**Gençler için Geleceğin Mesleklerine Yönelik Eğitim Programları**

**SAKARYA DİJİTAL GENÇLİK MERKEZİ (DIGEM) EĞİTİMLERİ**

**Ders Programları ve İçerikleri**

**TEMA: DİJİTAL TASARIM  
Eğitim Tarihleri: 20 Ekim 2025 – 27 Şubat 2026**

İçindekiler Tablosu

[**1.** **BTD01 – Başlangıç Temel Dersler** 3](#_Toc211614633)

[**a.** **Temel Bilgisayar Bilimleri** 3](#_Toc211614634)

[**i.** **Temel İşletim Sistemi Bilgisi** 3](#_Toc211614635)

[**ii.** **Ofis Uygulamaları** 17](#_Toc211614636)

[**1.** **Microsoft Word** 17](#_Toc211614637)

[**2.** **Microsoft Excel** 22](#_Toc211614638)

[**3.** **Microsoft PowerPoint** 38](#_Toc211614639)

[**iii.** **Dijital Okur Yazarlık** 44](#_Toc211614640)

[**iv.** **Temel Yazılım Rehberi** 50](#_Toc211614641)

[**1.** **Temel Yazılım Bilgisi** 50](#_Toc211614642)

[**2.** **Temel SQL Bilgisi** 56](#_Toc211614643)

[**4.** **Temel Python Eğitimi** 83](#_Toc211614644)

[**6.** **Temel Web Uygulaması Geliştirme** 87](#_Toc211614645)

[**a.** **Temel Html Bilgisi** 87](#_Toc211614646)

[**b.** **Temel CSS Bilgisi** 98](#_Toc211614647)

[**c.** **Web Teknolojisi Bilgisi** 117](#_Toc211614648)

[**7.** **Doküman Versiyon Takibi** 148](#_Toc211614649)

[**a.** **Git Eğitimi** 148](#_Toc211614650)

### BTD01 – Başlangıç Temel Dersler

### **Temel Bilgisayar Bilimleri**

### **Temel İşletim Sistemi Bilgisi**

**İşletim Sistemi (OS): Bilgisayarın "AKIL HOCASI"**

Kullandığınız akıllı telefon, dizüstü bilgisayar veya tabletin içinde, görünmez bir kahraman çalışıyor. İşte o, **İşletim Sistemi**dir.

**Kısa ve Net Tanım:**

İşletim Sistemi; bilgisayarın donanımı (fiziksel parçaları) ile sizin aranızda bir köprü kuran, bilgisayarı çalışır hale getiren ve uygulamalarınızı çalıştırmak için gerekli ortamı sağlayan temel yazılımdır.

Yani, bilgisayarınızın "beyni" ve "ruhu"dur.

**Benzetmeyle Anlatalım: Bir Restoran Düşünün!**

Bir restorana gittiğinizi hayal edin:

* **Siz:** Müşterisiniz. (Kullanıcı)
* **Sipariş Ettiğiniz Yemekler:** Uygulamalardır. (Chrome, Instagram, Oyunlar, Word)
* **Masa:** Kullanıcı Arayüzüdür (GUI). (Windows'taki masaüstü, menüler)
* **Garson:** İşletim Sisteminin bir parçasıdır. Sizin siparişinizi (tıklamalarınızı) mutfağa iletir.
* **Mutfağın İçi (Mutfağa Giremediğiniz Yer):** Donanımdır. (İşlemci, RAM, Sabit Disk)
* **Şef ve Mutfak Ekibi:** İşletim Sisteminin diğer parçalarıdır. Garsonun getirdiği siparişleri alır, hangi malzemenin ne zaman kullanılacağını planlar, yemeği pişirir ve garsona teslim eder.

İşletim Sistemi, bu restorandaki "Garson" ve "Şef"in yaptığı her şeyi yapar. Sizinle makinenin karmaşık donanımı arasında tercümanlık yapar, her şeyin düzgün ve verimli çalışmasını sağlar.

**İşletim Sistemi Ne Yapar? (Asıl Görevleri)**

* **Donanımı Yönetir (Trafik Polisi Gibi):**
* **İşlemciyi (CPU) yönetir:** Hangi programın ne kadar işlemci gücü kullanacağına karar verir.
* **Belleği (RAM) yönetir:** Aynı anda açık olan programların birbirine karışmamasını sağlar.
* **Depolamayı (SSD/HDD) yönetir:** Dosyalarınızı kaydeder, gerektiğinde bulup size getirir.
* Çevre birimlerini (klavye, fare, yazıcı) yönetir. Yazıcıya "yazdır" dediğinizde, bu komutu yazıcının anlayacağı dile çevirir.
* **Uygulamaları Çalıştırır ve Yönetir (Ev Sahibi Gibi):**
* Uygulamalara, ihtiyaç duydukları kaynakları (bellek, işlemci gücü) sağlar.
* Farklı uygulamaların birbirini engellemeden, uyum içinde çalışmasını sağlar.
* **Kullanıcı Arayüzünü Sağlar (Tercüman Gibi):**
* Sizin bilgisayarla iletişim kurmanızı sağlayan ekran, pencereler, menüler ve butonlardır. Tıklamalarınızı makine diline çevirir. (Windows, macOS, Linux ve Android/iOS'un farklı görünmesinin sebebi budur).

**Günlük Hayattan Örnekler (Hangi Cihazlarda Var?)**

* **Bilgisayarlarda:** **Windows** (en yaygın), **macOS** (Apple bilgisayarlar), **Linux** (geliştiriciler ve sunucular için).
* **Akıllı Telefonlarda:** **Android** (Google) ve **iOS** (Apple).
* **Hatta akıllı saatlerin, televizyonların ve araç içi ekranların bile kendine özgü işletim sistemleri vardır!**

**Sonuç:**

**İşletim sistemi olmadan, bilgisayarınız sadece plastik ve metal bir yığındır.** Siz bir tuşa bastığınızda o tuşun ne anlama geldiğini bilmez. İşletim sistemi, ona "hayat" verir ve sizinle anlaşabilmesini sağlar. Bu yüzden, bir bilgisayarı ilk açtığınızda karşınıza çıkan ilk şey işletim sistemidir.

DİGEM'deki bir sonraki adımınız, hangi işletim sistemini kullandığınızı keşfetmek ve onun sunduğu olanakları daha iyi anlamak olabilir!

**İşletim Sisteminin Üç Temel Kahramanı: Kernel, Shell ve Dosya Yöneticisi**

Restoranımızı hatırlayın. İşletim sistemi, restoranın tüm işleyişinden sorumluydu. Şimdi bu restoranın çalışanlarını daha yakından tanıyalım:

**1. Kernel (Çekirdek) - Restoranın "Şefi"**

**Kernel, işletim sisteminin kalbidir ve en temel görevleri yapar.**

* **Nerede Çalışır?** Tüm zamanını **mutfakta** (sistemin derinliklerinde) geçirir. Müşteriler onu (siz) genelde görmezsiniz bile.
* **Ne İş Yapar?**
  + **Kaynakları Yönetir:** Mutfaktaki tüm malzemeleri (İşlemci, RAM, Disk) yönetir. "Bu yemek için ne kadar et gidecek?", "Pilav pişerken sebzeler doğransın" gibi kararları o verir.
  + **Süreçleri Yönetir:** Mutfakta aynı anda ızgarada et, ocakta çorba, fırında tatlı pişer. Şef, bunların hepsinin birbirine karışmadan, zamanında ve verimli bir şekilde pişmesini sağlar.
  + **Donanımı Doğrudan Kontrol Eder:** Ocağın ateşini ayarlar, fırının sıcaklığını kontrol eder. Yani donanımla (fiziksel parçalarla) doğrudan iletişim kuran kişidir.

**Özetle:** Kernel, bilgisayarın donanımını yöneten, her şeyin arka planda sorunsuz çalışmasını sağlayan "Şef"tir. **Güç buradadır.**

**2. Shell (Kabuk) - Restoranın "Garsonu"**

**Shell, sizin (kullanıcının) Kernel (Şef) ile iletişim kurmanızı sağlayan arayüzdür.**

* **Nerede Çalışır?** **Masa** ile **mutfak** arasında mekik dokur. Sizinle etkileşimde olan kişidir.
* **Ne İş Yapar?**
  + **Komutları İletir:** Siz "Şu dosyayı aç" veya "Şu programı çalıştır" dediğinizde, garson bu siparişi alır ve mutfağa (Kernel'e) iletir: "Masa 5, bir adet Chrome istiyor!"
  + **Sonuçları Getirir:** Kernel (Şef) istediğiniz işlemi tamamladığında, sonucu size Shell (Garson) getirir. "İşte buyrun, açtığınız dosya."
  + **İki Türü Vardır:**
    - **Grafiksel Shell (GUI):** Windows'taki masaüstü, başlat menüsü, ikonlar. Garsonun size getirdiği **menüdür**. Siz sadece istediğiniz yemeğe (programa) tıklarsınız.
    - **Komut Satırı (CLI):** Windows'ta "CMD" veya "PowerShell", Linux/macOS'ta "Terminal". Bu, garsonla **konuşarak sipariş vermektir**. "get-me chrome.exe" gibi komutlar yazarsınız. Daha güçlü ve hızlıdır ama komutları bilmeniz gerekir.

**Özetle:** Shell, sizinle bilgisayarın "beyni" arasındaki tercüman ve aracıdır. **Kontrol buradan sağlanır.**

**3. Dosya Yöneticisi (File Manager) - Restoranın "Depo Sorumlusu "**

**Dosya Yöneticisi, Shell'in özel bir parçasıdır ve bilgisayarınızdaki tüm dosya ve klasörleri düzenlemekle görevlidir.**

* **Nerede Çalışır?** Restoranın **depo** ve **arsiv** bölümlerinde. Aynı zamanda misafirlere (kullanıcıya) bu depoların düzenini gösterir.
* **Ne İş Yapar?**
  + **Dosyaları Düzenler:** Dosyaları taşır, kopyalar, siler, yeniden adlandırır. Yani depoları ve dosya dolaplarını düzenler.
  + **Dosya Konumlarını Bilir:** "Word belgeleri nerede?", "İndirilenler klasörüne nasıl giderim?" sorularının cevabını bilir ve size gösterir.
  + **Görsel Bir Arayüz Sunar:** Windows'taki "Dosya Gezgini" veya macOS'taki "Finder" gibi. Bu kişi, size deponun bir haritasını ve katalogunu verir. Siz bir dosyaya tıkladığınızda, o aslında Shell (Garson) aracılığıyla Kernel'e (Şef'e) "Depodan şu dosyayı getir" komutunu verir.

**Özetle:** Dosya Yöneticisi, bilgisayarınızdaki tüm dijital "eşyalarınızı" düzenleyen, bulmanızı ve yönetmenizi sağlayan **organizatördür**. **Düzen buradan sağlanır.**

**Restoran Senaryosu ile Özet:**

1. **Siz** (Müşteri), **Dosya Gezgini'nde** (Depo Sorumlusu) bir film dosyasına çift tıklarsınız.
2. **Shell** (Garson), bu tıklama emrini alır ve "Müşteri şu film dosyasını izlemek istiyor!" diye bağırır.
3. **Kernel** (Şef), bu emri duyar ve hemen harekete geçer:
   * **İşlemciye** (Ocağa) talimat verir: "Video kodeklerini çalıştır, gücünü artır!"
   * **RAM'e** (Hazırlık Tezgahına) talimat verir: "Film verilerini buraya yükle!"
   * **Ekran Kartına** (Sunum Tabağına) talimat verir: "Gelen bu verileri işle ve monitöre gönder!"
4. Sonuç olarak, film pencerenizde oynar. Tüm bu karmaşık süreç, üç kahramanın uyumlu çalışması sayesinde saniyeler içinde gerçekleşir.

**Sonuç:**

* **Kernel (Şef):** **Güç** ve **kontrol** buradadır. Arkada, her şeyi yönetir.
* **Shell (Garson):** **İletişim** ve **etkileşim** buradan sağlanır. Sizin komutlarınızı iletir.
* **Dosya Yöneticisi (Depocu):** **Düzen** ve **erişim** buradan sağlanır. Dosyalarınızı yönetmenizi kolaylaştırır.

Bu üçlü birlikte çalışarak, sizin bilgisayarınızı verimli ve kolay bir şekilde kullanmanızı sağlar.

**Single Tasking vs Multi Tasking: İşletim Sisteminin "Yapabilecekleri"**

Bilgisayarın işlemcisini (CPU), bir **aşçı** olarak düşünün. Bu aşçının aynı anda kaç iş yapabildiği, işletim sisteminin görev yönetim düzenine bağlıdır.

**1. Single Tasking (Tek Görevli) - "Sırayla Yemek Pişiren Aşçı"**

**Tanım:** İşlemcinin, aynı anda **sadece bir iş** (process) yapabildiği düzendir. Bir iş bitmeden diğerine geçemez.

**Benzetme:**

* **Aşçı:** Sadece **bir tane yemek** pişirebilir.
* **Senaryo:** Aşçı, makarna pişiriyor. Makarna pişene kadar:
  + Salatayı hazırlayamaz.
  + Soğanı doğrayamaz.
  + Çay demleyemez.
  + Hepsi sırayla olacak! Önce makarna bitecek, sonra salataya geçilecek.

**Özellikleri:**

* **Basittir:** Yönetmesi kolaydır, karmaşık değildir.
* **Yavaştır:** Bir program yazdırma işlemi yapıyorsa, yazdırma bitene kadar bilgisayarda başka hiçbir şey yapamazsınız.
* **Kaynak Kullanımı Düşüktür:** Aynı anda çok fazla kaynağı yönetmesi gerekmez.
* **Örnekler:** İlk kişisel bilgisayarlar (MS-DOS gibi), günümüzde bazı basit gömülü sistemler (bir hesap makinesi gibi).

**Özet: "Bir iş bitsin, sonra diğerine geçelim."**

**2. Multi Tasking (Çok Görevli) - "Profesyonel Bir Restoranın Şefi"**

**Tanım:** İşlemcinin, aynı anda **birden fazla işi** yürüttüğü **izlenimini** veren düzendir. Aslında işlemci çok hızlı olduğu için görevler arasında çok kısa sürelerde geçiş yapar.

**Benzetme:**

* **Aşçı:** Aynı anda **birden fazla tencerede** farklı yemekler pişirebilir.
* **Senaryo:** Şef aynı anda:
  + Makarnayı karıştırır,
  + Hemen ardından sosun altını kısar,
  + Sonra salatayı kontrol eder,
  + Ardından makarnaya tekrar döner.
  + Bu geçişler o kadar hızlıdır ki, dışarıdan bakan için tüm yemekler aynı anda pişiyor gibi görünür.

**Özellikleri:**

* **Verimlidir:** Kullanıcı aynı anda müzik dinlerken, internette gezerken bir dosya indirebilir.
* **Hızlıdır:** İşlemcinin gücünden etkili bir şekilde yararlanır.
* **Karmaşıktır:** Hangi göreve ne kadar süre ayrılacağını (zamanlama - scheduling) işletim sisteminin akıllıca yönetmesi gerekir.
* **Günümüzün Standardıdır:** Windows, macOS, Linux, Android, iOS gibi tüm modern işletim sistemleri çok görevlidir.

**Özet: "Her işe bir saniye ayır, hepsi aynı anda ilerlesin."**

**Karşılaştırmalı Özet Tablosu**

| Özellik | Single Tasking (Tek Görevli) | Multi Tasking (Çok Görevli) |
| --- | --- | --- |
| **Temel Mantık** | "Sırayla İşlem" | "Hızlı Geçiş" |
| **Aynı Anda Çalışma** | **Hayır.** Tek bir görev. | **Evet.** Birden fazla görev. |
| **Verimlilik** | Düşük | Yüksek |
| **Karmaşıklık** | Basit | Karmaşık |
| **Kaynak Kullanımı** | Düşük | Yüksek |
| **Kullanıcı Deneyimi** | Sınırlı. Beklemek gerekir. | Zengin. Aynı anda birçok şey yapılabilir. |
| **Güncel Örnek** | Hesap makinesi, eski sistemler | Akıllı telefonunuz, dizüstü bilgisayarınız |

**Toplu İşlem (Batch Processing) ve Etkileşimli İşlem (Interactive Processing)**

**Toplu İşlem vs Etkileşimli İşlem: "Hepsi Bir Arada" vs "Anında Yanıt"**

Bu kavramlar, bilgisayarın işleri *nasıl ve ne zaman* yaptığıyla ilgilidir. İşlemciyi (CPU) yine bir aşçı olarak düşünelim.

**1. Toplu İşlem (Batch Processing)**

**Tanım:**Benzer türdeki bir grup işin (job), bir araya getirilip (toplu halde) ve önceden belirlenmiş bir sırayla, kullanıcıyla etkileşime girmeden işlendiği yöntemdir.

**Benzetme:**

* Aşçı, bir "yemek partisi" için hazırlık yapmaktadır.
* **Senaryo:**Aşçının yapacağı tüm işler bellidir:
  1. 100 adet köfte yoğur.
  2. 5 kilo patatesi soy ve doğra.
  3. 100 adet içli köfte hazırla.
* Aşçı, bu işleri sırayla ve aralıksız olarak yapar. Bir işi bitirmeden diğerine geçmez. Siz ona "Dur, şu köfteden bir ısırık alayım" diye müdahale edemezsiniz. Tüm işler tamamlandığında size "Hepsi hazır!" der.

