

# API DOCUMENTATION

## Genel API Formatları

Eğer gelen istek başarılı ve yanıt olarak data döndürmeyecekse gelecek olan cevap şu şekildedir:

```
{
  "data": null,
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Hangi eylem yapıldıysa başarılı mesajı"
}
```

Eğer gelen istek başarılı ve data döndürecekse gelecek olan cevap şu şekildedir. Datanın içi liste şeklinde ve adres, ürün gibi entitylerin özelliklerini içerir. Bu örnekte adres bilgilerini getirmiş:

```
{
  "data": [
    {
      "id": 2,
      "userId": 10,
      "addressLine1": "stringss",
      "addressLine2": "string",
      "city": "string",
      "postalCode": "string",
      "country": "string",
      "isDefault": false
    },
    {
      "id": 4,
      "userId": 10,
      "addressLine1": "string",
      "addressLine2": "string",
      "city": "string",
      "postalCode": "string",
      "country": "string",
      "isDefault": true
    }
  ],
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Başarılı!"
}
```

Eğer gelen istek başarısızsa gelecek olan cevap şu şekildedir. 2 örnek:

```
{
  "data": null,
  "statusCode": 404,
  "isSuccessful": false,
  "error": {
    "errors": [
      "Adres bulunamadı."
    ],
    "isShow": true
  },
  "message": null
}
```

```
{
  "Data": null,
  "StatusCode": 500,
  "IsSuccessful": false,
  "Error": {
    "Errors": [
      "Something went wrong! An error occurred while saving the entity changes. See the inner exception for details."
    ],
    "IsShow": true
  }
}
```

```
},  
"Message": null  
}
```

## Addresses API

**GET** /api/Addresses/getAllByUserId/{id}

- **Ne işe yarar:** Belirtilen kullanıcı ID'sine ait tüm adresleri listeler.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/Addresses/getAllByUserId/{id}> adresine GET isteği atılır. {id} yerine kullanıcı ID'si yazılmalıdır.
- **Yanıt Formatı:** Başarılı olduğunda, kullanıcı adres bilgilerini içeren bir JSON dizisi döner. Birden fazla adres dönebilir eğer varsa.

```
{  
  "data": [  
    {  
      "id": 2,  
      "userId": 10,  
      "addressLine1": "stringss",  
      "addressLine2": "string",  
      "city": "string",  
      "postalCode": "string",  
      "country": "string",  
      "isDefault": false  
    },  
    {  
      "id": 4,  
      "userId": 10,  
      "addressLine1": "string",  
      "addressLine2": "string",  
      "city": "string",  
      "postalCode": "string",  
      "country": "string",  
      "isDefault": true  
    }  
  ],  
  "statusCode": 200,  
  "isSuccessful": true,  
  "error": null,  
  "message": "Başarılı!"  
}
```

**GET** /api/Addresses/getById/{id}

- **Ne işe yarar:** Belirtilen adres ID'sine sahip tek bir adresi getirir.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/Addresses/getById/{id}> adresine GET isteği atılır. {id} yerine adres ID'si yazılmalıdır.
- **Yanıt Formatı:** Başarılı olduğunda, adres bilgilerini içeren bir JSON nesnesi döner.

```
{  
  "data": {  
    "id": 2,  
    "userId": 10,  
    "addressLine1": "stringss",  
    "addressLine2": "string",  
    "city": "string",  
    "postalCode": "string",  
    "country": "string",  
    "isDefault": true  
  },  
  "statusCode": 200,  
  "isSuccessful": true,  
  "error": null,  
}
```

```
"message": "Başarılı!"
}
```

## POST /api/Addresses/create

- **Ne işe yarar:** Yeni bir kullanıcı adresi oluşturur.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/Addresses/create> adresine POST isteği atılır. İstek gövdesinde (body) application/json formatında adres bilgileri gönderilir.
- **İstek Gövdesi Örnek:**

```
{
  "userId": 10,
  "addressLine1": "string",
  "addressLine2": "string",
  "city": "string",
  "postalCode": "string",
  "country": "string"
}
```

- **Yanıt Formatı:** Başarılı olduğunda, işlemin başarılı olduğuna dair bir mesaj içeren JSON nesnesi döner.

```
{
  "data": null,
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Adres Ekleme Başarılı!"
}
```

