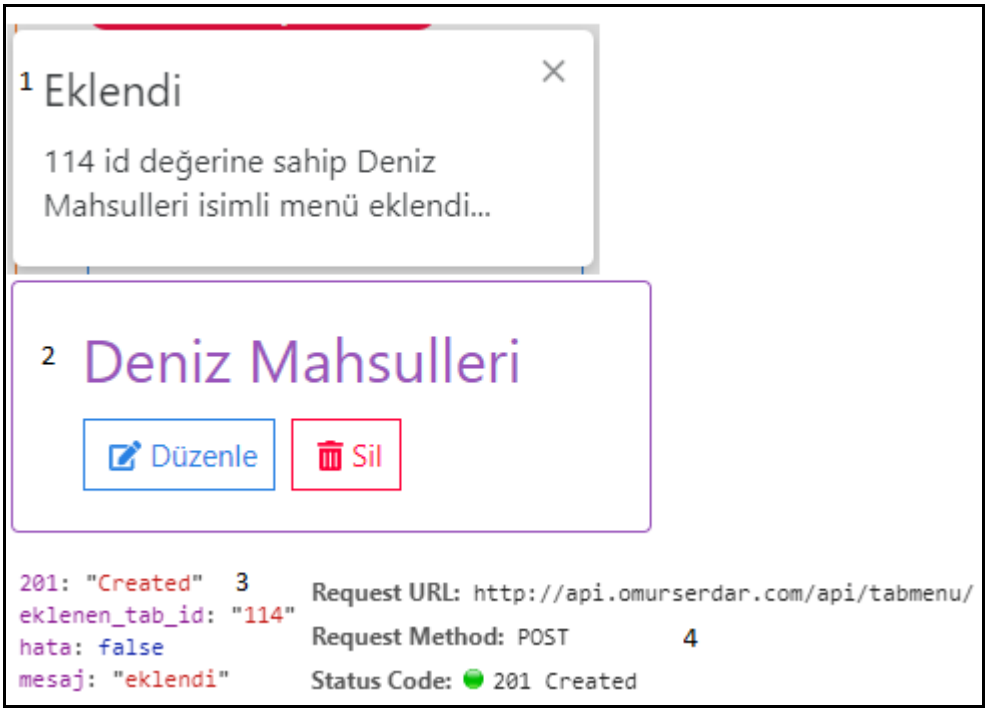
```javascript
            success: function(apitabekle){
                if(apitabekle.mesaj=="eklendi"){
                   var eklenenid=apitabekle.eklenen_tab_id;
                   var ekletabMenuBos="";
ekletabMenuBos+='<div id="cardtab'+eklenenid+'" class="card
    border-info mb-2 ml-2 col-md-4" style="max-width: 20rem
    ;"><div class="card-body">';
ekletabMenuBos+='<h2 class="card-title text-info" id="ma'
    +eklenenid+'"><b>'+ekletad+'</b></h2>';
ekletabMenuBos+='<button type="button" id="'+ekletad+'" data
    -id="'+eklenenid+'" class="btntabduzenle btn btn-outline
    -primary pr-2 mr-2"><i class="fas fa-edit"></i> Düzenle
    </button>';
ekletabMenuBos+='<button type="button" id="'+ekletad+'" data
    -id="'+eklenenid+'" class="btntabsil btn btn-outline
    -danger btn'+eklenenid+' pr-2 mr-2"><i class="fas fa
    -trash-alt"></i> Sil</button> </div></div>';
$(".sipmenu2Div").append(ekletabMenuBos).fadeIn("normal");
            $.dialog({
                title: 'Eklendi',
                content: eklenenid+' id değerine sahip '
                    +ekletad+' isimli menü eklendi...'
            });
                }
                else{ $.dialog("eklenemedi"); }
```

Şekil 8: Menü Ekleme - API sayfasından gelen cevaba göre gerçekleşen işlemler

Şekil 9'da;

- 1 numaralı görselde, eklendi mesajı,
- 2 numaralı görselde, eklenen menü adı ve menüye ait butonlar,
- 3 numaralı görselde, api/tabmenu sayfasından json olarak iletilen veriler,
- 4 numaralı görselde, api/tabmenu sayfasının post işlemi neticesinde ayarlanan header durumu

gösterilmiştir.