**Gerçek Hayat Örnekleri:**

* **Banka Ay Sonu Ekstreleri**: Ay boyunca yapılan tüm işlemler kaydedilir, ay sonunda bir kerede hesaplanır ve ekstreniz hazırlanır.
* **Üniversite Not Ortalaması Hesaplama:**Dönem sonunda tüm sınav notları ve krediler toplanır, tek seferde ortalama hesaplanır.
* **Fotoğraf Filtreleme:**1000 fotoğrafı seçip "hepsini siyah-beyaz yap" dersiniz. Bilgisayar tüm dosyaları sırayla işler, bitince size haber verir.

**Özellikleri:**

* **Zamanında Esnektir**: İşler genellikle sistemin yoğun olmadığı zamanlarda (gece) çalıştırılır.
* **Yüksek Verim:**Sürekli bir iş yaptığı için verimlilik yüksektir.
* **Etkileşim Yoktur:**İşlem başladı mı, durdurup ona müdahale edemezsiniz.
* **Öncelik:**İşlem Tamamlanma Süresi (Throughput). Aynı sürede en fazla işi bitirmek hedeflenir.

**Özet:** "Ben işleri listeledim, sen hepsini sırayla bitir, sonra gel bana söyle."

**2. Etkileşimli İşlem (Interactive Processing)**

**Tanım:**Kullanıcının, işlemin sonucunu anında görebildiği ve işleme anında müdahale edebildiği yöntemdir.

**Benzetme:**

* Aşçı, bir restoranda siparişleri hazırlamaktadır.
* **Senaryo:**Siz masada menüye bakarsınız ve garsona "Bir pizza istiyorum" dersiniz (Bu bir komuttur).
  + Aşçı pizzayı yapmaya başlar.
  + Siz "Bir dakika, lütfen üzerine ekstra peynir ekler misiniz?" diye seslenirsiniz (Bu bir müdahaledir).
  + Aşçı işlemi durdurur, peyniri ekler ve pişirmeye devam eder.
  + Siparişiniz hazır olunca size getirilir (anında sonuç).

**Gerçek Hayat Örnekleri:**

* **Kelime İşlemci (Word) ile Yazı Yazmak**: Her harfe bastığınızda, harf anında ekranda belirir. Yazı tipini değiştirdiğinizde anında değişir.
* **Web'de Gezinmek (Surfing):**Bir linke tıkladığınızda, sayfa anında yüklenmeye başlar.
* **Bir Oyun Oynamak:**Klavyeye her bastığınızda, karakteriniz anında tepki verir.

**Özellikleri:**

* **Anında Yanıt (Low Latency) Esastır:**Buradaki en kritik kavram, işlemin başlangıcı ile sonucu arasındaki sürenin (gecikmenin) çok kısa olmasıdır.
* **Sürekli Etkileşim Vardır:**Kullanıcı sürekli komutlar verir, sistem sürekli yanıtlar.
* **Öncelik:**Yanıt Süresi (Response Time). İşi en hızlı şekilde bitirmek değil, kullanıcıya en hızlı şekilde dönüş yapmak önemlidir.

**Özet:** "Ben ne yaparsam, sen anında karşılık ver."

Bu iki kavram, aslında günlük hayatımızda sürekli iç içe geçmiş durumda:

* **Etkileşimli İşlem:** Siz bir sosyal medya uygulamasında fotoğraf seçip "Paylaş" butonuna tıklarsınız. Bu bir etkileşimli işlemdir; siz komut verdiniz, uygulama "Tamam!" dedi ve fotoğrafınız yüklenmeye başladı.
* **Toplu İşlem:** Arkadaşınız o fotoğrafa yorum yazdığında, uygulamanın sunucusu o yorumu hemen size bildirmez. Genellikle siz uygulamayı tekrar açtığınızda, biriken tüm bildirimleri (yorumlar, beğeniler) toplu halde görürsünüz. Bu da bir toplu işlem mantığıdır.

Kısacası, anında sonuç beklediğimiz her şey Etkileşimli İşlem, sonucun ne zaman geldiğinin çok önemli olmadığı, önemli olanın çok sayıda işin yapılması olduğu durumlar ise Toplu İşlemdir.

**Karşılaştırmalı Özet Tablosu**

| Özellik | Toplu İşlem | Etkileşimli İşlem |
| --- | --- | --- |
| Temel Mantık | "Listeyi Tamamla, Getir" | "Anında Tepki Ver" |
| Kullanıcı Etkileşimi | Yoktur. İşlem başlayınca müdahale edilmez. | Vardır. Sürekli etkileşim ve müdahale mümkündür. |
| Öncelik | İş Tamamlama (Throughput) | Yanıt Süresi (Response Time) |
| Zamanlama | Sistemin boş olduğu zamanlar | Kullanıcının istediği an |
| Girdi/Komut | Önceden hazırlanmış komut dosyaları | Kullanıcının anlık komutları (tıklama, yazma) |
| Örnek | Ay sonu bordrosu, fatura kesme | Video oyunu, web gezintisi, ofis programları |

**Gömülü İşletim Sistemleri: "Gizli Kahramanlar"**

**Kısa Tanım:** Gömülü işletim sistemleri, belirli bir amaca hizmet eden cihazların içine "gömülmüş", küçük, hızlı ve genellikle görünmez olarak çalışan özel işletim sistemleridir.

**Temel Felsefe: "Bir İşi, Mükemmel Şekilde Yapsın"**

Normal bir işletim sistemi (Windows, macOS) genel amaçlıdır; siz onu ofis programları, oyunlar, tasarım gibi birçok farklı iş için kullanırsınız. Gömülü işletim sistemleri ise özel amaçlıdır; sadece içine yerleştirildiği cihazın yapması gereken işe odaklanmıştır.

**Benzetme: "Akıllı Ev Aletleri vs Bilgisayar"**

* **Genel Amaçlı İşletim Sistemi (Windows/macOS):**Bir şefin mutfağı gibidir. Şef burada omlet yapabilir, pasta yapabilir, et pişirebilir. Çok yönlüdür.
* **Gömülü İşletim Sistemi:**Akıllı bir ekmek kızartma makinesi gibidir. Sadece ekmeği kızartmak için vardır. Bu işi çok iyi yapar, basittir ve hızlıdır. İçindeki mini "beyin" (gömülü işletim sistemi), sadece kızartma süresini, sıcaklığını ve ekmeğin nasıl ısınacağını bilir. Size bir film izletemez veya doküman yazdıramaz.

**Gömülü İşletim Sistemlerinin Özellikleri (Neden Farklılar?):**

1. **Kaynakları Sınırlıdır (Minimalisttir):**
   * Çok az RAM (bellek) ve depolama alanı ile çalışacak şekilde tasarlanmışlardır. Amacı, gigabytelarca dosya depolamak değil, birkaç temel görevi yerine getirmektir.
2. **Gerçek Zamanlı Olabilirler (Real-Time):**
   * Bir otomobildeki hava yastığı sensörünü düşünün. Bir kaza anında, "şimdi çalışsın" demez, "anında çalışır". Bu tür sistemler için **zamanlama hayati önem taşır**. Buna **Gerçek Zamanlı İşletim Sistemi (RTOS)** denir.
3. **Güvenilir ve Kararlıdır:**
   * Çamaşır makinenizin her açtığınızda "mavi ekran hatası" verdiğini hayal edin! Gömülü sistemler, uzun süre sorunsuz çalışacak şekilde tasarlanır.
4. **Enerji Verimlidir:**
   * Pil ile çalışan bir cihazda (akıllı saat, termostat) enerji tüketimi çok önemlidir. Gömülü işletim sistemleri bu yüzden çok düşük güç tüketir.
5. **Kullanıcı Arayüzü Basit veya Hiç Yoktur:**
   * Çoğu zaman bizimle doğrudan bir masaüstü veya pencere üzerinden değil, cihazın düğmeleri, ışıkları veya basit bir ekranı üzerinden etkileşime girer.

**Gömülü İşletim Sistemlerine Günlük Hayattan Örnekler:**

**"Aman Tanrım, Bunların Hepsinin İçinde Bir İşletim Sistemi Mi Var?" Dedirtecek Örnekler:**

* **Akıllı Saatler ve Giyilebilir Cihazlar: (Apple Watch, Fitbit)**
* **Akıllı TV'ler ve Medya Oynatıcılar: (Android TV, Tizen, webOS)**
* **Akıllı Ev Cihazları:**
  + **Akıllı Termostat (Nest):**Sıcaklığı öğrenir ve programlar.
  + **Akıllı Buzdolabı:**İçindekileri takip eder.
  + **Robot Süpürgeler (Roomba):**Oda haritası çıkarır ve temizlik rotası planlar.
* **Arabalar:**
  + Gösterge Paneli, hava yastığı kontrolü, ABS fren sistemi, eğlence-entertainment sistemi.
* **Akıllı Telefonlar:**Aslında Android ve iOS da birer gömülü işletim sistemidir! Çünkü onlar da belirli bir donanım (telefon) için optimize edilmişlerdir.
* **Endüstriyel Makineler:**Bir fabrikadaki robot kolu.
* **Medikal Cihazlar:**Kalp pili, insülin pompası.

**Popüler Gömülü İşletim Sistemi Örnekleri:**

* **FreeRTOS:**Küçük, hafif ve ücretsiz bir sistem. Akıllı ev cihazlarında çok yaygındır.
* **Embedded Linux:**Daha gelişmiş cihazlarda (akıllı TV'ler, router'lar) kullanılan, Linux çekirdeğine dayalı bir sistem.
* **Android Things (Artık kullanım dışı olsa da fikir vermek için):**IoT (Nesnelerin İnterneti) cihazları için Google'ın geliştirdiği bir sistemdi.
* **QNX:**Arabalardaki gösterge panelleri ve sistemlerde sıklıkla kullanılan, güvenilirliği yüksek bir sistem.

Gömülü işletim sistemleri, **Nesnelerin İnterneti (IoT)** ve **akıllı cihazlar devriminin** arkasındaki gizli kahramanlardır. Etrafımızdaki "akıllanan" her şeyin beynidir.

* **Yazılım Geliştirici Açısından:** Eğer yazılımla uğraşıyorsanız, geleceğin büyük bir kısmı bu "küçük" sistemlerde gizli. Donanıma yakın, kaynakları verimli kullanan yazılımlar yazmak giderek daha önemli hale geliyor.
* **Kullanıcı Açısından:** Bir sonraki "harika" akıllı cihazı tasarlamak, aslında onun için doğru gömülü işletim sistemini seçmek ve programlamakla başlar.

**Unutmayın:** Artık sadece bilgisayarların değil, neredeyse her şeyin bir işletim sistemi var. Ve bu sistemler, o cihazı sıradan bir makineden, "akıllı" bir cihaza dönüştürüyor.

**Gerçek Zamanlı İşletim Sistemleri: "Zamanın Efendileri"**

**Kısa Tanım:** Gerçek Zamanlı İşletim Sistemleri, bir işin **kesinlikle belirli bir süre içinde** tamamlanmasını garanti eden özel işletim sistemleridir. Burada önemli olan "hız" değil, **"tahmin edilebilirlik" ve "zamanlama garantisi"** dir.

**Temel Felsefe: "Ne Yaptığın Değil, Ne Zaman Yaptığın Önemli!"**

Normal bir işletim sisteminde (Windows, macOS) "bu iş en kısa sürede bitsin" hedeflenir. RTOS'ta ise "bu iş **en geç 5 milisaniye içinde** bitsin" hedeflenir ve bu bir **garantidir**.

**Benzetme: "Hava Yastığı vs. Video İndirme"**

* **Normal İşletim Sistemi:** **Bir video dosyası indirmek** gibidir. İndirme 10 saniye de sürebilir, 15 saniye de. Sonuçta video indilir ve izlenir. Burada "gecikme" can sıkıcı olsa da hayati değildir.
* **Gerçek Zamanlı İşletim Sistemi:** **Bir arabanın hava yastığı** gibidir. Bir kaza anında sensör çarpışmayı algılar ve hava yastığının **tam olarak belirlenen süre içinde (örneğin 20 milisaniyede)** şişmesi **ŞARTTIR**. 10 milisaniyede şişse harika, 30 milisaniyede şişse bu, kabul edilemez ve trajik bir hatadır.

**İki Temel Türü:**

**1. Sert Gerçek Zamanlı (Hard Real-Time) - "ASLA Geç Kalma!"**

* **Kural:** Zaman sınırının aşılması **tamamen başarısızlık** kabul edilir.
* **Motto:** "Ya zamanında, ya hiç."
* **Örnekler:**
  + Hava yastık sistemleri
  + Uçak kontrollu sistemleri
  + Kalp pilleri
  + Nükleer reaktör kontrolleri

**2. Esnek Gerçek Zamanlı (Soft Real-Time) - "Mümkünse Geç Kalma!"**

* **Kural:** Zaman sınırının aşılması **sistemin çökmesine neden olmaz** ancak performansı ve kaliteyi düşürür.
* **Motto:** "Zamanında olursan harika, biraz gecikirsen de idare eder."
* **Örnekler:**
  + Canlı video yayını (buffering olursa kalite düşer ama sistem çökmez)
  + Ses işleme yazılımları
  + Çevrimiçi oyunlar (gecikme olursa oyun deneyimi bozulur)

**RTOS'ların Temel Özellikleri:**

1. **Tahmin Edilebilirlik (Predictability):**
   * Bir işlemin en kötü ihtimalle ne kadar süreceği bellidir. ("Bu görev asla 2 ms'den fazla sürmeyecek")
2. **Kesin Zamanlama (Determinism):**
   * Aynı koşullar altında, aynı iş her zaman aynı sürede tamamlanır.
3. **Öncelik Tabanlı Zamanlama (Priority-Based Scheduling):**
   * En kritik görevler (acil durum freni), her zaman daha az kritik görevlerden (radyo değiştirme) önce işlenir.
4. **Kaynak Yönetimi:**
   * Bellek ve işlemci gücü gibi kaynaklar, kritik görevlerin aksamasına izin vermeyecek şekilde titizlikle yönetilir.
5. **Küçük Boyut ve Verimlilik:**
   * Çoğu RTOS, minimum kaynakla çalışacak şekilde tasarlanmıştır.

**Nerede Kullanılır? (Günlük Hayattan Örnekler)**

* **Otomotiv Endüstrisi:**Motor kontrol üniteleri, ABS fren sistemleri, sürücüsüz araç sistemleri.
* **Tıp Cihazları:**Kalp pilleri, insülin pompaları, tıbbi görüntüleme cihazları (MR, CT tarayıcılar).
* **Endüstriyel Otomasyon:**Robot kolları, üretim hatları, proses kontrol sistemleri.
* **Tüketici Elektroniği:**Dijital kameralar (oto-odaklama), yazıcılar, drone uçuş kontrolleri.
* **Telekomünikasyon:**Ağ anahtarlama ve yönlendirme ekipmanları.

**Popüler RTOS Örnekleri:**

* **FreeRTOS:**Dünyada en yaygın kullanılan, açık kaynaklı ve ücretsiz bir RTOS.
* **VxWorks:**Havacılık ve savunma sanayii gibi yüksek güvenlik gerektiren alanlarda kullanılan, endüstri standardı bir RTOS.
* **Zephyr:**Linux Vakfı tarafından desteklenen, modern ve ölçeklenebilir bir RTOS.
* **QNX:**Özellikle otomotiv endüstrisinde yaygın olarak kullanılan, güvenilirliği yüksek bir RTOS.

RTOS'lar, hata payının sıfır veya sıfıra çok yakın olması gereken yerlerde kullanılan özel sistemlerdir. Normal bir işletim sisteminin "en kısa sürede" felsefesine karşılık, RTOS'un felsefesi "en geç şu zamana kadar" dır.

**Yazılım Geliştirici Açısından:** RTOS programlamak, bir yarış pistinde F1 pilotu olmak gibidir. Her milisaniye hesap edilir, her hareket planlanır ve hata kabul edilmez. Eğer donanıma yakın, yüksek güvenilirlik gerektiren sistemler yazmak ilginizi çekiyorsa, RTOS dünyası sizin için heyecan verici bir alan olabilir.

**Unutmayın:** Hayatınızı doğrudan etkileyen birçok kritik teknolojinin arkasında, zamanla olan bu "sert" anlaşmasını bozmayan Gerçek Zamanlı İşletim Sistemleri vardır.

### **Ofis Uygulamaları**

### **Microsoft Word**

**1. Şeridi Özelleştirme (Ribbon Customization)**

* **Bu Nedir?:** Word'ün en üstündeki (Giriş, Ekle, Sayfa Düzeni... yazılı) menü çubuğuna "Şerit" denir. İstediğiniz araçları buraya ekleyip çıkarabilir, kendi ihtiyaçlarınıza göre kişiselleştirebilirsiniz.
* **Ne İşe Yarar?:** Sık kullandığınız düğmeleri (örneğin "PDF'e Kaydet" veya "WordArt") tek tıkla ulaşabileceğiniz bir yere koyarak zamandan tasarruf edersiniz. Sadece ihtiyacınız olan araçları görüntüleyerek ekranı sadeleştirirsiniz.
* **Nasıl Yapılır?:** Dosya > Seçenekler > Şeridi Özelleştir yolunu izleyebilirsiniz. Buradan yeni sekmeler oluşturabilir, mevcut sekmelere yeni komut grupları ekleyebilirsiniz. Ayrıca, şeridin sağ alt köşesindeki küçük ok işaretiyle şeridi simge durumuna küçültebilirsiniz.
* **Özet Açıklama:** Word'deki üst menü şeridi, kullanıcıya özel hale getirilebilir. Yeni sekmeler eklenebilir, sık kullanılan komutlar sabitlenebilir.
* **Görsel Önerisi:** Şerit üzerinde "Dosya > Seçenekler > Şeridi Özelleştir" ekran görüntüsü.