## PUT /api/Addresses/update

- **Ne işe yarar:** Mevcut bir adresi günceller.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/Addresses/update> adresine PUT isteği atılır. İstek gövdesinde (body) application/json formatında güncellenecek adres bilgileri gönderilir.
- **İstek Gövdesi Örnek:**

```
{
  "id": 3,
  "userId": 10,
  "addressLine1": "string",
  "addressLine2": "string",
  "city": "string",
  "postalCode": "string",
  "country": "string"
}
```

- **Yanıt Formatı:** Başarılı olduğunda, "Güncelleme başarılı!" mesajını içeren bir JSON nesnesi döner.

```
{
  "data": null,
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Güncelleme başarılı!"
}
```

## DELETE /api/Addresses/delete/{id}

- **Ne işe yarar:** Belirtilen ID'ye sahip adresi siler.
- **Link ve İstek Yöntemi:** `http://192.168.25.122:5102/api/Addresses/delete/{id}` adresine DELETE isteği atılır. {id} yerine adres ID'si yazılmalıdır.
- **Yanıt Formatı:** Başarılı olduğunda, "Silme Başarılı!" mesajını içeren bir JSON nesnesi döner.

```
{
  "data": null,
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Silme Başarılı!"
}
```

## POST /api/Addresses/setDefault/{userId}/{addressId}

- **Ne işe yarar:** Bir kullanıcının varsayılan adresini belirler. Bu sipariş vermesi için gereklidir.
- **Link ve İstek Yöntemi:** `http://192.168.25.122:5102/api/Addresses/setDefault/{userId}/{addressId}` adresine POST isteği atılır. {userId} ve {addressId} yerine ilgili ID'ler yazılmalıdır.
- **Yanıt Formatı:** Başarılı olduğunda, "Adres seçimi başarılı!" mesajını içeren bir JSON nesnesi döner.

```
{
  "data": null,
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Adres seçimi başarılı!"
}
```

# Auth API

## POST /api/Auth/register

- **Ne işe yarar:** Yeni bir kullanıcı kaydı oluşturur.
- **Link ve İstek Yöntemi:** `http://192.168.25.122:5102/api/Auth/register` adresine POST isteği atılır. İstek gövdesinde `application/json` formatında kullanıcı bilgileri (username, email, firstName, lastName, password) gönderilir.
- **İstek Gövdesi Örnek:**

```
{
  "username": "string",
  "email": "string@gmail.com",
  "firstName": "string",
  "lastName": "string",
  "password": "string"
}
```

- **Yanıt Formatı:** Başarılı olduğunda, "Kayıt başarılı" mesajını içeren bir JSON nesnesi döner.

```
{
  "data": null,
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Kayıt başarılı"
}
```

## POST /api/Auth/login

- **Ne işe yarar:** Kullanıcının sisteme giriş yapmasını sağlar.
- **Link ve İstek Yöntemi:** `http://192.168.25.122:5102/api/Auth/login` adresine POST isteği atılır. İstek gövdesinde `application/json` formatında email ve password bilgileri gönderilir.
- **İstek Gövdesi Örnek:**

```
{
  "email": "string@gmail.com",
  "password": "string"
}
```

- **Yanıt Formatı:** Başarılı olduğunda, kullanıcı bilgileri, access token ve refresh token içeren bir JSON nesnesi döner.

```
{
  "data": {
    "userInfo": {
      "id": 13,
      "username": "string",
      "email": "string@gmail.com",
      "firstName": "string",
      "lastName": "string",
      "isEmailConfirmed": false
    },
    "accessToken": {
      "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMyIsImh0dHA6Ly9yZ2h1bWVzLnhtbHNvYXAub3NjL3dzLzIwMDUvMDUvaWRlbnpRpdHkvY2xhaW1zL25hbWUiOiJzdHJpbmcilCJodHRwOi8vc2NoZW1hcy54bWxzbnBwLm9yZy93cy8yMDA1LzA1L2lkZW50aXR5L2NsYWltcy9lbWFnZW50aWw1c3MiO1sic3RyaW5nQGdtYWlsLmNvbSI6InN0cmduZ0BnbWFpbC5jb20iXSwiaHR0cDovL3NjaGVtYXMubWljcm9zb2Z0LmNvbS93cy8yMDA4LzA2L2lkZW50aXR5L2NsYWltcy9yb2x1IjoiaXBXNlciIsImV4cCI6MTc1MjY2NzY5ODcwiaXNzIjoiaHR0cHM6Ly9sb2NhbgHvc3Q6NmZlE5NyIsImF1ZCI6Imh0dHBzOi8vbG9jYXRob3N0Ojc0TcifQ.weXUmHg6Jc00hrD2CUhgRG_su3p8ntDeKCUg_AR8XPI",
      "accessTokenExpTime": "2025-07-16T15:08:18.1894706+03:00"
    },
    "refreshToken": {
      "refreshToken": "27330537-39e0-4ee3-8673-fe8f1beda237",
      "refreshTokenExpTime": "2025-07-23T15:07:18.1899213+03:00"
    }
  },
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Giriş başarılı"
}
```