Şekil 9: Menü Ekleme - Ekleme sonucunda gerçekleşen işlemler

Menü güncellemek için Şekil 9'daki 2 numaralı görselde bulunan "Düzenle" butonuna basılması gereklidir. Bu butona basıldığında menü düzenleme penceresi (confirm) gösterilir. Bu işlemi gerçekleştiren kod bloğu Şekil 10'da verilmiştir.

```
            //tabuzenle click
$(document).on('click', '.btntabduzenle', function(){
  var tad=$(this).attr("id");
  var tid=$(this).attr("data-id")
  $.confirm({
  title: 'Menü Düzenleme',
  content: '' +
  '<form action="" class="formName">' +
  '<div class="form-group"><label>Menü Adı: </label>' +
  '<input type="text" value="'+tad+'" class="xtabad form
      -control" required />' +
  '</div></form>',
```

Şekil 10: Menü Düzenleme Penceresi Kod Bloğu

Gösterilen penceredeki "GÖNDER" butonuna basıldığında, "Düzenle" butonunun id ve data-id seçici alanlarından menü adı ve menü id bilgileri elde edilir. Elde edilen bilgilerden sonra

açılan penceredeki xtabad seçici ismindeki input boş bırakılırsa, boş geçilirse ya da menü adında değişiklik yapılmazsa herhangi bir işlem gerçekleşmeyecek ve geriye false değer döndürülecek. Eğer değişiklik yapılmışsa api/tabmenu/ sayfasına PUT isteğinde bulunulacak ve istekle beraber butondan elde edilen id değeri ve düzenleme formundan elde edilen menü adı parametreleri json formatında gönderilecek. Bu işlem Şekil 11'deki gibi kodlanmıştır.

```
buttons: {
    formSubmit: {
        text: 'Gönder',
        btnClass: 'btn-blue',
        action: function () {

            var xad=$('.xtabad').val().trim();
             if(!xad){
                $.dialog('Menü adı boş geçilemez');
                return false;
            }
            if(xad==tad){
                $.dialog('Herhangi bir değişiklik
                yapılmadı');
                return false;
            }
    var dobje={"id":tid,"ad":xad};
    dobje=JSON.stringify(dobje);
         $.ajax({
        method: 'PUT',
        url : "/api/tabmenu/",
        data : dobje,
```

Şekil 11: Menü Düzenleme Formu, Uyarı Mesajları ve PUT İstek Gönderimi

İstek, api/tabmenu/ safyasında süzülecek ve gelen ham veriler json formatından çözülecektir. Çözülen id değerine göre menü varsa güncelleme sorgusu yapılacak ve başarılı olma / olmama durumlarına göre, bilgi gösterimi için mesaj içerikleri, iletilecek diziye eklenecek. Son olarak header ayarlanacak ve iletilecek olan dizi json formatına dönüştürüldükten sonra iletim gerçekleşecektir. Bu işlem Şekil 12'deki gibi kodlanmıştır.

```
else if($istekMOD=="PUT") {
 $gelenler=json_decode(file_get_contents("php://input"));
    if($db->query("select * from tabMenu where id='$gelenler->id'")->rowCount
        ()==0){
        $httpKOD = 400;
        $jsonArray["hata"] = TRUE;
        $jsonArray["hataMesaj"] = "tabMenu bulunamadı";
        }
    else{
        $tabguncelle=$db->prepare("UPDATE tabMenu SET ad=:pad WHERE id=:pid");
        $tabguncelle->execute(array("pad" => $gelenler->ad,"pid" => $gelenler
            ->id));
            if($tabguncelle->rowCount()>0){
                $httpKOD = 200;
                $jsonArray["mesaj"] = "güncellendi";
            }
            else{
                $httpKOD = 400;
                $jsonArray["hata"] = TRUE;
                $jsonArray["hataMesaj"] = "güncellenmedi";
                }
        }
}
```

Şekil 12: Menü Düzenleme - api/tabmenu/ sayfası PUT İsteği

İsteğin süzüldüğü sayfa olan api/tabmenu/ sayfasından iletilen veriler sonucunda kullanıcıya bilgi gösterimi ve güncellemenin başarılı olması durumunda menü adının yeni adıyla html kodlarında güncellenmesi gerçekleştirilecektir.Son olarak "Düzenle" butonunun id değerinde bulunan eski menü adı, yeni menü adıyla güncellenecektir. Bu işlemleri gerçekleştiren kod blogu Şekil 13'de verilmiştir.

```
 $.ajax({
method: 'PUT',
url : "/api/tabmenu/",
data : dobje,
success: function(apitabduz){
    if(apitabduz.mesaj=="güncellendi"){
        $("#ma"+tid).html("<b>"+xad+"</b>");
        $('#'+tad).attr("id",xad);
        $('.btn'+tid).attr("id",xad);
$.dialog({
    title: 'Güncellendi',
    content: 'Menü Adı Başarıyla Güncellendi'
});
    }
}
 })
```