**2. Yazı Tipi Özellikleri (Font Properties)**

* **Bu Nedir?:** Metninizin görünümünü değiştiren temel araçlardır. Yazı tipi (Arial, Calibri, Times New Roman), punto (11, 12, 14), kalın/italik/altı çizili, renk, karakter aralığı gibi özellikleri içerir.
* **Ne İşe Yarar?:** Belgenizi görsel olarak düzenler, okunabilirliği artırır ve önemli noktaları vurgularsınız. Ödev başlıklarını büyük ve kalın yapmak, alıntıları italik yapmak gibi.
* **Nasıl Yapılır?:** Metni seçtikten sonra "Giriş" sekmesindeki "Yazı Tipi" grubundaki araçlarla tüm ayarlamaları yapabilirsiniz. Gelişmiş ayarlar için "Yazı Tipi" grubunun sağ alt köşesindeki küçük oka tıklayarak daha detaylı bir pencere açabilirsiniz.
* **Özet Açıklama:** Yazı tipi, boyutu, kalınlık, italik, altı çizili gibi biçimlendirme seçenekleriyle metin estetiği sağlanır.
* **Görsel Önerisi:** "Giriş" sekmesindeki yazı tipi araç çubuğu ekran görüntüsü.

**3. Paragraf Özelleştirme ve Madde İmleri & Numaralandırma**

* **Bu Nedir?:** Paragrafların satır aralığı, girinti (sağa/sola kaydırma), hizalama (sola, ortaya, iki yana yaslı) ve madde imleri/numaralandırma ile düzenlenmesidir.
* **Ne İşe Yarar?:** Metninizi daha düzenli, profesyonel ve takip edilmesi kolay hale getirir. Özellikle listelemelerde ve uzun paragraflarda okuma kolaylığı sağlar.
* **Nasıl Yapılır?:** "Giriş" sekmesindeki "Paragraf" grubundan hizalama, satır aralığı, girinti ve madde imleri/numaralandırma düğmelerine ulaşabilirsiniz. Tüm paragraf ayarları için "Paragraf" grubunun sağ alt köşesindeki oka tıklayın.
* **Özet Açıklama:** Paragraf hizalama, girinti, satır aralığı gibi ayarlar metnin okunabilirliğini artırır. Madde işaretleri ve numaralandırma ile listeler oluşturulabilir.
* **Görsel Önerisi:** "Giriş" sekmesinde paragraf ayarları ve madde imleri araçları.

**4. Stiller ve Fontlar (Styles and Fonts)**

* **Bu Nedir?:** "Stiller", başlık, alt başlık ve normal metin gibi önceden tanımlanmış biçimlendirme setleridir. "Fontlar" ise yazı tipleridir. Bu ikisini birlikte kullanarak tutarlı bir belge tasarımı oluşturursunuz.
* **Ne İşe Yarar?:** Stiller, tüm belgenizin görünümünü hızlıca değiştirmenizi sağlar. En önemlisi, **İçindekiler Tablosu** oluştururken Word, sadece "Başlık 1", "Başlık 2" gibi stillerle biçimlendirdiğiniz metinleri otomatik olarak algılar. Ayrıca, bir stili değiştirdiğinizde o stili kullanan tüm metinler otomatik olarak güncellenir.
* **Nasıl Yapılır?:** "Giriş" sekmesindeki "Stiller" galerisinden bir stil seçebilirsiniz. Bir stili özelleştirmek için stile sağ tıklayıp "Değiştir" diyebilirsiniz.
* **Açıklama:** Belgeye profesyonel görünüm kazandırmak için başlık, alt başlık gibi stiller kullanılır. Fontlar ile görsel kimlik oluşturulabilir.
* **Görsel Önerisi:** "Stiller" bölümü ve özel font seçimi ekran görüntüsü.

**5. Bul ve Değiştir (Find and Replace)**

* **Bu Nedir?:** Belge içinde bir kelimeyi veya ifadeyi aramanızı ve isterseniz başka bir kelimeyle toplu olarak değiştirmenizi sağlayan güçlü bir araçtır.
* **Ne İşe Yarar?:** 50 sayfalık bir ödevde "teknoloji" kelimesinin geçtiği tüm yerleri bulmak veya yanlış yazdığınız bir ismi (Ahmet -> Mehmet) tüm belgede düzeltmek için idealdir.
* **Nasıl Yapılır?:** "Giriş" sekmesinin en sağındaki "Değiştir" düğmesine tıklayın (veya kısayol: Ctrl + H). "Bul" kısmına aradığınız, "Değiştir" kısmına ise yeni kelimeyi yazın. "Tümünü Değiştir" diyerek işlemi toplu yapabilirsiniz.
* **Özet Açıklama:** Belgedeki kelimeleri veya ifadeleri hızlıca bulup topluca değiştirmek için kullanılır.
* **Görsel Önerisi:** "Giriş > Düzenleme > Bul / Değiştir" ekran görüntüsü.

**6. Resim Ekleme Seçenekleri ve Düzenleme**

* **Bu Nedir?:** Belgenize resim, şekil, simge, grafik gibi görsel öğeler eklemek ve bunların boyut, konum, metin kaydırma, efektler gibi özelliklerini düzenlemektir.
* **Ne İşe Yarar?:** Belgenizi görsel açıdan zenginleştirir, anlatılmak isteneni daha iyi ifade eder. Ödevlerinizde, raporlarınızda görsel destek sağlar.
* **Nasıl Yapılır?:** Resmi eklemek için "Ekle" sekmesinden "Resimler"i seçin. Resme tıkladığınızda üstte "Resim Biçimi" sekmesi çıkar. Buradan "Metin Kaydırma" (resmin yazının neresinde duracağı), "Sanatsal Efektler", "Kenarlık" gibi birçok ayarı yapabilirsiniz.
* **Özet Açıklama:** Belgeye görsel eklemek anlatımı zenginleştirir. Resimler konumlandırılabilir, boyutlandırılabilir, metinle uyumlu hale getirilebilir.
* **Görsel Önerisi:** "Ekle > Resimler" ve resim biçimlendirme araçları ekran görüntüsü.

**7. Tablolar ile Çalışmak (Working with Tables)**

* **Bu Nedir?:** Satır ve sütunlardan oluşan, verileri düzenli bir şekilde göstermenizi sağlayan yapılardır.
* **Ne İşe Yarar?:** Not listesi, karşılaştırma tablosu, ders programı veya sayısal verileri göstermek için kullanılır. Okunması kolay ve düzenli bir sunum sağlar.
* **Nasıl Yapılır?:** "Ekle" sekmesinden "Tablo"yu seçin. Fareyle sürükleyerek boyutunu belirleyebilir veya "Tablo Ekle" diyerek satır/sütun sayısını manuel girebilirsiniz. Tabloyu seçtiğinizde üstte "Tasarım" ve "Düzen" sekmeleri çıkar; buradan renk, stil, kenarlık ve hücre birleştirme/bölme işlemlerini yapabilirsiniz.
* **Özet Açıklama:** Verileri düzenli sunmak için tablo eklenebilir. Hücre biçimlendirme, kenarlıklar ve gölgelendirme ile görsellik artırılır.
* **Görsel Önerisi:** "Ekle > Tablo" ve tablo düzenleme araçları ekran görüntüsü.

**8. Açıklama ve Yorum Ekleme (Comments)**

* **Bu Nedir?:** Belgenin herhangi bir yerine, metni değiştirmeden, bir not veya soru ekleme özelliğidir.
* **Ne İşe Yarar?:** Grup projelerinde arkadaşınıza "Bu kısmı açıklar mısın?" gibi bir soru sorabilir veya öğretmeniniz ödevinize "Burayı geliştirebilirsin" gibi bir yorum bırakabilir. Metnin kendisini değiştirmez, kenar boşluğunda bir sohbet başlatır.
* **Nasıl Yapılır?:** Yorum eklemek istediğiniz metni seçin, "Gözden Geçir" sekmesinden "Yeni Açıklama" düğmesine tıklayın. Açıklamayı yazın. Yanıtlayabilir, çözüldü olarak işaretleyebilirsiniz.
* **Özet Açıklama:** Grup çalışmalarında belgeye yorum eklemek geri bildirim sağlar. Değişiklik izleme ile kim neyi düzenlemiş görülebilir.
* **Görsel Önerisi:** "Gözden Geçir > Yeni Yorum" ve yorum balonları ekran görüntüsü.

**9. İçindekiler Tablosu Oluşturmak ve Güncellemek (Table of Contents)**

* **Bu Nedir?:** Belgenin başında, başlık ve sayfa numaralarından oluşan otomatik bir indekstir.
* **Ne İşe Yarar?:** Uzun belgelerde (bitirme tezi, kitap, uzun rapor) okuyucunun istediği bölüme hızlıca atlamasını sağlar. Çok profesyonel bir görünüm kazandırır.
* **Nasıl Yapılır?:** **ÖNEMLİ:** Önce tüm başlıklarınızı "Başlık 1", "Başlık 2" gibi stillerle biçimlendirin. Sonra, "Başvurular" sekmesine gidin, "İçindekiler Tablosu"na tıklayın ve hazır bir şablon seçin. Otomatik olarak oluşacaktır. Bir şey değiştirdiğinizde tablonun üzerine tıklayıp "Tabloyu Güncelleştir" diyerek yenilersiniz.
* **Özet Açıklama:** Belgenin başlık yapısına göre otomatik içerik tablosu oluşturulabilir. Sayfa numaralarıyla birlikte güncellenebilir.
* **Görsel Önerisi:** "Başvurular > İçindekiler Tablosu" ekran görüntüsü.

**10. Belgeyi Koruma Altına Almak (Protect Document)**

* **Bu Nedir?:** Belgenize şifre koymak veya başkalarınin belge üzerinde yapabileceği değişiklikleri kısıtlamaktır.
* **Ne İşe Yarar?:** Kişisel veya önemli belgelerinizi yetkisiz kişilerden korur. Bir belgeyi "Salt Okunur" yaparak sadece okunmasını, düzenlenememesini sağlayabilirsiniz.
* **Nasıl Yapılır?:** Dosya > Bil > Belgeyi Koru seçeneğine gidin. Buradan:
  + **Parola ile Şifrele:** Belgeyi açmak için şifre ister.
  + **Düzenlemeyi Kısıtla:** İnsanların sadece yorum eklemesine izin verebilir veya değişiklikleri izleme (Track Changes) modunda çalışmaya zorlayabilirsiniz.
* **Özet Açıklama:** Belgeye parola koyarak yetkisiz erişim engellenebilir. Salt okunur hale getirilebilir.
* **Görsel Önerisi:** "Dosya > Bilgi > Belgeyi Koru" ekran görüntüsü.

### **Microsoft Excel**

**1. Veri Girişi ve Veri Değişikliği**

* **Bu Nedir?:** Excel'de hücrelere metin, sayı, tarih gibi bilgileri girme ve bu bilgileri sonradan düzenleme işlemidir.
* **Ne İşe Yarar?:** Excel'in temelidir. Alışveriş listesi, not çizelgesi, arkadaşlarınızın iletişim bilgileri gibi her türlü bilgiyi düzenli bir şekilde kaydetmeye yarar.
* **Nasıl Yapılır?:** Bir hücreye tıklayıp yazmaya başlayın. Değişiklik için hücreye çift tıklayın veya "F2" tuşuna basın. Bir hücreyi veya aralığı seçip "Delete" tuşuyla içeriğini silebilirsiniz.
* **Özet Açıklama:** Hücreye veri girmek için hücreye tıklayıp yazmak yeterlidir. Değişiklik için hücreye çift tıklanabilir veya formül çubuğu kullanılabilir.
* **Görsel Önerisi:** Hücreye metin ve sayı girilmiş örnek tablo.

**2. Otomatik Tamamlama**

* **Bu Nedir?:** Bir sütuna ardışık veriler (1, 2, 3... veya Pazartesi, Salı, Çarşamba...) girdiğinizde, Excel'in devamını tahmin edip otomatik doldurma teklif etmesidir.
* **Ne İşe Yarar?:** Uzun listeleri elle girmekten kurtarır, zaman kazandırır ve hata yapma ihtimalini azaltır.
* **Nasıl Yapılır?:** İlk birkaç değeri yazın (Örn: "Ocak", "Şubat"). Hücrenin sağ alt köşesindeki küçük noktadan (doldurma kulpu) tutup aşağı veya sağa doğru sürükleyin. Excel otomatik olarak "Mart", "Nisan" şeklinde doldursun.
* **Özet Açıklama:** Excel, önceki hücrelerdeki verileri algılayarak sonraki hücreleri otomatik tamamlayabilir.
* **Görsel Önerisi:** Aynı kelimenin birkaç hücrede otomatik tamamlandığı örnek.

**3. Hücreleri Biçimlendirme**

* **Bu Nedir?:** Hücrelerdeki verilerin görünümünü değiştirmektir. Yazı tipi, renk, kenarlık, hizalama ve en önemlisi **sayı biçimi** (para birimi, tarih, yüzde) gibi ayarları içerir.
* **Ne İşe Yarar?:** Verileri anlamlı hale getirir. Örneğin, bir hücreye 15 yazıp onu "Para Birimi" yaparsanız ₺15,00 görünür. Bu, o sayının bir fiyat olduğunu anlamanızı sağlar.
* **Nasıl Yapılır?:** Hücreleri seçin, "Giriş" sekmesindeki "Sayı" grubundaki açılır menüden (Genel yazan yer) para birimi, tarih, yüzde gibi formatları seçebilirsiniz.
* **Özet Açıklama:** Hücrelerde yazı tipi, renk, kenarlık, hizalama gibi görsel düzenlemeler yapılabilir.
* **Görsel Önerisi:** "Giriş" sekmesindeki biçimlendirme araçları ekran görüntüsü.

**4. Metni Sütunlara Böl**

* **Bu Nedir?:** Tek bir sütunda birleşik duran verileri (Ad Soyad, Şehir/İlçe gibi) mantıklı parçalara bölerek ayrı sütunlara ayırma sihirbazıdır.
* **Ne İşe Yarar?:** Dışarıdan aktarılan verileri düzenlemek için çok kullanışlıdır. Örneğin, "Ahmet Yılmaz" yazan bir sütunu "Ad" ve "Soyad" olarak iki ayrı sütuna bölebilirsiniz.
* **Nasıl Yapılır?:** Bölmek istediğiniz sütunu seçin. "Veri" sekmesinden "Metni Sütunlara Dönüştür"ü seçin. Sihirbaz sizi yönlendirecektir. Ayırıcı olarak genellikle "boşluk" (space) veya "virgül" seçilir.
* **Özet Açıklama:** Tek hücredeki metni belirli ayırıcılarla (virgül, boşluk) birden fazla sütuna ayırmak için kullanılır.
* **Görsel Önerisi:** "Veri > Metni Sütunlara Dönüştür" ekran görüntüsü.

**5. Çıktı Alma ve Görünüm Ayarları**

* **Bu Nedir?:** Excel tablonuzu kağıda düzgün bir şekilde yazdırmak için yapılan ayarlardır. Sayfa sonları, kenar boşlukları, üstbilgi/altbilgi ve başlıkların her sayfada tekrarlanması gibi özellikleri içerir.
* **Ne İşe Yarar?:** Kocaman bir tabloyu yazdırdığınızda son sütunun veya başlıkların ikinci sayfada kalmasını engeller. Profesyonel ve okunabilir çıktılar almanızı sağlar.
* **Nasıl Yapılır?:** "Sayfa Düzeni" sekmesinden ve "Dosya > Yazdır" menüsünden ayarlara ulaşabilirsiniz. Başlıkları sabitlemek için "Sayfa Düzeni > Yazdırma Başlıkları"na gidip "Satırları tekrarla" alanını seçin.
* **Açıklama:** Sayfa düzeni, kenar boşlukları, yazdırma alanı gibi ayarlar çıktı kalitesini etkiler.
* **Görsel Önerisi:** "Sayfa Düzeni" sekmesi ve yazdırma önizleme ekranı.