## POST /api/Auth/refresh-token

- **Ne işe yarar:** Süresi dolmuş access token'ı yenilemek için kullanılır.
- **Link ve İstek Yöntemi:** `http://192.168.25.122:5102/api/Auth/refresh-token` adresine POST isteği atılır. İstek gövdesinde `application/json` formatında `refreshToken` gönderilir.
- **İstek Gövdesi Örnek:**

```
{
  "refreshToken": "cddec4ce-f2fa-458f-86ec-eaedcded4554"
}
```

- **Yanıt Formatı:** Başarılı olduğunda, yeni access token ve refresh token içeren bir JSON nesnesi döner.

[illegible]

```
{
  "accessTokenExpTime": "2025-07-16T11:55:56.8647959+03:00",
},
"refreshToken": {
  "refreshToken": "fd2d6b13-9148-4090-8188-a9c1bfa834ea",
  "refreshTokenExpTime": "2025-07-23T11:54:56.8648367+03:00"
}
},
"statusCode": 200,
"isSuccessful": true,
"error": null,
"message": "Token güncellendi"
}
```

#### POST /api/Auth/email-send-confirm-token

- **Ne işe yarar:** Kullanıcının email adresine doğrulama kodu gönderir.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/Auth/email-send-confirm-token/{userId}> adresine POST isteği atılır. {id} yerine kullanıcı ID'si yazılmalıdır..
- **Yanıt Formatı:** Başarılı olduğunda, "Email doğrulama kodu gönderildi" mesajını içeren bir JSON nesnesi döner.

```
{
  "data": null,
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Email doğrulama kodu gönderildi"
}
```

#### POST /api/Auth/verify-email-confirm-token

- **Ne işe yarar:** Email'e gönderilen doğrulama kodunu onaylar.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/Auth/verify-email-confirm-token?userId={userId}&confirmationToken={token}> adresine POST isteği atılır. userId ve confirmationToken query parametreleri olarak gönderilir.

**Örnek Query** = <http://192.168.25.122:5102/api/Auth/verify-email-confirm-token?userId=11&confirmationToken=943781>

- **Yanıt Formatı:** Başarılı olduğunda, "Email doğrulandı" mesajını içeren bir JSON nesnesi döner.

```
{
  "data": null,
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Email doğrulandı"
}
```

#### POST /api/Auth/change-password

- **Ne işe yarar:** Kullanıcının mevcut şifresini değiştirmesini sağlar.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/Auth/change-password> adresine POST isteği atılır. İstek gövdesinde userId, oldPassword ve newPassword bilgileri application/json formatında gönderilir.
- **Örnek İstek Gövdesi:**

```
{
  "userId": 13,
  "oldPassword": "string",
  "newPassword": "stringg"
}
```

```
}
```

- **Yanıt Formatı:** Başarılı olduğunda, "Şifre değiştirildi" mesajını içeren bir JSON nesnesi döner.

```
{
  "data": null,
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Şifre değiştirildi"
}
```

## CardItem API

**GET** /api/CardItem/getUsersCardItems/{userId}

- **Ne işe yarar:** Belirtilen kullanıcının sepetindeki ürünleri listeler.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/CardItem/getUsersCardItems/{userId}> adresine GET isteği atılır. {userId} yerine kullanıcı ID'si yazılmalıdır.
- **Yanıt Formatı:** Başarılı olduğunda, sepetteki ürünlerin listesini içeren bir JSON dizisi döner. Data içinde birden fazla cardItem olabilir.

```
{
  "data": [
    {
      "id": 7,
      "userId": 10,
      "productId": 14,
      "quantity": 50
    }
  ],
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Başarılı!"
}
```

**GET** /api/CardItem/getById/{id}

- **Ne işe yarar:** Belirli bir sepet öğesini (card item) ID'sine göre getirir.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/CardItem/getById/{id}> adresine GET isteği atılır. {id} yerine sepet öğesi ID'si yazılmalıdır.
- **Yanıt Formatı:** Başarılı olduğunda, sepet öğesi bilgilerini içeren bir JSON nesnesi döner

```
{
  "data": {
    "id": 7,
    "userId": 10,
    "productId": 14,
    "quantity": 50
  },
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Başarılı!"
}
```

**POST** /api/CardItem/create

- **Ne işe yarar:** Bir kullanıcının sepetine yeni bir ürün ekler.