Şekil 13: Menü Düzenleme- API sayfasından Gelen Cevaba göre Gerçekleşen İşlemler

Şekil 14'de;

- 1 numaralı görselde, kullanıcının gönderdiği parametreler json veri formatında,
- 2 numaralı görselde, istek metodu, isteğin adresi ve http durum kodu,
- 3 numaralı görselde, api/tabmenu sayfasından json olarak iletilen veriler,
- 4 numaralı görselde, api/tabmenu sayfasından iletilen bilgiye göre mesaj,
- 5 numaralı görselde, kullanıcın aktif sayfasında güncellenen menünün adı ve butonları gösterilmiştir.

Şekil 14: Menü Düzenleme – Düzenleme Sonucunda Gerçekleşen İşlemler

Menü silmek için, Şekil 14'deki 5 numaralı görselde bulunan "Sil" butonuna basıldığında menü silme penceresi (confirm) gösterilir. Menü silme penceresi Şekil 15'de verilmiştir.

Şekil 15:Menü Silme Penceresi

Gösterilen pencerede "EVET, SİL" butonuna basıldığında, id parametresi ile ("Sil" butonuna basıldığında, data-id değerinden gelen, menünün id bilgisi) api/tabmenu sayfasına DELETE isteğinde bulunulacak. İşlem, Şekil 16'da gösterilmiştir.

```
//tabmenü sil
    $(document).on('click', '.btntabsil', function(){
      var tabadi=$(this).attr("id");
      var tabid=$(this).attr("data-id");
      $.confirm({
    title: 'Silmek istediğine emin misin?',
    content: tabadi+' isimli menü silinecek...',
    buttons: {
        evet: {
            text: 'Evet, sil',
            btnClass: 'btn-danger',
            action: function () {
                    $.ajax({
                    method: 'DELETE',
                    url : "/api/tabmenu?id="+tabid,
```

Şekil 16 - Menü Silme - DELETE İsteği Gönderme

İsteğin gittiği sayfada (api/tabmenu) yapılacak DELETE işlemi Şekil 17'de verilmiştir.

```
else if($istekMOD=="DELETE") {
 if(isset($_GET["id"]) && !empty(trim($_GET["id"]))){
       $id=$_GET["id"];
 $tabVarMi = $db->query("select * from tabMenu where id=$id");
 if($tabVarMi->rowCount()==1){
  $tabsil = $db->query("delete from tabMenu where id=$id");
          if($tabsil->rowCount()==1){
           $httpKOD = 200;
           $jsonArray["mesaj"] = "silindi";
         }else{
         $httpKOD = 400;
         $jsonArray["hata"] = TRUE;
         $jsonArray["mesaj"] = "silinemedi";
         }
    }else {
    $httpKOD = 400;
    $jsonArray["hata"] = TRUE;
    $jsonArray["mesaj"] = "kayıt bulunamadı";
    }
 }else {
 $httpKOD = 400;
 $jsonArray["hata"] = TRUE;
 $jsonArray["mesaj"] = "tabMenu id değerini gönder";
 }
}
```

Şekil 17 : Menü Silme – API (api/tabmenu) sayfasında DELETE İsteği İşlemleri

API (api/tabmenu) sayfasında DELETE işlemi dahilinde id değeri kontrol edilecek, id değeri varsa id değerinin eşit olduğu olduğu kayıt veritabanında aranacak, arama sonucunda eşleşen kayıt bulunursa silme sorgusu çalıştırılacak ve silme sorgusu sonucunda mesaj bilgisi diziye eklenecek. Son olarak HTTP durum kodu ile header ayarlanacak ve isteğin geldiği yere (sayfaya) bilgiler, json formatında iletilecek.