**6. Temel Matematiksel Fonksiyonlar**

* **Bu Nedir?:** Toplama, çıkarma, ortalama, en büyük/en küçük değer bulma gibi temel hesaplamaları yapan formüllerdir.
* **Ne İşe Yarar?:** Elle hesaplama yapmaktan kurtarır. Not ortalaması hesaplama, aylık harcamaların toplamını bulma gibi işlemlerde kullanılır.
* **Nasıl Yapılır?:**
  + **TOPLA:** =TOPLA(A1:A10) -> A1'den A10'a kadar olan hücreleri toplar.
  + **ORTALAMA:** =ORTALAMA(B2:B20) -> B2'den B20'ye kadar olan hücrelerin ortalamasını alır.
  + **MAK/MİN:** =MAK(C1:C100) -> C sütunundaki en yüksek değeri bulur.
* **Özet Açıklama:** Toplama (SUM), Ortalama (AVERAGE), En Büyük (MAX), En Küçük (MIN) gibi işlemler yapılabilir.
* **Görsel Önerisi:** Fonksiyonların kullanıldığı örnek tablo.
* **Matematiksel Fonksiyonlar - Gelişmiş Örnekler**
* **1. ETOPLA (SUMIF) - Koşullu Toplama**
* **Örnek:** Sadece "Bilgisayar" kategorisindeki ürünlerin satışlarını topla.
* =ETOPLA(A2:A100; "Bilgisayar"; C2:C100)
* **Mantık:** Aralık (Kategori sütunu), Ölçüt ("Bilgisayar"), Toplam\_aralığı (Satış Tutarı sütunu). Yani "Kategorisi Bilgisayar olanların, Satış Tutarlarını topla."
* **2. ÇOKEĞERTOPLA (SUMIFS) - Çoklu Koşullu Toplama**
* **Örnek:** "Bilgisayar" kategorisinden ve "İstanbul" bölgesinden yapılan satışları topla.
* =ÇOKEĞERTOPLA(C2:C100; A2:A100; "Bilgisayar"; B2:B100; "İstanbul")
* **Mantık:** Toplam\_aralığı (Satış Tutarı), Koşul\_aralığı1 (Kategori), Koşul1 ("Bilgisayar"), Koşul\_aralığı2 (Bölge), Koşul2 ("İstanbul").
* **3. EĞERSAY (COUNTIF) - Koşullu Saydırma**
* **Örnek:** Listedeki "Tamamlandı" durumundaki işlerin sayısını bul.
* =EĞERSAY(B2:B100; "Tamamlandı")
* **Örnek 2:** 50'den yüksek not alan kaç kişi var?
* =EĞERSAY(C2:C50; ">50")
* **4. YUVARLA (ROUND) Ailesi**
* **Örnek:** Kuruş hesaplarında en yakın kuruşa yuvarlama.
* =YUVARLA(A2; 2) // 15,567 -> 15,57 olur.
* =AŞAĞIYUVARLA(A2; 2) // 15,567 -> 15,56 olur (Her zaman aşağı).
* =YUKARIYUVARLA(A2; 2) // 15,561 -> 15,57 olur (Her zaman yukarı).
* =TAMYUVARLA(15,567; 0,05) // 15,567 -> 15,55 olur (5 kuruşun katlarına yuvarlar).
* **5. BÖLÜM (QUOTIENT) & KALAN (MOD)**
* **Örnek:** 125 dakikanın kaç saat ve kaç dakika olduğunu hesapla.
* Saat = BÖLÜM(125; 60) // Sonuç: 2 (125/60'ın tam kısmı)
* Dakika = KALAN(125; 60) // Sonuç: 5 (125/60'ın kalanı)
* **6. EĞERORTALAMA (AVERAGEIF) - Koşullu Ortalama**
* **Örnek:** Sadece "Kadın" müşterilerin yaş ortalamasını al.
* =EĞERORTALAMA(C2:C100; "Kadın"; B2:B100)
* **Mantık:** Ortalama\_aralığı (Yaş sütunu), Ölçüt\_aralığı (Cinsiyet sütunu), Ölçüt ("Kadın").
* **7. RASTGELEARADA (RANDBETWEEN)**
* **Örnek:** 1 ile 90 arasında rastgele sayı üret (örneğin loto için tahmin).
* =RASTGELEARADA(1; 90)

**Gerçek Hayat Senaryosu: Kombinasyon Örneği**

**Senaryo:** Bir mağaza için basit bir performans raporu hazırlıyorsunuz.

|  | A | B | C | D | E |
| --- | --- | --- | --- | --- | --- |
| **1** | **Ürün** | **Bölge** | **Satış** | **Hedef** | **Durum** |
| **2** | Bilgisayar | İstanbul | 15.000 ₺ | 10.000 ₺ | ? |
| **3** | Telefon | Ankara | 8.000 ₺ | 12.000 ₺ | ? |

**"Durum" sütunu için formül (**E2**hücresi):**

=EĞER(VE(C2>=D2; A2="Bilgisayar"); "Prim Kazandı!"; EĞER(C2>=D2; "Hedefe Ulaştı"; "Hedefin Altında"))

* **Mantığın Adım Adım Açıklaması:**
  1. **İlk Kontrol (**VE(C2>=D2; A2="Bilgisayar")**):** Satış hedefi tutturulmuş MU (C2>=D2) VE ürün Bilgisayar MI (A2="Bilgisayar")?
     + **Evetse:** "Prim Kazandı!" yaz. (Çünkü bilgisayar satışlarına ek prim var).
     + **Hayırsa:** İkinci kontrole geç.
  2. **İkinci Kontrol (**C2>=D2**):** Satış hedefi tutturulmuş MU?
     + **Evetse:** "Hedefe Ulaştı" yaz.
     + **Hayırsa:** "Hedefin Altında" yaz.

Bu örnekte, mantıksal fonksiyonları (EĞER, VE) kullanarak karmaşık bir iş kuralını Excel'in anlayabileceği basit bir dile çevirdik.

**7. Hücre Sabitleme (Mutlak Referans - $ İşareti)**

* **Bu Nedir?:** Bir formülü kopyaladığınızda, formül içindeki belirli bir hücre adresinin sabit kalmasını sağlamaktır.
* **Ne İşe Yarar?:** Örneğin, tüm ürünlerin fiyatını sabit bir KDV oranı (%18) ile çarpmak istiyorsunuz. KDV'nin yazılı olduğu hücreyi sabitlemezseniz, formülü aşağı çektiğinizde KDV hücresi de kayar ve hata olur.
* **Nasıl Yapılır?:** Formülde sabitlemek istediğiniz hücre adresinin satır ve sütununa $ işareti koyun. =B2\*$C$1. F4 tuşu bu işlemi hızlı yapmanızı sağlar.
* **Özet Açıklama:** Belirli hücreleri sabitleyerek formüllerde sabit referans olarak kullanmak için $ işareti kullanılır.
* **Görsel Önerisi:** $A$1 gibi sabit hücre referansı örneği.

**8. Gelişmiş Filtreleme**

* **Bu Nedir?:** Basit filtrelemeden daha gelişmiş, birden fazla karmaşık koşula göre veri süzme yöntemidir.
* **Ne İşe Yarar?:** Örneğin, "İstanbul'da oturan ve yaşı 20'den büyük olan müşterileri" veya "Matematik notu 80'den fazla **VE** Fizik notu 70'ten az olan öğrencileri" listelemek için kullanılır.
* **Nasıl Yapılır?:** Önce verinizin olduğu aralıktaki başlıkları aynen kopyalayıp boş bir alana yapıştırın. Bu altına filtre koşullarınızı yazın (örn: Şehir sütununun altına "İstanbul", Yaş sütununun altına ">20"). Sonra "Veri > Sırala ve Filtre Uygula > Gelişmiş"i seçin.
* **Açıklama:** Belirli kriterlere göre veri süzmek için gelişmiş filtreleme kullanılır.
* **Görsel Önerisi:** "Veri > Gelişmiş" filtre ekranı.

**9. Mantıksal Fonksiyonlar (EĞER, VE, YADA)**

* **Bu Nedir?:** Belirli koşulları kontrol edip, sonuca göre farklı işlemler yapan akıllı formüllerdir.
* **Ne İşe Yarar?:** Otomatik karar vermeyi sağlar.
  + **EĞER:** =EĞER(B2>=50; "Geçti"; "Kaldı") -> Not 50 ve üzeriyse "Geçti", değilse "Kaldı" yazar.
  + **VE:** =EĞER(VE(B2>=50, C2>=50); "Geçti"; "Kaldı") -> İki dersin de notu 50'den yüksekse geçer.
  + **YADA:** Koşullardan herhangi biri doğruysa "DOĞRU" döndürür.
* **Özet Açıklama:** Belirli koşullara göre işlem yapmak için mantıksal fonksiyonlar kullanılır.
* **Görsel Önerisi:** =EĞER(A1>10;"Geçti";"Kaldı") gibi örnek formül.
* **Mantıksal Fonksiyonlar - Gelişmiş Örnekler**
* **1. İÇ İÇE EĞER (Nested IF)**
* **Örnek:** Bir sınav notuna göre harf notu vermek.
* =EĞER(B2>=90; "AA"; EĞER(B2>=80; "BA"; EĞER(B2>=70; "BB"; EĞER(B2>=60; "CB"; EĞER(B2>=50; "CC"; "FF")))))
* **Mantık:** "Eğer not 90'dan büyük eşitse AA, DEĞİLSE eğer 80'den büyük eşitse BA, DEĞİLSE eğer..." şeklinde ilerler. Formülü yazarken en dar aralıktan en geniş aralığa doğru gidin.
* **2. EĞER & VE (IF & AND) - Çoklu Koşul**
* **Örnek:** Stok ve fiyat kontrolü. Bir ürünün "Uygun" olması için hem stokta olması (>0) hem de fiyatının girilmiş olması gerekir.
* =EĞER(VE(C2>0; B2<>""); "Uygun"; "Uygun Değil")
* **Mantık:** **VE** fonksiyonu, içindeki *tüm* koşullar sağlanıyorsa DOĞRU döndürür. Sadece o zaman "Uygun" yazar.
* **3. EĞER & YADA (IF & OR) - Şartlardan Biri**
* **Örnek:** Öğrenci devamsızlık durumu. Devamsızlığı 10 günden fazla VEYA not ortalaması 50'nin altındaysa "Uyarı" ver.
* =EĞER(YADA(C2>10; B2<50); "UYARI"; "Normal")
* **Mantık:** **YADA** fonksiyonu, içindeki koşullardan *herhangi biri* sağlanıyorsa DOĞRU döndürür.
* **4. ÇOKEĞER (IFS) - Daha Modern Yöntem**
* **Örnek:** Yukarıdaki iç içe EĞER örneğinin daha temiz hali.
* =ÇOKEĞER(B2>=90; "AA"; B2>=80; "BA"; B2>=70; "BB"; B2>=60; "CB"; B2>=50; "CC"; DOĞRU; "FF")
* **Mantık:** "Koşul1, Sonuç1, Koşul2, Sonuç2..." şeklinde gider. DOĞRU en sonunda "hiçbiri tutmazsa bunu yap" anlamına gelir. Daha okunaklıdır.
* **5. EĞERHATA (IFERROR) - Hataları Gizleme**
* **Örnek:** Bir formül (örneğin bölme işlemi) hata verebiliyorsa ve hatayı daha anlamlı bir mesajla değiştirmek istiyorsanız.
* =EĞERHATA(A2/B2; "Hesaplanamadı")
* **Mantık:** A2/B2 formülü çalışırsa sonucu gösterir (örn: B2 boşsa #SAYI/0! hatası verir). Hata verirse "Hesaplanamadı" yazar. Kullanıcı dostu formüller için çok önemlidir.

**10. Koşullu Biçimlendirme**

* **Bu Nedir?:** Hücrelere, içlerindeki değerlere göre otomatik olarak renk, ikon veya çubuk gibi biçimler vermektir.
* **Ne İşe Yarar?:** Veri içindeki önemli eğilimleri ve değerleri görsel olarak vurgular. En yüksek notları yeşile, en düşük notları kırmızıya boyamak gibi.
* **Nasıl Yapılır?:** Biçimlendirmek istediğiniz hücreleri seçin. "Giriş" sekmesinden "Koşullu Biçimlendirme"ye tıklayın. Renk ölçekleri, veri çubukları veya hücre kurallarını seçin.
* **Açıklama:** Belirli kurallara göre hücrelerin rengi veya biçimi otomatik değiştirilebilir.
* **Görsel Önerisi:** "Giriş > Koşullu Biçimlendirme" ekran görüntüsü.
* **Koşullu Biçimlendirme Örneği: Öğrenci Not Takip Çizelgesi**
* **Senaryo:** 10 öğrencilik bir sınıfın sınav notlarını görselleştirmek istiyoruz.

| **1** | **Ad** | **Not** | **Durum** | **Performans** |
| --- | --- | --- | --- | --- |
| **2** | Ahmet | 85 | Geçti | İyi |
| **3** | Ayşe | 42 | Kaldı | Zayıf |
| **4** | Mehmet | 92 | Geçti | Mükemmel |
| **5** | Fatma | 78 | Geçti | Orta |
| **6** | Ali | 55 | Geçti | Orta |
| **7** | Zeynep | 36 | Kaldı | Zayıf |
| **8** | Can | 68 | Geçti | Orta |
| **9** | Demet | 95 | Geçti | Mükemmel |
| **10** | Burak | 29 | Kaldı | Zayıf |
| **11** | Ceren | 88 | Geçti | İyi |

* **Uygulayacağımız Koşullu Biçimlendirme Kuralları:**
* **1. Renk Ölçekleri - Notlara Gradyan Uygulama**
* **Ne Yapar?:** Notları düşükten yükseğe doğru kırmızıdan sarıya, yeşile geçen bir renk skalasıyla boyar.
* **Nasıl Yapılır?**
* B2:B11 aralığını seç (Notların olduğu hücreler).
* "Giriş" > "Koşullu Biçimlendirme" > "Renk Ölçekleri"
* "Kırmızı - Sarı - Yeşil" renk ölçeğini seç.
* **Sonuç:** 29 (Koyu Kırmızı), 95 (Koyu Yeşil) ve diğer notlar da değerlerine göre ara tonlarda renklenecek.
* **2. Veri Çubukları - Notları Görsel Çubuklarla Gösterme**
* **Ne Yapar?:** Hücrelerin içinde, not değeriyle orantılı uzunlukta yatay çubuklar oluşturur.
* **Nasıl Yapılır?**
* Yine B2:B11 aralığını seç.
* "Giriş" > "Koşullu Biçimlendirme" > "Veri Çubukları"
* İstediğiniz bir gradyan veya düz renk stilini seçin.
* **Sonuç:** En yüksek not olan 95, hücrenin neredeyse tamamını dolduracak. En düşük not 29 ise çok kısa bir çubuk olarak görünecek.
* **3. Hücre Kuralları - Belirli Koşulları Vurgulama**
* **Kural 1: 50'nin Altındaki Notları Kırmızı ve Kalın Yap**
* **Ne Yapar?:** Dersten kalan öğrencileri net bir şekilde vurgular.
* **Nasıl Yapılır?**
* B2:B11 aralığını seç.
* "Giriş" > "Koşullu Biçimlendirme" > "Hücre Kurallarını Vurgula" > "Küçüktür"
* Değer olarak 50 yaz.
* "Biçimlendir" seçeneğinden dolgu rengini "Açık Kırmızı", yazı tipini "Kalın" ve rengini "Koyu Kırmızı" yap.
* **Kural 2: 85 ve Üzeri Notları Yeşil ve İtalik Yap**
* **Ne Yapar?:** Üstün başarı gösteren öğrencileri belirler.
* **Nasıl Yapılır?**
* Aynı aralığı seç (B2:B11).
* "Giriş" > "Koşullu Biçimlendirme" > "Hücre Kurallarını Vurgula" > "Büyüktür"
* Değer olarak 85 yaz.
* "Biçimlendir" seçeneğinden dolgu rengini "Açık Yeşil", yazı tipini "İtalik" yap.
* **4. Simge Kümeleri - Performans Durumunu Görselleştirme**
* **Ne Yapar?:** "Performans" sütunundaki yazılı değerleri yeşil onay, sarı ünlem, kırmızı çarpı gibi simgelere dönüştürür.
* **Nasıl Yapılır?**
* D2:D11 aralığını seç (Performans sütunu).
* "Giriş" > "Koşullu Biçimlendirme" > "Simge Kümeleri"
* "Şekil" kategorisinden yeşil daire, sarı elmas, kırmızı üçgen gibi bir set seçin.
* **Sonuç:**
* "Mükemmel" ve "İyi" -> Yeşil daire
* "Orta" -> Sarı elmas
* "Zayıf" -> Kırmızı üçgen

**Tüm Kurallar Uygulandıktan Sonra Tablonun Görünümü:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Ad** | **Not** | **Durum** | **Performans** |
| 1 | Ahmet | 🟦 ████████ 85 | Geçti | ✅ İyi |
| 2 | Ayşe | 🟥 ██ 42 | Kaldı | 🔶 Zayıf |
| 3 | Mehmet | 🟩 █████████ 92 | Geçti | ✅ Mükemmel |
| 4 | Fatma | 🟦 ███████ 78 | Geçti | 🔶 Orta |
| 5 | Ali | 🟦 ██████ 55 | Geçti | 🔶 Orta |
| 6 | Zeynep | 🟥 █ 36 | Kaldı | 🔶 Zayıf |
| 7 | Can | 🟦 ███████ 68 | Geçti | 🔶 Orta |
| 8 | Demet | 🟩 █████████ 95 | Geçti | ✅ Mükemmel |
| 9 | Burak | 🟥 █ 29 | Kaldı | 🔶 Zayıf |
| 10 | Ceren | 🟩 ████████ 88 | Geçti | ✅ İyi |

**Açıklama:**

* **🟥🟦🟩:** Renk ölçeğini temsil eder (Kırmızı -> Mavi -> Yeşil tonları).
* **████:** Veri çubuklarını temsil eder.
* **Kalın & Kırmızı:** 50'nin altındaki notlar.
* **İtalik & Yeşil:** 85 ve üzeri notlar.
* **✅🔶:** Simge kümelerini temsil eder.