- **Link ve İstek Yöntemi:** `http://192.168.25.122:5102/api/CardItem/create` adresine POST isteği atılır. İstek gövdesinde `userId`, `productId`, ve `quantity` bilgileri `application/json` formatında gönderilir.
- **İstek Gövdesi Örnek:**

```
{
  "userId": 10,
  "productId": 15,
  "quantity": 40
}
```

- **Yanıt Formatı:** Başarılı olduğunda, "Kart Ekleme Başarılı!" mesajını içeren bir JSON nesnesi döner.

```
{
  "data": null,
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Kart Ekleme Başarılı!"
}
```

#### PUT /api/CardItem/updateQuantity

- **Ne işe yarar:** Sepetteki bir ürünün miktarını günceller.
- **Link ve İstek Yöntemi:** `http://192.168.25.122:5102/api/CardItem/updateQuantity?cardItemId={cardItemId}&newQuantity={newQuantity}` adresine PUT isteği atılır. `cardItemId` ve `newQuantity` query parametreleri olarak gönderilir.

**Query Örnek:** `http://192.168.25.122:5102/api/CardItem/updateQuantity?cardItemId=7&newQuantity=70`

- **Yanıt Formatı:** Başarılı olduğunda, "Güncelleme başarılı!" mesajını içeren bir JSON nesnesi döner.

```
{
  "data": null,
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Güncelleme başarılı!"
}
```

#### DELETE /api/CardItem/delete/{id}

- **Ne işe yarar:** Sepetten belirli bir ürünü (card item) ID'si ile siler.
- **Link ve İstek Yöntemi:** `http://192.168.25.122:5102/api/CardItem/delete/{id}` adresine DELETE isteği atılır. `{id}` yerine sepet öğesi ID'si yazılmalıdır.
- **Yanıt Formatı:** Başarılı olduğunda, "Silme Başarılı!" mesajını içeren bir JSON nesnesi döner.

```
{
  "data": null,
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Silme Başarılı!"
}
```

#### DELETE /api/CardItem/clearUsersCardItems/{userId}

- **Ne işe yarar:** Belirtilen kullanıcının sepetindeki tüm ürünleri temizler.
- **Link ve İstek Yöntemi:** `http://192.168.25.122:5102/api/CardItem/clearUsersCardItems/{userId}` adresine DELETE isteği atılır. `{userId}` yerine kullanıcı ID'si yazılmalıdır.



- **Yanıt Formatı:** Başarılı olduğunda, "Kullanıcın sepeti temizlendi!" mesajını içeren bir JSON nesnesi döner.

```
{
  "data": null,
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Kullanıcın sepeti temizlendi!"
}
```

## Categories API

**GET** /api/Categories/getAll

- **Ne işe yarar:** Tüm ürün kategorilerini listeler.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/Categories/getAll> adresine GET isteği atılır.
- **Yanıt Formatı:** Başarılı olduğunda, tüm kategorileri içeren bir JSON dizisi döner.

```
{
  "data": [
    {
      "id": 3,
      "name": "Beyaz Eşya",
      "description": "asdasdasdadad"
    },
    {
      "id": 4,
      "name": "sadasd",
      "description": "adasdasdasd"
    },
    {
      "id": 5,
      "name": "asddasda",
      "description": "strasdasdasding"
    },
    {
      "id": 6,
      "name": "elektronik",
      "description": "asdasdasdasdasda"
    }
  ],
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Başarılı!"
}
```

**GET** /api/Categories/getById/{id}

- **Ne işe yarar:** Belirtilen ID'ye sahip kategoriye getirir.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/Categories/getById/{id}> adresine GET isteği atılır. {id} yerine kategori ID'si yazılmalıdır.
- **Yanıt Formatı:** Başarılı olduğunda, kategori bilgilerini içeren bir JSON nesnesi döner.

```
{
  "data": {
    "id": 3,
    "name": "Beyaz Eşya",
    "description": "asdasdasdadad"
  },
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Başarılı!"
}
```

```
}
```