İletilen veriler sonucunda "silindi" mesajı dönmüş ise menü bilgileri (menü adı ve butonları içeren bootstrap card) aktif sayfadan jquery fonksiyonları ile 1 saniyede kaldırılacak. Bu işlemleri gerçekleştiren kod bloğu Şekil 18'de verilmiştir.

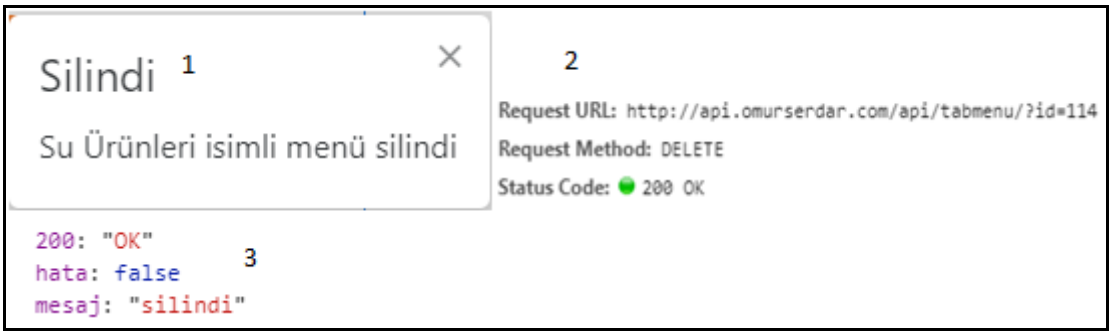
```
success: function(apitabsil){
    if(apitabsil.mesaj=="silindi"){
      $("#cardtab"+tabid).slideUp(1000, function() {
        $(this).remove();
       });
      $.dialog({ title: 'Silindi', content: tabadi+'
          isimli menü silindi'
            });
      }
    }
```

Şekil 18 : Menü Silme - İstek Sonucunda Çalışacak Kod Blokları

Şekil 19'da;

- 1 numaralı görselde, kullanıcıya gösterilen "silindi" bilgisi,
- 2 numaralı görselde, istek metodu, isteğin adresi ve http durum kodu,
- 3 numaralı görselde, api/tabmenu sayfasından json olarak iletilen veriler,

gösterilmiştir.



Şekil 19: Menü Silme – Silme Sonucunda Bilgi Gösterimi ve Header

Şekil 20'de, "Menüler" ismindeki tab panel ile, kendisine ait envantere sahip olan ve olmayan tüm menüleri görüntüleyebilmek için api/kurumsal/ sayfasına, id (session) parametresi ile GET isteğinde bulunma durumu kod bloğu gösterilmiştir.

```
$(document).ready(function(){
  $('.nav-pills a[href="#pills-kurtabMenuler"]').click(function(){
    var kurid=$(".kid").data("id");
    $.ajax({
        method: 'GET',
        url : "/api/kurumsal",
        data : {id:kurid},
        contentType: "application/json",
```

Şekil 20 : Menüler – GET isteği

GET isteği api/kurumsal sayfasına yapılacak, bu sayfada (api/kurumsal) istek metodu süzülecek ve işlenecek. Gönderilen id değeri ile veritabanında eşleşen kayıt bulunursa kullanıcıya ait menülerin adları ve menülere ait envanter sayısı için veritabanında sorgulama yapılacak. İşlemlere ait kod bloğu Şekil 21'de verilmiştir.

```
else if($istekMOD=="GET"){
 if(isset($_GET["id"]) && !empty(trim($_GET["id"]))){
   $kid=$_GET["id"];
   $kurVarMi = $db->query("select * from kurumsal where id='$kid'");
   if($kurVarMi->rowCount()>0){
       $jsonArray["kurumsalTab"]=array();
       $jsonArray["kurumsalEnv"]=array();

       $tabAdveSayi=$db->prepare("select tabMenu.id,tabMenu.ad,COUNT(tabMenu.id
           ) as envsayi FROM envanter,tabMenu,kurumsal
            WHERE envanter.tabMenu_id=tabMenu.id AND
            kurumsal.id=envanter.kurumsal_id AND
            tabMenu.kurumsal_id=kurumsal.id AND
            kurumsal.id=?
            GROUP BY tabMenu.ad
            ORDER BY tabMenu.id");
           $tabAdveSayi->execute(array($kid));
```

Şekil 21: Menüler - API sayfasında GET İşlemleri -1 (kurumsala ait menü bilgisi)