**11. Hızlı Doldurma (Flash Fill)**

* **Bu Nedir?:** Excel'in sizin yaptığınız bir deseni algılayıp, kalan verileri otomatik olarak aynı şekilde doldurmasıdır.
* **Ne İşe Yarar?:** Örneğin, "Ahmet Yılmaz" yazan bir sütundan sadece isimleri ("Ahmet") ayıklamak istiyorsunuz. İlk hücreye "Ahmet" yazıp Ctrl + E (Hızlı Doldurma) tuşuna basarsanız, Excel diğer tüm isimleri otomatik olarak doldurur.
* **Nasıl Yapılır?:** Deseni elle bir-iki hücrede oluşturun. Hemen yanındaki hücreyi seçin ve Ctrl + E tuşlarına basın.
* **Açıklama:** Excel, örüntüleri algılayarak hücreleri otomatik doldurabilir.
* **Görsel Önerisi:** Ad-soyad ayrımı veya e-posta oluşturma gibi örnekler.

**12. Tarihsel Fonksiyonlar**

* **Bu Nedir?:** Tarih ve saatle ilgili hesaplamalar yapan formüllerdir.
* **Ne İşe Yarar?:** İki tarih arasındaki gün sayısını bulma, bir tarihe belirli bir gün sayısı ekleyerek vade tarihi hesaplama gibi işlemlerde kullanılır.
* **Nasıl Yapılır?:**
  + **BUGÜN:** =BUGÜN() -> Bilgisayarın tarihini verir.
  + **TAMİŞGÜNÜ:** =TAMİŞGÜNÜ(BAŞLANGIÇ\_TARİHİ; BİTİŞ\_TARİHİ) -> İki tarih arasındaki iş günü sayısını hesaplar.
* **Özet Açıklama:** Tarih hesaplamaları için BUGÜN(), YIL(), AY() gibi fonksiyonlar kullanılır.
* **Görsel Önerisi:** Tarih fonksiyonlarının uygulandığı tablo.

**13. Grafikler**

* **Bu Nedir?:** Sayısal verileri görsel bir forma (çubuk, pasta, çizgi gibi) dönüştürme aracıdır.
* **Ne İşe Yarar?:** Verilerdeki trendleri, karşılaştırmaları ve oranları anlamayı çok daha kolay hale getirir. Sunumlar için vazgeçilmezdir.
* **Nasıl Yapılır?:** Grafiğe dönüştürmek istediğiniz veri aralığını seçin. "Ekle" sekmesinden istediğiniz grafik türünü (Örn: Sütun, Çizgi, Pasta) seçin.
* **Özet Açıklama:** Verileri görselleştirmek için sütun, çizgi, pasta gibi grafikler oluşturulabilir.
* **Görsel Önerisi:** "Ekle > Grafik" ekran görüntüsü.
* **Örnek Veri Seti: "Cem'in Aylık Harcamaları"**

|  |  |  |
| --- | --- | --- |
|  | **Harcama Kategorisi** | **Tutar (₺)** |
| **1** | Kira | 3000 |
| **2** | Gıda | 1500 |
| **3** | Ulaşım | 600 |
| **4** | Eğlence | 400 |
| **5** | Telefon/İnternet | 350 |
| **6** | Diğer | 750 |
|  | **TOPLAM** | **6600** |

**a. PASTA GRAFİK ÖRNEĞİ**

* **Grafik Başlığı: "Cem'in Aylık Harcama Dağılımı"**
* **Bu Grafik Ne Anlatıyor?**
* Pasta grafik, bir bütünün (toplam harcamanın) parçalara (harcama kategorilerine) nasıl ayrıldığını gösterir. Her dilim, toplam içindeki payı temsil eder.
* **Hangi Soruları Cevaplar?**
* **"En büyük harcama kalemi hangisi?"** → **Kira** (En büyük dilim, yaklaşık %45)
* **"Gelirimin ne kadarını kiraya veriyorum?"** → 3000 / 6600 ≈ **%45**
* **"Eğlence için bütçemin ne kadarı gidiyor?"** → 400 / 6600 ≈ **%6** (En küçük dilimlerden biri)
* **Ne Zaman Kullanılır?**
* **Oranları** ve **parça-bütün ilişkisini** göstermek istediğinizde
* Çok fazla veri dilimi olmadığında (genelde 5-7'den fazla dilim okuması zorlaştırır)
* **Toplamın** anlamlı olduğu durumlarda (Bu örnekte toplam harcama 6600₺)

**b. ÇİZGİ GRAFİK ÖRNEĞİ**

* Aynı kişi için farklı bir veri seti kullanalım:
* **Veri Seti: "Cem'in Yıllık Elektrik Faturası"**

|  |  |  |
| --- | --- | --- |
|  | **Ay** | **Fatura (₺)** |
| **1** | Ocak | 320 |
| **2** | Şubat | 280 |
| **3** | Mart | 190 |
| **4** | Nisan | 120 |
| **5** | Mayıs | 90 |
| **6** | Haziran | 85 |
| **7** | Temmuz | 110 |
| **8** | Ağustos | 130 |
| **9** | Eylül | 100 |
| **10** | Ekim | 140 |
| **11** | Kasım | 210 |
| **12** | Aralık | 290 |

* **Grafik Başlığı: "Cem'in Yıllık Elektrik Faturası Değişimi"**
* **Bu Grafik Ne Anlatıyor?**
* Çizgi grafik, verilerdeki **zaman içindeki değişimi**, **trendi** ve **dalgalanmaları** gösterir.
* **Hangi Soruları Cevaplar?**
* **"Hangi aylarda fatura en yüksek?"** → **Kış ayları** (Ocak, Şubat, Aralık)
* **"Hangi aylarda en düşük?"** → **Yaz başı** (Haziran)
* **"Genel bir düşüş veya yükseliş trendi var mı?"** → Kışın yükselip yazın düşen bir **mevsimsel dalgalanma** var
* **"Son bir yılda faturalarım nasıl değişmiş?"** → Grafiğin tamamı bu sorunun cevabı!
* **Ne Zaman Kullanılır?**
* **Zamana bağlı değişimleri** göstermek istediğinizde (aylık, yıllık)
* **Trend analizi** yapmak istediğinizde (artış/azalış/eğilim)
* **Karşılaştırmalı verileri** göstermek istediğinizde (örneğin 2 yılın aynı dönemlerini karşılaştırmak)
* **Özet: Hangi Grafiği Ne Zaman Kullanmalı?**

|  | **PASTA GRAFİK** | **ÇİZGİ GRAFİK** |
| --- | --- | --- |
| **Ana Amaç** | Parçaların bütün içindeki payı | Zaman içindeki değişim |
| **En İyi Kullanım** | "Bir şeyin dağılımı nasıl?" sorusuna cevap verir | "Zamanla nasıl değişiyor?" sorusuna cevap verir |
| **Örnek** | Harcamaların kategorilere dağılımı | Aylık fatura tutarlarının değişimi |
| **Eksen** | Genellikle tek eksenli | X (yatay) ve Y (dikey) olmak üzere iki eksenli |

**Öğrenci Tavsiyesi:** Kendi hayatınızdan verilerle (aylık harçlık, sınav notları, kitap okuma süreleri) bu grafikleri oluşturmayı deneyin. Excel'de verileri seçip "Ekle" sekmesinden grafik türünü seçmek yeterli! Pratik yaptıkça hangi grafiğin nerede kullanılacağı içgüdüsel hale gelecektir.

**Hangi Grafik Ne Zaman?**

* **"Dağılım" soruyorsan** → 🥧 **PASTA**
* **"Değişim" soruyorsan** → 📈 **ÇİZGİ**

**14. Pivot Tablo**

* **Bu Nedir?:** Büyük ve karmaşık veri listelerini özetlemek, analiz etmek ve anlamlı sonuçlar çıkarmak için kullanılan en güçlü Excel aracıdır.
* **Ne İşe Yarar?:** Yüzlerce satırlık bir satış listesinden, "Her şehrin toplam satışını" veya "Her ürünün aylık cirosunu" saniyeler içinde özet bir tabloya dönüştürebilirsiniz. Veri analizinin süper kahramanıdır.
* **Nasıl Yapılır?:** Veri listenizdeki herhangi bir hücreye tıklayın. "Ekle" sekmesinden "PivotTable"ı seçin. Yeni bir sayfa açılacaktır. Sağdaki bölmeden, hangi veriyi satır, sütun veya değer alanına sürüklemek istediğinizi seçin.
* **Özet Açıklama:** Büyük veri setlerini özetlemek, gruplamak ve analiz etmek için kullanılır.
* **Görsel Önerisi:** "Ekle > PivotTable" ve örnek özet tablo ekranı.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Tarih** | **Etkinlik** | **Katılımcı** | **Bölüm** | **Ücret(₺)** |
| 1 | 14.10.2025 22:49 | Kodlama | Ahmet | Mühendislik | 100 |
| 2 | 14.10.2025 19:10 | Kodlama | Ayşe | Mühendislik | 100 |
| 3 | 14.10.2025 17:26 | Resim | Mehmet | Güzel Sanatlar | 50 |
| 4 | 14.10.2025 01:28 | Müzik | Fatma | Müzik | 75 |
| 5 | 13.10.2025 23:30 | Müzik | Ali | Mühendislik | 75 |
| 6 | 13.10.2025 22:33 | Kodlama | Zeynep | Mühendislik | 100 |
| 7 | 13.10.2025 22:33 | Resim | Can | Güzel Sanatlar | 50 |
| 8 | 13.10.2025 21:08 | Müzik | Demet | Müzik | 75 |
| 9 | 13.10.2025 17:48 | Kodlama | Burak | Mühendislik | 100 |
| 10 | 13.10.2025 13:11 | Resim | Ceren | Güzel Sanatlar | 50 |

**PİVOT TABLE'IN GÜCÜ: 3 SANİYEDE 4 ANALİZ!  
Ham veride bulamayacağınız içgörüler:En popüler etkinlik:** Kodlama

1. **En fazla katılım:** Mühendislik bölümü
2. **Bölümler arası etkileşim:** YOK
3. **En karlı etkinlik:** Kodlama (400₺)

**ÖĞRENCİ ÇALIŞMA ÖDEVİ:**  
Kendi hayatından bir veri oluştur:

* 1 aylık yemek harcamaları
* 2 haftalık ders çalışma saatleri
* 1 aylık sosyal medya kullanımı

**Pratik yaptıkça, Pivot Table'ın ne kadar güçlü olduğunu göreceksin!**

**Unutma:** Pivot Table'da yanlış yapma korkusu yok! Sürükle-bırak, dene, değiştir...

### **Microsoft PowerPoint**

**1. Şablonlar (Templates)**

* **Bu Nedir?:** Önceden tasarlanmış, renk, yazı tipi ve düzen seçimleri yapılmış, üzerine kendi içeriğinizi ekleyebileceğiniz hazır sunum taslaklarıdır.
* **Ne İşe Yarar?:** Sıfırdan bir tasarım yapmakla vakit kaybetmezsiniz. Profesyonel ve tutarlı bir görünüm için en hızlı yoldur. Ödev sunumunuzu bir anda şık hale getirir.
* **Nasıl Yapılır?:** PowerPoint'i açtığınızda karşınıza çıkan ilk ekranda birçok şablon görürsünüz. "Dosya" > "Yeni" dediğinizde de çevrimiçi olarak yüzlerce şablonu görebilir, beğendiğinizi tıklayarak kullanmaya başlayabilirsiniz.

**2. Slayt (Saydam) Ekleme/Silme**

* **Bu Nedir?:** Sunumunuza yeni bir boş sayfa (slayt) eklemek veya var olan bir slaydı silmektir.
* **Ne İşe Yarar?:** Sunumunuzun yapısını oluşturmanın temelidir. Her yeni konu başlığı, görsel veya anafikir için yeni bir slayt ekleyebilirsiniz.
* **Nasıl Yapılır?:**
  + **Yeni Slayt Ekleme:** "Giriş" sekmesinde veya "Ekle" sekmesinde büyük "Yeni Slayt" butonunu göreceksiniz. Sadece boş bir slayt eklemek için Ctrl + M kısayolunu da kullanabilirsiniz.
  + **Slayt Silme:** Sol taraftaki slayt küçük resimlerinden silmek istediğinize tıklayıp Delete tuşuna basmanız yeterlidir.

**3. Nesnelerle Çalışmak**

* **Bu Nedir?:** Slaytlarınıza metin kutusu, resim, şekil, simge, grafik, video gibi her türlü öğeyi eklemek ve düzenlemektir.
* **Ne İşe Yarar?:** Görsel açıdan zengin ve anlaşılır bir sunum oluşturmanızı sağlar. Sadece yazıdan oluşan sıkıcı slaytlar yerine, görsellerle ve şemalarla desteklenmiş etkileyici bir sunum yaratırsınız.
* **Nasıl Yapılır?:** "Ekle" sekmesi nesnelerin ana merkezidir. Buradan istediğiniz nesneyi seçip slaydınıza ekleyebilirsiniz. Bir nesneye tıkladığınızda üstte çıkan "Biçim" sekmesinden renk, kenarlık, gölge gibi birçok efekti uygulayabilirsiniz.

**4. Geçiş Zamanlamasını Ayarlamak**

* **Bu Nedir?:** Slaytların birinden diğerine otomatik olarak, belirli bir süre sonunda geçmesini sağlamaktır.
* **Ne İşe Yarar?:** Sunumu siz anlatmıyorsanız (örneğin, bir fuar standında dönen bir sunum) veya slayt başına ayırdığınız süreyi kesin olarak kontrol etmek istiyorsanız (10 dakikalık bir sunum için 20 slaydınız varsa, her slayt 30 saniye gösterilsin gibi) kullanışlıdır.
* **Nasıl Yapılır?:** "Geçişler" sekmesinde, "Yenileme Süresi" (Advance Slide) kısmındaki "Sonra" (After) kutusunu işaretleyip yanına istediğiniz saniyeyi yazmanız yeterlidir (örn: 00:10:00).

**5. Geçiş Efektleri**

* **Bu Nedir?:** Bir slayttan diğerine geçerken kullanılan animasyon benzeri görsel efektlerdir. (Örn: Soldan kayma, kararma, yukarı açılma vb.)
* **Ne İşe Yarar?:** Sunumun akışına görsel bir hareket katar, izleyicinin dikkatini çeker ve slayt değişimlerini daha yumuşak hale getirir.
* **Nasıl Yapılır?:** "Geçişler" sekmesine gidin. Burada birçok hazır geçiş efektini göreceksiniz. Bir efekte tıkladığınızda o an seçili olan slayta uygulanır. Tüm slaytlara aynı efekti uygulamak için "Tümüne Uygula" butonunu kullanabilirsiniz.

**6. Alternatif Sunum Önerileri**

* **Bu Nedir?:** PowerPoint'in sunumunuzun tasarımı ve düzeni için otomatik olarak önerdiği değişikliklerdir. "Tasarım Fikirleri" (Designer) olarak da bilinir.
* **Ne İşe Yarar?:** Özellikle görsel ağırlıklı bir slayt hazırlarken (fotoğraf galerisi, başlık sayfası, grafik), PowerPoint sizin için birkaç farklı, profesyonel düzen seçeneği sunar. Bu, tasarım konusunda emin olamadığınızda veya hızlıca fikir almak istediğinizde müthiş bir yardımcıdır.
* **Nasıl Yapılır?:** Bir slayta bir resim eklediğinizde veya bir başlık yazdığınızda, sağ tarafta "Tasarım Fikirleri" paneline otomatik olarak öneriler gelir. Beğendiğiniz bir öneriye tıklayarak slaydınızı anında o şekle dönüştürebilirsiniz.

**Öğrenciler İçin Altın Değerinde PowerPoint İpuçları:**

1. **7x7 Kuralı:** Bir slaytta en fazla 7 satır, bir satırda en fazla 7 kelime olmalı. Slayt, konuşmanızın özeti olmalı, tam metni değil!
2. **Yazı Tipi Seçimi:** Okunabilir yazı tipleri kullanın (Arial, Calibri gibi). Dekoratif yazı tipleri göz yorar. Başlık için 36-44 punto, normal metin için 24-28 punto idealdir.
3. **Zıtlık Kuralı:** Açık renk bir arka plan üzerine koyu renk yazı VEYA koyu renk bir arka plan üzerine açık renk yazı kullanın. Asla benzer tonları üst üste kullanmayın (örneğin sarı üzerine beyaz).
4. **Görsel Kullanın:** Anlattığınız konuyu destekleyen yüksek kaliteli görseller kullanın. Bir slayda sıkıştırdığınız 3 paragrafı, tek bir iyi seçilmiş resim anlatabilir.
5. **Animasyonu Abartmayın:** Her kelimeye uçan, dönen efektler koymak profesyonel durmaz. Sade ve amaca hizmet eden animasyonlar kullanın.

**Unutmayın:** PowerPoint sadece bir araçtır. Asıl önemli olan, sizin anlatacaklarınızdır. Slaytlarınız sizi desteklesin, sizin yerinize konuşmasın!