## POST /api/Categories/create

- **Ne işe yarar:** Yeni bir kategori oluşturur.
- **Link ve İstek Yöntemi:** `http://192.168.25.122:5102/api/Categories/create` adresine POST isteği atılır. İstek gövdesinde `name` ve `description` bilgileri `application/json` formatında gönderilir.
- **İstek Gövdesi Örnek:**

```
{  
  "name": "elektronik",  
  "description": "asdasdasdasdasda"  
}
```

- **Yanıt Formatı:** Başarılı olduğunda, "Kategori Ekleme Başarılı!" mesajını içeren bir JSON nesnesi döner.

```
{  
  "data": null,  
  "statusCode": 200,  
  "isSuccessful": true,  
  "error": null,  
  "message": "Kategori Ekleme Başarılı!"  
}
```

## PUT /api/Categories/update

- **Ne işe yarar:** Mevcut bir kategoriye günceller.
- **Link ve İstek Yöntemi:** `http://192.168.25.122:5102/api/Categories/update` adresine PUT isteği atılır. İstek gövdesinde `id`, `name` ve `description` bilgileri `application/json` formatında gönderilir.
- **İstek Gövdesi Örnek:**

```
{  
  "id": 6,  
  "name": "elektronik updated",  
  "description": "string"  
}
```

- **Yanıt Formatı:** Başarılı olduğunda, "Güncelleme başarılı!" mesajını içeren bir JSON nesnesi döner.

```
{  
  "data": null,  
  "statusCode": 200,  
  "isSuccessful": true,  
  "error": null,  
  "message": "Güncelleme başarılı!"  
}
```

## DELETE /api/Categories/delete/{id}

- **Ne işe yarar:** Belirtilen ID'ye sahip kategoriye siler.
- **Link ve İstek Yöntemi:** `http://192.168.25.122:5102/api/Categories/delete/{id}` adresine DELETE isteği atılır. `{id}` yerine kategori ID'si yazılmalıdır.
- **Yanıt Formatı:** Başarılı olduğunda, "Silme Başarılı!" mesajını içeren bir JSON nesnesi döner.

```
{  
  "data": null,  
  "statusCode": 200,  
  "isSuccessful": true,  
  "error": null,  
  "message": "Silme Başarılı!"  
}
```

# FavoriteProducts API

## GET /api/FavoriteProducts/list/{userId}

- **Ne işe yarar:** Belirtilen kullanıcının favori ürünlerini listeler.
- **Link ve İstek Yöntemi:** `http://192.168.25.122:5102/api/FavoriteProducts/list/{userId}` adresine GET isteği atılır. {userId} yerine kullanıcı ID'si yazılmalıdır.
- **Yanıt Formatı:** Başarılı olduğunda, favori ürünlerin listesini içeren bir JSON dizisi döner.

```
{
  "data": [
    {
      "id": 14,
      "categoryId": 3,
      "name": "asdasdasdasd",
      "description": "asdasdasdasd",
      "price": 213,
      "stock": 123,
      "imageUrl": "https://yirmibesyazilim.blob.core.windows.net/productimages/4513de7d-4a83-4f45-8aa0-8fe345074cf4.png"
    }
  ],
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Favori ürünler başarıyla getirildi"
}
```

## POST /api/FavoriteProducts/addFavorite

- **Ne işe yarar:** Bir ürünü kullanıcının favorilerine ekler.
- **Link ve İstek Yöntemi:** `http://192.168.25.122:5102/api/FavoriteProducts/addFavorite` adresine POST isteği atılır. İstek gövdesinde productId ve userId bilgileri application/json formatında gönderilir.
- **İstek Gövdesi Örnek:**

```
{
  "productId": 14,
  "userId": 10
}
```

- **Yanıt Formatı:** Başarılı olduğunda, "Favori ürün eklendi" mesajını içeren bir JSON nesnesi döner.

```
{
  "data": null,
  "statusCode": 201,
  "isSuccessful": true,
  "error": null,
  "message": "Favori ürün eklendi"
}
```