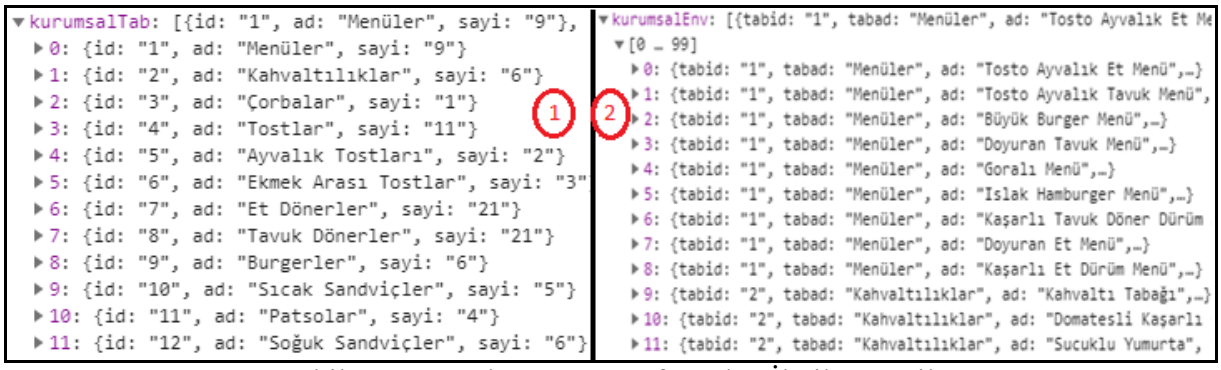
Bu işlemler devamında while döngüsü ile tabAdveSayi sorgusundan dönen her kayıt, kurumsalTab dizisine eklendikten sonra yeni bir sorgu ile menülere ait envanterlerin bilgileri sorgulanacak ve döngüye yeni bir while döngüsü eklenerek her bir envanter bilgisi kurumsalEnv dizisine eklenecek. Diziler, http durum kodu ve mesaj bilgisi json formatına çevrilip isteğin geldiği yere iletilecek ve header ayarlanacak. Şekil 22'de işlemler gösterilmiştir.

```
while($cek=$tabAdveSayi->fetch(PDO::FETCH_ASSOC)){
 $ptmid=$cek["id"];$ptmad=$cek["ad"];$ptmes=$cek["envsayi"];
 $kurumsaltabeklenecek=array("id"=>$ptmid,"ad"=>$ptmad,"sayi"=>$ptmes);
 array_push($jsonArray["kurumsalTab"],$kurumsaltabeklenecek);

 $tabenv=$db->prepare("select envanter.ad,envanter.tanim,fiyat,alinabilirMi
     ,envanter.id FROM envanter,tabMenu,kurumsal
WHERE envanter.tabMenu_id=tabMenu.id AND kurumsal.id=envanter.kurumsal_id AND
tabMenu.kurumsal_id=kurumsal.id AND kurumsal.id=? AND tabMenu.id=?");
$tabenv->execute(array($kid,$ptmid));
    while($cekenv=$tabenv->fetch(PDO::FETCH_ASSOC)){
$tid=$cek["id"]; $tad=$cek["ad"]; $ead=$cekenv["ad"]; $etan=$cekenv["tanim"];
    $efi=$cekenv["fiyat"]; $eal=$cekenv["alinabilirMi"]; $envid=$cekenv["id"];
    $kurumsalenveklenecek=array("tabid"=>$tid,"tabad"=>$tad,"ad"=>$ead,"tanim"
        =>$etan,"fiyat"=>$efi,"alim"=>$eal,"envanterid"=>$envid );
        array_push($jsonArray["kurumsalEnv"],$kurumsalenveklenecek);
            } //while menüenv
      } //while menüadvesayi
      $httpKOD = 200;
            }else {
                $httpKOD = 400;
                $jsonArray["hata"] = TRUE;
                $jsonArray["hataMesaj"] = "Üye bulunamadı";
                   }
  }else {
     $httpKOD = 400;
     $jsonArray["hata"] = TRUE;
     $jsonArray["hataMesaj"] = "kurumsal id gönder";
 }
}
```

Şekil 22 : Menüler - API sayfasında GET İşlemleri -2 (menüye ait envanter bilgisi)

Şekil 23'de; 1 numaralı görselde, api sayfasından iletilen kurumsalTab dizisi, 2 numaralı görselde, api sayfasından iletilen kurumsalEnv dizisi gösterilmiştir.