**MODERN SUNUM ARAÇLARI**

**1. Canva**

* **Nedir?:** Bulut tabanlı, sürükle-bırak mantığıyla çalışan bir tasarım platformu.
* **Artıları (+):**
  + **Kullanım Kolaylığı:** PowerPoint'ten daha sezgisel ve hızlı
  + **Harika Şablonlar:** Binlerce profesyonel, modern şablon
  + **Zengin Medya Kütüphanesi:** Ücretsiz fotoğraf, ikon, video
  + **İşbirliği:** Aynı anda takım çalışması yapılabilir
  + **Animasyonlar:** Basit ama etkili animasyon efektleri
* **Eksileri (-):** İnternet gerektiriyor, bazı premium öğeler ücretli
* **Kim İçin?:** Hızlı, görsel açıdan etkileyici sunumlar hazırlamak isteyen herkes

**2. Prezi**

* **Nedir?:** Doğrusal olmayan, zoom in/out yapabilen dinamik sunum aracı
* **Artıları (+):**
  + **Akılda Kalıcı:** Geleneksel slaytlardan farklı, hafızalara kazınır
  + **Görsel Akış:** Büyük resmi ve detayları aynı sunumda gösterebilir
  + **Dinamik:** Statik slayt yerine harita gibi gezinebilirsiniz
* **Eksileri (-):** Başlangıçta öğrenmesi biraz zaman alabilir, bazı izleyicilerde baş dönmesi yapabilir
* **Kim İçin?:** Yaratıcı sunumlar yapmak isteyen, hikaye anlatımını sevenler

**3. Google Slides**

* **Nedir?:** PowerPoint'in bulut tabanlı, ücretsiz Google versiyonu
* **Artıları (+):**
  + **Ücretsiz ve Erişilebilir:** Google hesabı olan herkes kullanabilir
  + **Gerçek Zamanlı İşbirliği:** Aynı anda düzenleme, yorum yapma
  + **Otomatik Kaydetme:** Tüm değişiklikler bulutta anında kaydedilir
  + **Platform Bağımsız:** Tüm cihazlardan erişilebilir
* **Eksileri (-):** PowerPoint kadar gelişmiş özellikleri yok
* **Kim İçin?:** Grup projeleri, hızlı ve basit sunumlar, işbirliği gerektiren çalışmalar

**ETKİLEŞİMLİ SUNUM TEKNİKLERİ**

**4. Mentimeter**

* **Nedir?:** İzleyici katılımını sağlayan interaktif sunum aracı
* **Nasıl Çalışır?:**
  + Sunucu soruları hazırlar
  + İzleyiciler akıllı telefonlarıyla QR kodunu okutup katılır
  + Anket sonuçları, bulut kelime haritaları canlı olarak ekranda görünür
* **Kullanım Alanları:**
  + Beyin fırtınası seansları
  + Anlık geri bildirim toplama
  + Eğlenceli buz kırıcı aktiviteler

**5. PechaKucha (20x20)**

* **Nedir?:** Japon kökenli, zamanı sınırlı bir sunum formatı
* **Nasıl Çalışır?:**
  + 20 slayt × 20 saniye = Toplam 6 dakika 40 saniye
  + Slaytlar otomatik olarak ilerler
  + Konuşmacı zaman sınırına uymak zorunda
* **Faydaları:**
  + Sunumu odaklı ve öz tutar
  + Dinleyicinin dikkati dağılmaz
  + Çok sayıda konuşmacıya kısa sürede yer verilebilir

**YENİ NESİL SUNUM YAKLAŞIMLARI**

**6. Ignite Talks**

* **Format:** 5 dakika × 20 slayt (slaytlar 15 saniyede bir otomatik geçer)
* **Mantık:** Fikirlerin hızlı, enerjik ve öz şekilde paylaşılması
* **Örnek:** "Gelecekte Eğitim" konusunda 5 farklı kişinin 5'er dakikalık hızlı sunumları

**7. Video Temelli Sunumlar**

* **Araçlar:** Loom, Vimeo, hatta TikTok/Reels formatı
* **Nasıl Yapılır?:**
  + Ekran kaydı + web kamerası kaydı birlikte
  + Kısa, dinamik videolar (3-5 dakika)
  + Kişisel dokunuşlar içerir
* **Avantajı:** İzleyici istediği zaman izleyebilir, sunucuyla göz teması kurabilir.

**Hangisini Ne Zaman Kullanmalı?**

| **Amaç** | **En İyi Araç** | **Nedeni?** |
| --- | --- | --- |
| **Grup Ödevi** | Google Slides | Gerçek zamanlı işbirliği için mükemmel |
| **Yaratıcı Tasarım** | Canva | Profesyonel şablonlar ve kolay kullanım |
| **Akılda Kalıcı Sunum** | Prezi | Benzersiz zoom arayüzü ve akış |
| **İzleyici Katılımı** | Mentimeter | Anlık oylama, soru-cevap imkanı |
| **Hızlı Fikir Paylaşımı** | PechaKucha | Zorlayıcı zaman sınırı odak sağlar |
| **Uzaktan Eğitim** | Loom + Canva | Video anlatım + görsel slaytlar |

**ÖĞRENCİ PROJELERİ İÇİN İDEAL KULLANIM ÖRNEKLERİ**

**Tarih Projesi:**

* **Araç:** Prezi
* **Neden?:** Zaman çizelgesi üzerinde zoom in/out yaparak olayları görselleştirebilirsiniz

**Fen Deneyi:**

* **Araç:** Canva + Loom
* **Neden?:** Canva ile şık sunum, Loom ile deneyin videosunu kaydedip ekleyebilirsiniz

**Sosyoloji Araştırması:**

* **Araç:** Google Slides + Mentimeter
* **Neden?:** Grupça sunumu hazırlayıp, sunum sırasında izleyicilerden anketle veri toplayabilirsiniz

**Girişimcilik Fikri:**

* **Araç:** PechaKucha formatı
* **Neden?:** Fikrinizi kısa, öz ve etkili şekilde anlatmanız gerekir

**GELECEĞİN SUNUM TRENDLERİ**

1. **Yapay Zeka Destekli Sunumlar:** AI'nın sunum yapısı önermesi, görsel seçimi
2. **Sanal Gerçeklik Sunumları:** VR gözlüklerle 3D sunum deneyimleri
3. **Mikro Öğrenme:** 3-5 dakikalık mini sunum paketleri
4. **Kişiselleştirilmiş İçerik:** Aynı sunumun farklı izleyicilere farklı versiyonları

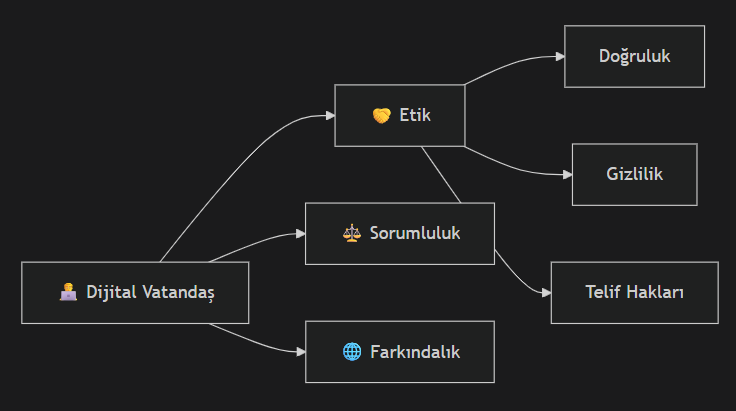
### **Dijital Okur Yazarlık**

**TEMEL BAŞLIKLAR ve ÖZET GÖRSELLER**

**1. Dijital Vatandaşlık ve Etik**

**İçerik:**

* Dijital ayak izi ve kalıcılığı
* Çevrimiçi nezaket (netiket)
* Fikri mülkiyet hakları
* Siber zorbalıkla mücadele



**2. Dijital Güvenlik ve Mahremiyet**

🔒 **GÜVENLİK DUVARI**

Şifre Yönetimi → 💪 Güçlü Parolalar

Kişisel Veriler → 🚫 Paylaşma

Gizlilik Ayarları → ⚙️ Kontrol

VPN & Güvenlik → 🛡️ Koruma

**İçerik:**

* Güçlü şifre oluşturma teknikleri
* İki faktörlü kimlik doğrulama
* Sosyal medya gizlilik ayarları
* Phishing (oltalama) saldırılarını tanıma

**3. Bilgi Okuryazarlığı**

**🧠 ELEŞTİREL DÜŞÜNCE SÜRECİ**

SOR → 🔍 Kaynak güvenilir mi?

ARA → 📊 Bilgiyi derinlemesine ara

DEĞERLENDİR → ⚖️ Taraflı/tarafsız analiz et

PAYLAŞ → ✅ Doğrulanmış bilgiyi yay

**İçerik:**

* Sahte haberleri tespit etme
* Bilgi kaynaklarını doğrulama
* Görsel manipülasyonu anlama
* Algorithm farkındalığı

**4. Dijital İletişim ve İşbirliği**

🌍 **DİJİTAL İLETİŞİM HARİTASI**

📧 E-posta → Profesyonel dil

💬 Anlık Mesajlaşma → Hızlı çözüm

📹 Video Konferans → Etkileşimli toplantı

👥 İşbirlikçi Araçlar → Takım çalışması

**İçerik:**

* Etkili e-posta yazma teknikleri
* Çevrimiçi toplantı kuralları
* Google Workspace/Azure araçları
* Dijital işbirliği protokolleri

**5. Dijital İçerik Üretimi**

🎨 **İÇERİK ÜRETİM MERKEZİ**

YAZI → ✍️ Blog & Makale

GÖRSEL → 🎨 İnfografik & Tasarım

VIDEO → 🎬 Kısa Film & Animasyon

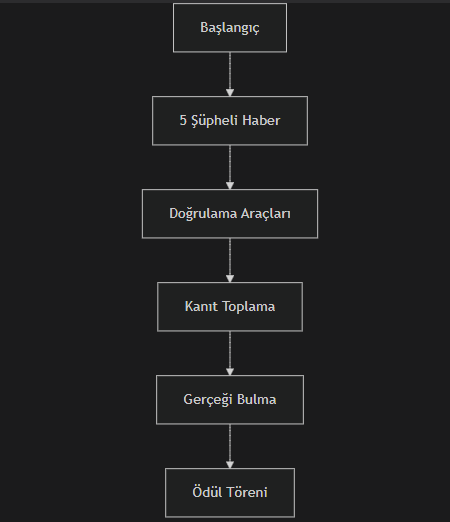
SES → 🎙️ Podcast & Ses Tasarımı

**İçerik:**

* Dijital hikaye anlatımı
* Temel grafik tasarım ilkeleri
* Video düzenleme teknikleri
* İçerik stratejisi oluşturma

**ETKİNLİK ÖNERİLERİ**

**1. "Siber Dedektif" Yarışması**



**Uygulama:**

* 5 sahte/gerçek haber içeren bir paket
* Öğrenciler doğrulama araçları kullanır
* En hızlı ve doğru tespit eden kazanır

**2. Dijital Ayak İzi Haritası**

**Aktivite:** Kendi dijital varlıklarını haritalandırma

🕵️ **KİŞİSEL ARAŞTIRMA FORMU**

─────────────────────────

Sosyal Medya → 3 platform

Arama Geçmişi → Son 24 saat

Uygulama İzinleri → 10 uygulama

Dijital Alışkanlıklar → Ekran süresi

**3. "Güvenli Şifre" Atölyesi**

**# Şifre Güç Testi Oyunu**

def sifre\_guclu\_mu(sifre):

uzunluk = len(sifre) >= 8

buyuk\_harf = any(c.isupper() for c in sifre)

kucuk\_harf = any(c.islower() for c in sifre)

rakam = any(c.isdigit() for c in sifre)

ozel\_karakter = any(not c.isalnum() for c in sifre)

return uzunluk and buyuk\_harf and kucuk\_harf and rakam and ozel\_karakter

**# Öğrenciler kendi şifrelerini test eder**

**4. Dijital Vatandaşlık Sözleşmesi**

**Etkinlik:** **Sınıf için dijital etik kuralları oluşturma**

📜 **DİJİTAL VATANDAŞLIK SÖZLEŞMESİ**

✓ Gerçek adımı kullanırım

✓ Başkalarının bilgilerini korurum

✓ Şüpheli bağlantıları tıklamam

✓ Dijital eserlerin telifine saygı duyarım

✓ Siber zorbalığa sessiz kalmam

**ÖRNEK PROJELER**

**1. Okul Dijital Güvenlik Rehberi**

**Proje Çıktısı: Okul için dijital güvenlik el kitabı**

📖 İÇİNDEKİLER

1. Güvenli Şifre Rehberi

2. Sosyal Medya Ayarları

3. Acil Durum Protokolleri

4. Güvenlik Kontrol Listesi

5. Yararlı Kaynaklar

**2. Dijital Okuryazarlık Oyunu**

**Oyun Tasarımı:**

* Seviye 1: Bilgi avcılığı (doğrulama becerileri)
* Seviye 2: Güvenlik kalesi (şifre ve gizlilik)
* Seviye 3: İletişim köprüsü (dijital iletişim)
* Seviye 4: İçerik üretici (yaratıcı beceriler)

**4. "Benim Dijital Kimliğim" Sunumları**

**Proje: Her öğrenci kendi dijital kimliğini anlatır**

🎤 SUNUM BAŞLIKLARI

• Dijital dünyada nasıl görünüyorum?

• Hangi becerileri geliştiriyorum?

• Gelecek dijital hedeflerim neler?

• Dijital topluma katkım ne olacak?

**DEĞERLENDİRME ARAÇLARI**

**1. Dijital Okuryazarlık Öz-Değerlendirme Anketi**

ÖLÇEK: 1-5 (Hiç Katılmıyorum - Tamamen Katılıyorum)

1. Bilgi kaynaklarını doğrulayabilirim [1][2][3][4][5]

2. Güçlü şifreler oluşturabilirim [1][2][3][4][5]

3. Dijital eserlerin telif haklarına saygı duyarım [1][2][3][4][5]

4. Çevrimiçi iletişim kurabilirim [1][2][3][4][5]

5. Dijital içerik üretebilirim [1][2][3][4][5]

**2. Portfolyo Değerlendirmesi**

Her öğrenci bir dijital portfolyo oluşturur:

* Doğrulanmış bir haber analizi
* Güvenlik kontrol listesi
* Dijital iletişim örneği
* Orijinal dijital içerik

**EĞİTİM ÇIKTILARI**

Bu programın sonunda öğrenciler:

* ✅ Dijital dünyada güvenle hareket edebilecek
* ✅ Bilgi kirliliği içinde doğruyu bulabilecek
* ✅ Etik değerleri içselleştirmiş olacak
* ✅ Yaratıcı dijital içerikler üretebilecek
* ✅ Dijital topluma aktif katkı sağlayabilecek

### **Temel Yazılım Rehberi**

### **Temel Yazılım Bilgisi**

**Eğitim Modülleri (Aşamalı ve Mantıklı Bir Sırayla)**

**MODÜL 1: Dijital Okuryazarlık ve Algoritmik Düşünce**

*Bu modül, yazılımın "neden" ve "nasıl" çalıştığını anlamaya yöneliktir.*

* **Bilgisayar Bilimine Giriş:** Donanım, yazılım, işletim sistemi nedir?
* **Algoritma ve Mantık Geliştirme:**
  + Günlük hayattan örnekler: "Çay nasıl demlenir?", "Okula nasıl gidilir?"
  + Akış şemaları (Flowcharts) çizerek problem çözme.
* **Pseudocode (Sözde Kod):** Doğal dil ile programlama dili arasında bir köprü. Problemi adım adım, kod yazmadan ifade etme.