## DELETE /api/FavoriteProducts/removeFavorite

- **Ne işe yarar:** Bir ürünü kullanıcının favorilerinden kaldırır.
- **Link ve İstek Yöntemi:** `http://192.168.25.122:5102/api/FavoriteProducts/removeFavorite` adresine DELETE isteği atılır. İstek gövdesinde productId ve userId bilgileri application/json formatında gönderilir.
- **İstek Gövdesi Örnek:**

```
{
  "productId": 14,
  "userId": 10
}
```

- **Yanıt Formatı:** Başarılı olduğunda, "Favori ürün başarıyla kaldırıldı" mesajını içeren bir JSON nesnesi döner.

```
{
  "data": null,
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Favori ürün başarıyla kaldırıldı"
}
```

## Orders API

### POST /api/Orders/confirmCard/{userId}

- **Ne işe yarar:** Kullanıcının sepetini onaylayarak bir sipariş oluşturur.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/Orders/confirmCard/{userId}> adresine POST isteği atılır. {userId} yerine kullanıcı ID'si yazılmalıdır.
- **Yanıt Formatı:** Başarılı olduğunda, oluşturulan siparişin detaylarını içeren bir JSON nesnesi döner.,
- **Not:** Sipariş oluşturma zamanını da buraya ekleyebiliriz.

```
{
  "data": {
    "id": 4,
    "userId": 10,
    "addressId": 2,
    "totalAmount": 11490,
    "status": true,
    "orderItems": [
      {
        "orderId": 4,
        "productId": 14,
        "price": 213,
        "quantity": 50
      },
      {
        "orderId": 4,
        "productId": 15,
        "price": 21,
        "quantity": 40
      }
    ]
  },
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Sipariş oluşturuldu."
}
```

### GET /api/Orders/getById/{id}

- **Ne işe yarar:** Belirtilen ID'ye sahip siparişi getirir.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/Orders/getById/{id}> adresine GET isteği atılır. {id} yerine sipariş ID'si yazılmalıdır.
- **Yanıt Formatı:** Başarılı olduğunda, sipariş detaylarını içeren bir JSON nesnesi döner.
- **Not:** OrderItems boş geliyor normalde gelmemesi lazım düzeltilecek.

```
{
  "data": {
    "id": 3,
    "userId": 10,
    "addressId": 1,
    "totalAmount": 19200,
    "status": true,
    "orderItems": []
  },
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Sipariş getirildi."
}
```

```
"statusCode": 200,  
"isSuccessful": true,  
"error": null,  
"message": "Başarılı!"  
}
```

**GET** /api/Orders/getUsersOrders/{userId}

- **Ne işe yarar:** Belirtilen kullanıcının tüm siparişlerini listeler.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/Orders/getUsersOrders/{userId}> adresine GET isteği atılır. {userId} yerine kullanıcı ID'si yazılmalıdır.
- **Yanıt Formatı:** Başarılı olduğunda, kullanıcının tüm siparişlerini içeren bir JSON dizisi döner.

```
{  
  "data": [  
    {  
      "id": 1,  
      "userId": 10,  
      "addressId": 1,  
      "totalAmount": 9600,  
      "status": true,  
      "orderItems": []  
    },  
    {  
      "id": 2,  
      "userId": 10,  
      "addressId": 1,  
      "totalAmount": 9600,  
      "status": true,  
      "orderItems": []  
    },  
    {  
      "id": 3,  
      "userId": 10,  
      "addressId": 1,  
      "totalAmount": 19200,  
      "status": true,  
      "orderItems": []  
    },  
    {  
      "id": 4,  
      "userId": 10,  
      "addressId": 2,  
      "totalAmount": 11490,  
      "status": true,  
      "orderItems": []  
    }  
  ],  
  "statusCode": 200,  
  "isSuccessful": true,  
  "error": null,  
  "message": "Başarılı!"  
}
```

## Products API

**GET** /api/Products/list

- **Ne işe yarar:** Ürünleri kategori, isim, fiyat aralığı gibi filtrelere göre listeler.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/Products/list> adresine GET isteği atılır. CategoryId, Name, MinPrice, MaxPrice, OrderByPriceAscending gibi query parametreleri ile filtreleme yapılabilir.