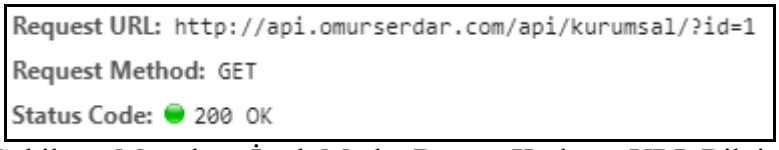
Şekil 23 : Menüler - API Sayfasından İletilen Veriler

API sayfasından iletilen bilgiler neticesinde, Şekil 20'deki kod bloğundan devam edilecek ve iletilen bilgiler sonucunda iç içe döngü ile hem menü bilgileri hem de menülerin barındırdığı envanterler, kullanıcının aktif penceresine html olarak eklenecektir. Bu işlemleri gerçekleştiren kod bloğu Şekil 24'de verilmiştir.

```
success: function(dataTabMenu){
    var renkler=["success","primary","info","danger","warning"];
    if(dataTabMenu.hata==false){
        var metin="<div class='row'>";
        for (i in dataTabMenu.kurumsalTab){
            metin+='<div class="card border-'+random_item(renkler)+' mb
                -2 ml-2 col-md-4" style="max-width: 20rem;"><div class
                ="card-body">';
            metin+='<h2 class="card-title text-'+random_item(renkler)+'"
                id="ma'+dataTabMenu.kurumsalTab[i].id+'"><b>'
                +dataTabMenu.kurumsalTab[i].ad+'</b></h2>';
            metin+='<h2> <span class="badge badge-pill badge-'
                +random_item(renkler)+'">'+dataTabMenu.kurumsalTab[i]
                .sayi+' adet envanter</span></h2><hr/>';
                for(j in dataTabMenu.kurumsalEnv){
                    if(dataTabMenu.kurumsalTab[i].id==dataTabMenu
                        .kurumsalEnv[j].tabid)
                            metin+='<span style="font-size: 14px;"
                            class="mb-1 mr-1 badge badge-pill badge
                            -'+random_item(renkler)+'">'+dataTabMenu
                            .kurumsalEnv[j].ad+'</span>';
                }//for j
            metin+='<hr/><button type="button" id="'+dataTabMenu
                .kurumsalTab[i].ad+'" data-id="'+dataTabMenu
                .kurumsalTab[i].id+'" class="btntabduzenle btn btn
                -outline-primary pr-2 mr-2 btn-block"><i class="fas fa
                -edit"></i> Düzenle</button></div></div>';
            }//for i
        $(".sipmenuDiv").html(metin+"</div>");
```

Şekil 24 : Menüler - API Sayfasından İletilen Veriler ile İşlem Yapma

Şekil 25'de istek modu, istek sayfası ve durum kodu bilgileri verilmiştir.



Şekil 25: Menüler - İstek Modu, Durum Kodu ve URL Bilgisi

Pencereye (kullanıcının aktif sayfası) eklenecek HTML çıktısı bilgisi Şekil 24'de verilmişti. İfade edilen html çıktısına ait verilerden bir kısmı (menü adları, menüye bağlı envanterler ve butonlar) Şekil 26'da verilmiştir.

Şekil 26: Menüler - GET İsteği Sonucunda Sayfaya Eklenen Menüler ve Envanterler

SONUÇLAR

Bu çalışmada API tasarımı ve kullanım örneği açıklanmıştır. Kullanılan teknolojiler; betik kodlama dili olarak PHP, veritabanı yönetim sistemi olarak MySQL, tasarım için Bootstrap, tarayıcı üzerindeki işlemler için Javascript ve Javascript kütüphanesi olarak Jquery teknolojileridir. API sayfalarına istekte bulunmak için Jquery kütüphanesinin ajax fonksiyonu kullanılmış ve bu sayede api sayfasından gelen cevaba göre aktif sayfaya tasarım elemanları eklenebilmiş, güncellenebilmiş, silinebilmiş ve kullanıcıya bilgi mesajları iletilebilmiştir. Güvenlik önlemlerinin alınması, eş zamanlılık için soket programlama, veritabanının ileri seviyede yapılacak işlemler için düzenlenmesi ve rota sistemi ile daha fonksiyonel bir API tasarımı gerçekleştirilmesi öngörülmektedir.