**Günlük Hayattan Algoritma Örnekleri**

**1. ÇAY DEMLEME Algoritması**

**Amaç:** Sıralı işlemler ve koşul mantığını anlatmak

1. BAŞLA

2. Suyu çaydanlığa koy

3. Çaydanlığı ocağa yerleştir

4. Ocağı yak

5. Su kaynayana kadar BEKLE

6. EĞER su kaynadıysa:

6.1. Demliğe çayı koy

6.2. Kaynar suyu demliğe ekle

6.3. Demliğin kapağını kapat

6.4. 15 dakika BEKLE

7. Bardaklara çayı koy

8. İsteğe göre şeker ekle

9. BİTİR

**Algoritmik Kavramlar:**

* **Sıralı İşlemler** (1-2-3...)
* **Koşul İfadesi** (EĞER kaynadıysa)
* **Döngü** (kaynama kontrolü)
* **Parametre** (bekleme süresi)

**2. OKULA GİTME Algoritması**

**Amaç:** Karar yapılarını ve alternatif durumları anlatmak

1. BAŞLA

2. Saatine bak

3. EĞER derse geç kalıyorsan:

3.1. Servise bin

4. DEĞİLSE:

4.1. EĞER hava güzelse:

4.1.1. Yürüyerek git

4.2. DEĞİLSE:

4.2.1. Otobüse bin

5. Okula var

6. BİTİR

**Algoritmik Kavramlar:**

* **İç-içe koşullar** (nested if)
* **Mantıksal operatörler** (VE/VEYA)
* **Alternatif yollar**

**3. ALIŞVERİŞ YAPMA Algoritması**

**Amaç:** Döngü ve liste kavramlarını anlatmak

1. BAŞLA

2. Alışveriş listesini al

3. Markete git

4. Alışveriş arabasını al

5. Liste tamamlanana kadar TEKRARLA:

5.1. Listeden bir ürün seç

5.2. Rafa git

5.3. Ürünü bul

5.4. Arabaya koy

5.5. Listeden ürünü işaretle

6. Kasa sırasına gir

7. Ödeme yap

8. BİTİR

**Algoritmik Kavramlar:**

* **Döngü** (liste bitene kadar)
* **Liste işleme**
* **Sayaç** (toplam tutar)

**❓ Gençlere Yönelik Algoritma Soruları**

**A. Basit Seviye - "Senaryo Tamamlama"**

1. **"Telefonum çaldığında ne yaparsın? Adım adım anlat."**
   * Beklenen: Arama geldi → Ekrana bak → Karar ver (aç/reddet) → Konuş
2. **"En sevdiğin yemeği yapmak için gereken adımlar neler?"**
   * Malzeme listesi → Hazırlık → Pişirme → Servis

**B. Orta Seviye - "Problem Çözme"**

1. **"Misafirin kapıda. İçeri aldın. Şimdi ne yaparsın? 5 adım sırala."**
   * Beklenen: Selamla → Ayakkabılığa yönlendir → Oturmasını sağla → İkram teklif et → Sohbete başla
2. **"Dondurma alacaksın ama 3 farklı dükkân var. Hangisini seçersin? Kararını nasıl verirsin?"**
   * Fiyat → Kalite → Mesafe → Lezzet faktörlerini değerlendirme

**C. İleri Seviye - "Optimizasyon"**

1. **"5 arkadaş sinemaya gideceksiniz. Herkesin farklı istekleri var. Nasıl ortak karar alırsınız?"**
   * Oylama → Uzlaşma → Alternatif bulma algoritması
2. **"Sabah okula yetişmek için en hızlı hazırlanma planını nasıl yaparsın? Hangi işleri aynı anda yapabilirsin?"**
   * Paralel işlemler (kahvaltı+kitap hazırlama) → Öncelik sıralaması

**🎮 Etkileşimli Grup Aktiviteleri**

**"İnsan Robot" Oyunu:**

* **Amaç:** Bir kişiyi "robot" yaparak komutlarla yönlendirme
* **Örnek:** "Kalemi al ve çöp kutusuna at"
* **Robot sadece verilen komutları uygular:**
  + Sağ elini kaldır
  + 3 adım ileri git
  + Eğil
  + Kalemi al
  + vb.

**"Algoritma Labirenti":**

* **Amaç:** Kâğıt üzerinde labirentten çıkış algoritması yazma
* **Kurallar:** Sadece "ileri", "sağa dön", "sola dön", "engel var mı?" komutları kullanılabilir

**💡 Öğretim İpuçları:**

1. **Somutlaştırın:** Fiziksel nesnelerle (lego, kağıt) örnekler yapın
2. **Hata Yapmaya Teşvik Edin:** Yanlış algoritmaları birlikte düzeltin
3. **Eğlenceli Hale Getirin:** Yarışma formatında problemler çözün
4. **Günlük Hayatla Bağ Kurun:** "Bugün hangi algoritmaları kullandın?" sorusunu sık sık sorun

**Unutmayın:** Amacımız gençlerin "Ben zaten bunu yapıyormuşum!" demesini sağlamak. Algoritmik düşünce aslında hepimizin doğal olarak kullandığı bir beceridir - sadece adlandırmayı ve sistematik hale getirmeyi öğreniyoruz!

**MODÜL 2: Programlamaya İlk Adım (Blok Tabanlı- Görsel Programlama)**

*Kod yazma korkusunu yenmek ve temel programlama mantığını eğlenceli bir şekilde öğretmek için idealdir.*

* **Araçlar:** Scratch, Microsoft MakeCode veya App Inventor.
* **Kazanımlar:**
  + Döngüler (tekrarlanan işlemler)
  + Koşul ifadeleri (eğer...ise...)
  + Değişken kavramı
* **Proje:** Basit bir labirent oyunu, kendi karikatürünü oluşturma veya interaktif bir hikâye yazma.

**MODÜL 3: Web Geliştirmeye Giriş (İnternet Nasıl Çalışır?)**

*Gençlerin en çok etkileşimde olduğu alan olan web'i anlamalarını sağlar.*

* **Temel Kavramlar:** İnternet, tarayıcı, sunucu, istemci (client) nedir?
* **Front-end (Ön Yüz) Geliştirme:**
  + **HTML:** Web sayfasının iskeleti. Başlık, paragraf, liste, resim, link ekleme.
  + **CSS:** Web sayfasının stil ve tasarımı. Renkler, fontlar, layout (düzen) oluşturma.
* **Proje:** Kendi kişisel tanıtım sayfasını veya sevdiği bir konuda basit bir blog sayfasını HTML ve CSS ile oluşturma.

**MODÜL 4: Metin Tabanlı Programlamaya Geçiş**

*Artık gerçek kod yazma zamanı! Burada sade bir dil seçmek önemlidir.*

* **Dil Seçimi:** **Python** tavsiye edilir. Sözdizimi (syntax) basit, okunması kolay ve çok güçlüdür.
* **Temel Kavramlar:**
  + Değişkenler ve Veri Tipleri (String, Integer, Boolean)
  + Temel Operatörler
  + Kullanıcı Etkileşimi (input() ve print())
  + Karar Yapıları (if, elif, else)
  + Döngüler (for, while)
* **Proje:** Basit bir hesap makinesi, sayı tahmin oyunu veya "Asal sayı kontrolü" yapan bir program.

**MODÜL 5: Versiyon Kontrol Sistemleri (Takım Çalışması ve Proje Yönetimi)**

*Profesyonel yazılım dünyasının vazgeçilmez aracı.*

* **Git ve GitHub'a Giriş:**
  + Neden ihtiyaç var? ("Geri al" butonu gibi düşünün.)
  + Temel Komutlar: commit, push, pull.
  + GitHub'da bir depo (repository) oluşturma ve projelerini yükleyerek "portföy" oluşturma.

**3. Öğretim Yöntemleri ve Aktiviteler**

* **Proje Tabanlı Öğrenme:** Her modül sonunda mutlaka somut bir proje yapılmalı.
* **Oyunlaştırma:** Kodlama yarışmaları, hackathon'lar, rozet sistemleri.
* **Akran Öğrenmesi:** Gençler birbirlerine öğreterek daha iyi kavrar.
* **Konuk Konuşmacılar:** Sektörden yazılımcılar, girişimciler gençlerle buluşturulabilir.
* **Hata Yapma Kültürü:** "Hata yapmaktan korkmayın, hatalar en iyi öğretmendir!" anlayışı hâkim olmalı.

**4. Kullanılabilecek Ücretsiz Kaynaklar ve Araçlar**

* **Scratch:** [scratch.mit.edu](https://scratch.mit.edu/)
* **Replit / CodePen:** Tarayıcıda kod yazmak için harika çevrimiçi editörler.
* **W3Schools / freeCodeCamp (Türkçe):** Mükemmel, ücretsiz ve Türkçe dokümantasyon.
* **Python Resmi Dokümantasyonu (Türkçe çevirileri mevcut)**
* **GitHub Education Pack:** Öğrenciler ve eğitimciler için ücretsiz premium araçlar.

**5. Örnek Eğitim Akışı (12 Haftalık)**

* **Hafta 1-2:** Algoritma, Akış Şemaları, Scratch ile Programlama.
* **Hafta 3-5:** HTML & CSS - Kişisel Web Sayfası Yapımı.
* **Hafta 6-9:** Python Temelleri - Değişkenler, Koşullar, Döngüler.
* **Hafta 10:** Python ile Mini Proje Geliştirme.
* **Hafta 11:** Git & GitHub'a Giriş, Projeleri GitHub'a Yükleme.
* **Hafta 12:** "Demo Günü" - Gençlerin projelerini sunması ve sertifika töreni.

**Önemli Tavsiyeler:**

* **Baskı Yok, Keşif Var:** Müfredat esnek olmalı, gençlerin kendi ilgi alanlarına yönelmesine izin verilmeli.
* **Sorun Çözme Becerisi:** Cevabı direkt vermek yerine, cevabı kendilerinin bulması için rehberlik edin.
* **Güncel Kalın:** Teknoloji hızla değişiyor. Popüler ve gelecek vaat eden teknolojiler (yapay zekâ, veri bilimi gibi) hakkında kısa merak uyandırıcı konuşmalar yapılabilir.

### **Temel SQL Bilgisi**

**SQL'e Giriş: "Veri Dünyasının Anahtarı"**

**Neden SQL? - Kütüphane Benzetmesi**

* **Veritabanı** = Kütüphane (Database)
* **Tablo** = Raf (Table)
* **Kayıt** = Kitap (Row)
* **SQL** = Kütüphaneciye sorular

**MySQL CRUD İşlemleri (Temel Operasyonlar)**

Önce örnek bir tablo oluşturalım:

*-- Öğrenci tablosu oluşturma*

CREATE TABLE ogrenciler (

id INT PRIMARY KEY AUTO\_INCREMENT,

adi VARCHAR(50) NOT NULL,

soyadi VARCHAR(50) NOT NULL,

yas INT,

not\_ortalamasi DECIMAL(3,2),

kayit\_tarihi DATE DEFAULT CURDATE()

);

*-- Sınıf tablosu*

CREATE TABLE siniflar (

sinif\_id INT PRIMARY KEY,

sinif\_adi VARCHAR(20),

kapasite INT

);

**1. CREATE (Oluşturma) - INSERT**

**A) Tek Kayıt Ekleme**

INSERT INTO ogrenciler (adi, soyadi, yas, not\_ortalamasi)

VALUES ('Ahmet', 'Yılmaz', 15, 85.50);

**B) Çoklu Kayıt Ekleme**

INSERT INTO ogrenciler (adi, soyadi, yas, not\_ortalamasi) VALUES

('Ayşe', 'Kaya', 16, 92.00),

('Mehmet', 'Demir', 15, 78.50),

('Zeynep', 'Şahin', 17, 88.75),

('Can', 'Öztürk', 16, 65.25);

**C) Tablodan Veriyle Ekleme**

*-- Başarılı öğrencileri yeni tabloya kopyalama*

INSERT INTO basarili\_ogrenciler

SELECT \* FROM ogrenciler WHERE not\_ortalamasi > 80;

**2. READ (Okuma) - SELECT**

**A) Temel Sorgular**

*-- Tüm öğrencileri listele*

SELECT \* FROM ogrenciler;

*-- Sadece ad ve not bilgileri*

SELECT adi, soyadi, not\_ortalamasi FROM ogrenciler;

*-- Takma isim (alias) kullanımı*

SELECT

adi AS 'İsim',

soyadi AS 'Soyisim',

not\_ortalamasi AS 'Ortalama'

FROM ogrenciler;

**B) WHERE ile Filtreleme**

*-- Not ortalaması 80'den büyük olanlar*

SELECT \* FROM ogrenciler WHERE not\_ortalamasi > 80;

*-- Yaşı 16 olan öğrenciler*

SELECT \* FROM ogrenciler WHERE yas = 16;

*-- İsmi 'A' ile başlayanlar*

SELECT \* FROM ogrenciler WHERE adi LIKE 'A%';

*-- Belirli isimlerde olanlar*

SELECT \* FROM ogrenciler WHERE adi IN ('Ahmet', 'Ayşe');

*-- Yaş aralığı sorgusu*

SELECT \* FROM ogrenciler WHERE yas BETWEEN 15 AND 16;

**C) Sıralama ve Gruplama**

*-- Not ortalamasına göre yüksekten düşüğe*

SELECT \* FROM ogrenciler

ORDER BY not\_ortalamasi DESC;

*-- Yaşa göre grupla ve say*

SELECT yas, COUNT(\*) AS 'Öğrenci Sayısı'

FROM ogrenciler

GROUP BY yas;

*-- Ortalama notu 70'ten fazla olan yaş grupları*

SELECT yas, AVG(not\_ortalamasi) AS ortalama

FROM ogrenciler

GROUP BY yas

HAVING AVG(not\_ortalamasi) > 70;

**3. UPDATE (Güncelleme)**

**A) Tek Kayıt Güncelleme***-- Ahmet'in notunu güncelle*

UPDATE ogrenciler

SET not\_ortalamasi = 90.00

WHERE adi = 'Ahmet' AND soyadi = 'Yılmaz';

**B) Toplu Güncelleme**

*-- Tüm 15 yaşındaki örencilere +5 puan bonus*

UPDATE ogrenciler

SET not\_ortalamasi = not\_ortalamasi + 5

WHERE yas = 15;

*-- Birden fazla alan güncelleme*

UPDATE ogrenciler

SET not\_ortalamasi = 85.00, yas = 16

WHERE id = 3;

**4. DELETE (Silme)**

**A) Belirli Kayıtları Silme**

*-- Not ortalaması 50'den düşük olanları sil*

DELETE FROM ogrenciler

WHERE not\_ortalamasi < 50;

**B) Tüm Kayıtları Temizleme**

*-- Tüm kayıtları sil (dikkatli kullanın!)*

DELETE FROM ogrenciler;

*-- Tabloyu tamamen resetleme*

TRUNCATE TABLE ogrenciler;

**FONKSİYONLAR (MySQL Functions)**

**1. Skaler Fonksiyonlar**

**A) Matematiksel Fonksiyonlar**

*-- Notu yuvarlama*

SELECT adi, not\_ortalamasi, ROUND(not\_ortalamasi) AS yuvarlanmis\_not

FROM ogrenciler;

*-- Rastgele öğrenci seçme*

SELECT \* FROM ogrenciler

ORDER BY RAND()

LIMIT 1;

**B) String Fonksiyonlar**

*-- İsimleri birleştirme*

SELECT CONCAT(adi, ' ', soyadi) AS tam\_adi,

UPPER(adi) AS buyuk\_harf,

LENGTH(adi) AS isim\_uzunlugu

FROM ogrenciler;

*-- İsim formatlama*

SELECT CONCAT(UPPER(LEFT(adi,1)), LOWER(SUBSTRING(adi,2))) AS duzenli\_isim

FROM ogrenciler;

**C) Tarih Fonksiyonları**

*-- Kayıt tarihi işlemleri*

SELECT

adi,

kayit\_tarihi,

YEAR(kayit\_tarihi) AS kayit\_yili,

MONTHNAME(kayit\_tarihi) AS kayit\_ayi,

DATEDIFF(CURDATE(), kayit\_tarihi) AS gun\_sayisi

FROM ogrenciler;

**2. Kullanıcı Tanımlı Fonksiyonlar**

**A) Not Durumu Belirleme Fonksiyonu**

DELIMITER //

CREATE FUNCTION not\_durumu(p\_not DECIMAL(3,2))

RETURNS VARCHAR(20)

READS SQL DATA

DETERMINISTIC

BEGIN

DECLARE durum VARCHAR(20);

IF p\_not >= 90 THEN SET durum = 'Mükemmel';

ELSEIF p\_not >= 80 THEN SET durum = 'Çok İyi';

ELSEIF p\_not >= 70 THEN SET durum = 'İyi';

ELSEIF p\_not >= 60 THEN SET durum = 'Orta';

ELSE SET durum = 'Kötü';

END IF;

RETURN durum;

END//

DELIMITER ;

**Kullanımı:**

SELECT

adi,

not\_ortalamasi,

not\_durumu(not\_ortalamasi) AS durum

FROM ogrenciler;

**B) Yaş Grubu Belirleme Fonksiyonu**

DELIMITER //

CREATE FUNCTION yas\_grubu(p\_yas INT)

RETURNS VARCHAR(15)

DETERMINISTIC

BEGIN

RETURN CASE

WHEN p\_yas < 15 THEN 'Çocuk'

WHEN p\_yas BETWEEN 15 AND 17 THEN 'Genç'

ELSE 'Yetişkin'

END;

END//

DELIMITER ;

**Kullanımı:**

SELECT

adi,

yas,

yas\_grubu(yas) AS yas\_grubu

FROM ogrenciler;

**PROCEDURE'ler (Saklı Yordamlar)**

**1. Öğrenci İstatistikleri Prosedürü**

DELIMITER //

CREATE PROCEDURE ogrenci\_istatistikleri()

BEGIN

-- Toplam öğrenci sayısı

SELECT COUNT(\*) AS toplam\_ogrenci FROM ogrenciler;

-- Not ortalaması

SELECT AVG(not\_ortalamasi) AS genel\_ortalama FROM ogrenciler;

-- Yaş dağılımı

SELECT yas, COUNT(\*) AS sayi

FROM ogrenciler

GROUP BY yas

ORDER BY yas;

-- Başarı durumu

SELECT

COUNT(CASE WHEN not\_ortalamasi >= 70 THEN 1 END) AS basarili,

COUNT(CASE WHEN not\_ortalamasi < 70 THEN 1 END) AS basarisiz

FROM ogrenciler;

END//

DELIMITER ;

**Kullanımı:**

CALL ogrenci\_istatistikleri();

**2. Parametreli Öğrenci Ekleme Prosedürü**

DELIMITER //

CREATE PROCEDURE ogrenci\_ekle(

IN p\_adi VARCHAR(50),

IN p\_soyadi VARCHAR(50),

IN p\_yas INT,

IN p\_not DECIMAL(3,2)