### Örnek Query:

```
http://192.168.25.122:5102/api/Products/list?CategoryId=3&Name=a&MinPrice=10&MaxPrice=1000&OrderByPriceAscending=true
```

- **Not:** Tüm queryleri göndermek zorunda değiliz. Sadece gerekenleri gönderebiliriz veya hiç göndermeyip sadece /list olarak istek atarsak filtreleme yapmadan ürünleri getirir.
- **Not:** OrderByPriceAscending “false” gönderilirse tam tersi descending olarak sıralar.
- **Yanıt Formatı:** Başarılı olduğunda, filtelenmiş ürünlerin listesini içeren bir JSON dizisi döner.

```
{
  "data": [
    {
      "id": 15,
      "categoryId": 3,
      "name": "aaa",
      "description": "aaa",
      "price": 21,
      "stock": 12,
      "imageUrl": "BU KISIM TEST İÇİN SİLİNECEK"
    },
    {
      "id": 14,
      "categoryId": 3,
      "name": "asdasdasdasd",
      "description": "asdasdasdasd",
      "price": 213,
      "stock": 123,
      "imageUrl": "https://yirmibesyazilim.blob.core.windows.net/productimages/4513de7d-4a83-4f45-8aa0-8fe345074cf4.png"
    }
  ],
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Başarılı!"
}
```

#### GET /api/Products/getById/{id}

- **Ne işe yarar:** Belirtilen ID'ye sahip ürünü getirir.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/Products/getById/{id}> adresine GET isteği atılır. {id} yerine ürün ID'si yazılmalıdır.
- **Yanıt Formatı:** Başarılı olduğunda, ürün detaylarını içeren bir JSON nesnesi döner.

```
{
  "data": {
    "id": 14,
    "categoryId": 3,
    "name": "asdasdasdasd",
    "description": "asdasdasdasd",
    "price": 213,
    "stock": 123,
    "imageUrl": "https://yirmibesyazilim.blob.core.windows.net/productimages/4513de7d-4a83-4f45-8aa0-8fe345074cf4.png"
  },
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Başarılı!"
}
```

#### POST /api/Products/create

- **Ne işe yarar:** Yeni bir ürün oluşturur ve ürün resmini yükler.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/Products/create> adresine POST isteği atılır. İstek multipart/form-data formatında olmalıdır ve CategoryId, Name, Description, Price, Stock ve Image (dosya) alanlarını içermelidir.

CategoryId  
integer(\$int32) 3  
☐ Send empty value

Name  
string string  
☐ Send empty value

Description  
string string  
☐ Send empty value

Price  
integer(\$int32) 20  
☐ Send empty value

Stock  
integer(\$int32) 200  
☐ Send empty value

Image  
string(\$binary)  Dosya seçilmedi  
☐ Send empty value

- **Yanıt Formatı:** Başarılı olduğunda, "Ürün eklendi" mesajını içeren bir JSON nesnesi döner.

```
{  
  "data": null,  
  "statusCode": 201,  
  "isSuccessful": true,  
  "error": null,  
  "message": "Ürün eklendi"  
}
```

#### PUT /api/Products/update

- **Ne işe yarar:** Mevcut bir ürünü günceller ve isteğe bağlı olarak resmini değiştirir.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/Products/update> adresine PUT isteği atılır. İstek multipart/form-data formatında olmalıdır ve Id, CategoryId, Name, Description, Price, Stock ve Image (dosya) alanlarını içermelidir.

Id  
integer(\$int32) 15  
☐ Send empty value

CategoryId  
integer(\$int32) 4  
☐ Send empty value

Name  
string string  
☐ Send empty value

Description  
string string  
☐ Send empty value

Price  
integer(\$int32) 66  
☐ Send empty value

Stock  
integer(\$int32) 100  
☐ Send empty value

Image  
string(\$binary)  Dosya seçilmedi  
☐ Send empty value

- **Yanıt Formatı:** Başarılı olduğunda, "Güncellendi" mesajını içeren bir JSON nesnesi döner.

```
{
  "data": null,
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Güncellendi"
}
```

**DELETE** /api/Products/delete/{id}

- **Ne işe yarar:** Belirtilen ID'ye sahip ürünü siler.
- **Link ve İstek Yöntemi:** <http://192.168.25.122:5102/api/Products/delete/{id}> adresine DELETE isteği atılır. {id} yerine ürün ID'si yazılmalıdır.
- **Yanıt Formatı:** Başarılı olduğunda, "Silindi" mesajını içeren bir JSON nesnesi döner.

```
{
  "data": null,
  "statusCode": 200,
  "isSuccessful": true,
  "error": null,
  "message": "Silindi"
}
```