)

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

SELECT 'Hata oluştu! Öğrenci eklenemedi.' AS mesaj;

END;

START TRANSACTION;

INSERT INTO ogrenciler (adi, soyadi, yas, not\_ortalamasi)

VALUES (p\_adi, p\_soyadi, p\_yas, p\_not);

COMMIT;

SELECT 'Öğrenci başarıyla eklendi!' AS mesaj;

END//

DELIMITER ;

**Kullanımı:**

CALL ogrenci\_ekle('Deniz', 'Kara', 16, 88.50);

**3. Not Güncelleme Prosedürü**

DELIMITER //

CREATE PROCEDURE not\_guncelle(

IN p\_ogrenci\_id INT,

IN p\_yeni\_not DECIMAL(3,2)

)

BEGIN

DECLARE eski\_not DECIMAL(3,2);

-- Eski notu al

SELECT not\_ortalamasi INTO eski\_not

FROM ogrenciler

WHERE id = p\_ogrenci\_id;

-- Güncelleme yap

UPDATE ogrenciler

SET not\_ortalamasi = p\_yeni\_not

WHERE id = p\_ogrenci\_id;

-- Değişiklik bilgisi

SELECT

CONCAT('Not ', eski\_not, ' → ', p\_yeni\_not, ' olarak güncellendi!') AS bilgi;

END//

DELIMITER ;

**Kullanımı:**

CALL not\_guncelle(1, 95.00);

**TRIGGER'lar (Tetikleyiciler)**

**1. Audit Trigger - Değişiklikleri Takip**

*-- Değişiklik log tablosu*

CREATE TABLE ogrenci\_log (

log\_id INT PRIMARY KEY AUTO\_INCREMENT,

ogrenci\_id INT,

islem VARCHAR(10),

eski\_not DECIMAL(3,2),

yeni\_not DECIMAL(3,2),

degisiklik\_tarihi TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

kullanici VARCHAR(50)

);

**A) UPDATE Trigger'ı**

DELIMITER //

CREATE TRIGGER ogrenci\_not\_degisimi

BEFORE UPDATE ON ogrenciler

FOR EACH ROW

BEGIN

IF OLD.not\_ortalamasi != NEW.not\_ortalamasi THEN

INSERT INTO ogrenci\_log (ogrenci\_id, islem, eski\_not, yeni\_not, kullanici)

VALUES (OLD.id, 'UPDATE', OLD.not\_ortalamasi, NEW.not\_ortalamasi, USER());

END IF;

END//

DELIMITER ;

**B) INSERT Trigger'ı**

DELIMITER //

CREATE TRIGGER yeni\_ogrenci\_kaydi

AFTER INSERT ON ogrenciler

FOR EACH ROW

BEGIN

INSERT INTO ogrenci\_log (ogrenci\_id, islem, yeni\_not, kullanici)

VALUES (NEW.id, 'INSERT', NEW.not\_ortalamasi, USER());

END//

DELIMITER ;

**C) DELETE Trigger'ı**

DELIMITER //

CREATE TRIGGER ogrenci\_silme

BEFORE DELETE ON ogrenciler

FOR EACH ROW

BEGIN

INSERT INTO ogrenci\_log (ogrenci\_id, islem, eski\_not, kullanici)

VALUES (OLD.id, 'DELETE', OLD.not\_ortalamasi, USER());

END//

DELIMITER ;

**2. Doğrulama Trigger'ı - Not Kontrolü**

DELIMITER //

CREATE TRIGGER not\_kontrolu

BEFORE INSERT ON ogrenciler

FOR EACH ROW

BEGIN

IF NEW.not\_ortalamasi < 0 OR NEW.not\_ortalamasi > 100 THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Not 0-100 arasında olmalıdır!';

END IF;

IF NEW.yas < 10 OR NEW.yas > 25 THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Yaş 10-25 arasında olmalıdır!';

END IF;

END//

DELIMITER ;

-- Log kayıtlarını görüntüle

SELECT \* FROM ogrenci\_log ORDER BY degisiklik\_tarihi DESC;

-- Son 1 gündeki değişiklikler

SELECT \* FROM ogrenci\_log

WHERE degisiklik\_tarihi >= DATE\_SUB(NOW(), INTERVAL 1 DAY);

**Interaktif Öğrenme:**

* **"SQL Yarışması"** - En hızlı sorguyu yazma
* **"Hata Avı"** - Yanlış sorguları düzeltme
* **"Gerçek Veri Setleri"** - Film, spor, müzik verileri

**Güvenlik Uyarıları:**

* **DELETE/UPDATE** öncesi mutlaka **SELECT** ile kontrol
* **WHERE** koşulu olmadan çalıştırmama
* **BACKUP** almayı unutmama

**Unutmayın:** "SQL öğrenmek süper güç kazanmaktır - verilerle konuşabilirsiniz!"

**SQL YARIŞMASI: VERİ KAŞİFLERİ MÜCADELESİ**

**📊 YARIŞMA ÖNCESİ HAZIRLIK**

**Veritabanı Kurulumu:**

sql

CREATE DATABASE sql\_yarismasi;

USE sql\_yarismasi;

-- Filmler tablosu

CREATE TABLE filmler (

film\_id INT PRIMARY KEY AUTO\_INCREMENT,

film\_adi VARCHAR(100) NOT NULL,

yonetmen VARCHAR(50),

yil INT,

tur VARCHAR(30),

imdb\_puanı DECIMAL(2,1),

sure\_dakika INT,

hasılat BIGINT

);

-- Oyuncular tablosu

CREATE TABLE oyuncular (

oyuncu\_id INT PRIMARY KEY AUTO\_INCREMENT,

oyuncu\_adi VARCHAR(50),

dogum\_yili INT,

ulke VARCHAR(30)

);

-- Film\_Oyuncu ilişki tablosu

CREATE TABLE film\_oyuncu (

film\_id INT,

oyuncu\_id INT,

rol VARCHAR(50),

PRIMARY KEY (film\_id, oyuncu\_id)

);

-- Örnek verileri ekleme (yarışma başlangıcında yapılacak)

**🎯 BAŞLANGIÇ SEVİYESİ SORULAR (10 Puan each)**

**Soru 1: Temel SELECT**

sql

-- Tüm filmleri listeleyen sorguyu yazınız.

-- Beklenen: film\_adi, yonetmen, yil sütunları gelsin

**Cevap:**

sql

SELECT film\_adi, yonetmen, yil FROM filmler;

**Soru 2: WHERE Filtreleme**

sql

-- 2000 yılından sonra çekilmiş filmleri bulunuz.

**Cevap:**

sql

SELECT \* FROM filmler WHERE yil > 2000;

**Soru 3: Sıralama**

sql

-- IMDB puanına göre en yüksek puanlı 5 filmi listeleyiniz.

**Cevap:**

sql

SELECT film\_adi, imdb\_puanı

FROM filmler

ORDER BY imdb\_puanı DESC

LIMIT 5;

**Soru 4: LIKE Operatörü**

sql

-- İsmi "A" harfi ile başlayan oyuncuları bulunuz.

**Cevap:**

sql

SELECT \* FROM oyuncular WHERE oyuncu\_adi LIKE 'A%';

**Soru 5: COUNT ve GROUP BY**

sql

-- Her yönetmenin kaç film çektiğini bulunuz.

**Cevap:**

sql

SELECT yonetmen, COUNT(\*) as film\_sayisi

FROM filmler

GROUP BY yonetmen;

**⚡ ORTA SEVİYE SORULAR (20 Puan each)**

**Soru 6: JOIN İşlemi**

sql

-- Her filmin oyuncularını listeleyiniz.

-- Beklenen: film\_adi, oyuncu\_adi, rol

**Cevap:**

sql

SELECT f.film\_adi, o.oyuncu\_adi, fo.rol

FROM filmler f

JOIN film\_oyuncu fo ON f.film\_id = fo.film\_id

JOIN oyuncular o ON fo.oyuncu\_id = o.oyuncu\_id;

**Soru 7: HAVING Kullanımı**

sql

-- Ortalama IMDB puanı 7.5'ten fazla olan türleri bulunuz.

**Cevap:**

sql

SELECT tur, AVG(imdb\_puanı) as ortalama\_puan

FROM filmler

GROUP BY tur

HAVING AVG(imdb\_puanı) > 7.5;

**Soru 8: SUBQUERY**

sql

-- En yüksek hasılat yapan filmden daha düşük hasılatı olan filmleri bulunuz.

**Cevap:**

sql

SELECT film\_adi, hasılat

FROM filmler

WHERE hasılat < (SELECT MAX(hasılat) FROM filmler);

**Soru 9: CASE WHEN**

sql

-- Filmleri puanlarına göre sınıflandırınız:

-- 8.0+ → "Mükemmel", 7.0-7.9 → "İyi", 6.0-6.9 → "Orta", diğer → "Zayıf"

**Cevap:**

sql

SELECT film\_adi, imdb\_puanı,

CASE

WHEN imdb\_puanı >= 8.0 THEN 'Mükemmel'

WHEN imdb\_puanı >= 7.0 THEN 'İyi'

WHEN imdb\_puanı >= 6.0 THEN 'Orta'

ELSE 'Zayıf'

END AS degerlendirme

FROM filmler;

**Soru 10: Tarih Fonksiyonları**

sql

-- Bugünün tarihini ve 30 gün sonrasını gösteren sorguyu yazınız.

**Cevap:**

sql

SELECT CURDATE() as bugun,

DATE\_ADD(CURDATE(), INTERVAL 30 DAY) as otuz\_gun\_sonra;

**🚀 İLERİ SEVİYE SORULAR (30 Puan each)**

**Soru 11: Window Fonksiyonu**

sql

-- Her türde en yüksek puanlı filmi bulunuz.

**Cevap:**

sql

SELECT tur, film\_adi, imdb\_puanı

FROM (

SELECT tur, film\_adi, imdb\_puanı,

ROW\_NUMBER() OVER (PARTITION BY tur ORDER BY imdb\_puanı DESC) as sıra

FROM filmler

) ranked

WHERE sıra = 1;

**Soru 12: Recursive CTE**

sql

-- 1'den 10'a kadar sayıları üreten sorguyu yazınız.

**Cevap:**

sql

WITH RECURSIVE sayilar AS (

SELECT 1 as sayı

UNION ALL

SELECT sayı + 1 FROM sayilar WHERE sayı < 10

)

SELECT \* FROM sayilar;

**Soru 13: Pivot Tablo**

sql

-- Her yıl için film sayısını gösteren pivot tablo oluşturun.

**Cevap:**

sql

SELECT

COUNT(CASE WHEN yil = 2020 THEN 1 END) as "2020",

COUNT(CASE WHEN yil = 2021 THEN 1 END) as "2021",

COUNT(CASE WHEN yil = 2022 THEN 1 END) as "2022",

COUNT(CASE WHEN yil = 2023 THEN 1 END) as "2023"

FROM filmler;

**Soru 14: Complex JOIN**

sql

-- Aynı filmde oynayan oyuncu çiftlerini bulunuz.

**Cevap:**

sql

SELECT DISTINCT

f.film\_adi,

o1.oyuncu\_adi as oyuncu1,

o2.oyuncu\_adi as oyuncu2

FROM film\_oyuncu fo1

JOIN film\_oyuncu fo2 ON fo1.film\_id = fo2.film\_id AND fo1.oyuncu\_id < fo2.oyuncu\_id

JOIN filmler f ON fo1.film\_id = f.film\_id

JOIN oyuncular o1 ON fo1.oyuncu\_id = o1.oyuncu\_id

JOIN oyuncular o2 ON fo2.oyuncu\_id = o2.oyuncu\_id;

**Soru 15: Performance Optimization**

sql

-- En çok oyuncusu olan 3 filmi bulunuz (performanslı şekilde).

**Cevap:**

sql

SELECT f.film\_adi, COUNT(fo.oyuncu\_id) as oyuncu\_sayisi

FROM filmler f

JOIN film\_oyuncu fo ON f.film\_id = fo.film\_id

GROUP BY f.film\_id, f.film\_adi

ORDER BY oyuncu\_sayisi DESC

LIMIT 3;

**🎪 YARATICI BONUS SORULAR (50 Puan each)**

**Bonus 1: Hikaye Tamamlama**

sql

/\*

SENARYO: Bir film prodüksiyon şirketisiniz.

2024'te çekeceğiniz yeni film için:

- 7.5+ IMDB puanı olan yönetmenler

- Bu yönetmenlerle çalışmış oyuncular

- Bu oyuncuların yer aldığı başarılı filmler (7.0+)

listesini getirin.

\*/

**Cevap:**

sql

SELECT DISTINCT

f.yonetmen,

o.oyuncu\_adi,

f2.film\_adi as basarili\_film,

f2.imdb\_puanı

FROM filmler f

JOIN film\_oyuncu fo ON f.film\_id = fo.film\_id

JOIN oyuncular o ON fo.oyuncu\_id = o.oyuncu\_id

JOIN film\_oyuncu fo2 ON o.oyuncu\_id = fo2.oyuncu\_id

JOIN filmler f2 ON fo2.film\_id = f2.film\_id

WHERE f.imdb\_puanı > 7.5

AND f2.imdb\_puanı > 7.0

ORDER BY f2.imdb\_puanı DESC;

**Bonus 2: Veri Analizi**

sql

/\*

Görev: Film endüstrisi analizi yapın:

- Her türün ortalama puanı ve hasılatı

- En başarılı 3 tür (puan + hasılat)

- Bu türlerdeki en iyi yönetmenler

\*/

**Cevap:**

sql

WITH tur\_istatistikleri AS (

SELECT

tur,

AVG(imdb\_puanı) as ortalama\_puan,

AVG(hasılat) as ortalama\_hasılat,

(AVG(imdb\_puanı) \* 0.6 + AVG(hasılat) / 1000000 \* 0.4) as basari\_puani

FROM filmler

GROUP BY tur

),

en\_iyi\_turler AS (

SELECT tur

FROM tur\_istatistikleri

ORDER BY basari\_puani DESC

LIMIT 3

),

yonetmen\_basarilari AS (

SELECT

f.tur,

f.yonetmen,

AVG(f.imdb\_puanı) as yonetmen\_ortalamasi,

COUNT(\*) as film\_sayisi

FROM filmler f

JOIN en\_iyi\_turler et ON f.tur = et.tur

GROUP BY f.tur, f.yonetmen

HAVING COUNT(\*) >= 2

)

SELECT \* FROM yonetmen\_basarilari ORDER BY tur, yonetmen\_ortalamasi DESC;

**⏱️ HIZ TESTİ SORULARI (Süre: 5 Dakika)**

**Hız Testi 1:**

sql

-- 5 tabloyu JOIN ederek kompleks rapor oluşturma

**Hız Testi 2:**

sql

-- 1,000,000 kayıt üzerinde optimize sorgu yazma

**Hız Testi 3:**

sql

-- Veritabanı tasarımı ve index önerileri

**🏅 DEĞERLENDİRME KRİTERLERİ**

**Puanlama Sistemi:**

* **Doğruluk:** %40
* **Performans:** %25
* **Kod Temizliği:** %20
* **Yaratıcılık:** %15

**Zorluk Katsayıları:**

* ✅ Başlangıç: 1.0x
* ⚡ Orta: 1.5x
* 🚀 İleri: 2.0x
* 🎪 Bonus: 3.0x

**Ödüller:**

* **Altın SQL Kahramanı:** 90+ puan
* **Gümüş Veri Ustası:** 75-89 puan
* **Bronz Kod Şampiyonu:** 60-74 puan

**💡 YARIŞMA İPUÇLARI**

1. **Önce SELECT ile test edin**
2. **WHERE koşullarını basitleştirin**
3. **INDEX kullanımını düşünün**
4. **Subquery yerine JOIN deneyin**
5. **Süreyi iyi kullanın**

**Gençlere Motivasyon:** "Unutmayın, her SQL kahramanı bir zamanlar WHERE yazmayı öğreniyordu! Hatalardan korkmayın, onlar sizi daha güçlü yapar! 🚀"

**Farklı kategorilerde hatalı SQL örnekleri:**

**🐞 SQL HATA AVCILIĞI YARIŞMASI**

**🎯 TEMEL SYNTAX HATALARI**

**Örnek 1: Eksik Noktalı Virgül**

sql

SELECT \* FROM ogrenciler

INSERT INTO ogrenciler (adi, yas) VALUES ('Ali', 20)

**✅ Düzeltilmiş Hali:**

sql

SELECT \* FROM ogrenciler;

INSERT INTO ogrenciler (adi, yas) VALUES ('Ali', 20);

**Örnek 2: Yanlış String Kullanımı**

sql

SELECT \* FROM ogrenciler WHERE adi = Ahmet

**✅ Düzeltilmiş Hali:**

sql

SELECT \* FROM ogrenciler WHERE adi = 'Ahmet';

**Örnek 3: Eksik FROM**

sql

SELECT adi, yas WHERE yas > 18

**✅ Düzeltilmiş Hali:**

sql

SELECT adi, yas FROM ogrenciler WHERE yas > 18;

**🔗 JOIN HATALARI**

**Örnek 4: Ambiguous Column**

sql

SELECT id, adi FROM ogrenciler

JOIN notlar ON ogrenciler.id = notlar.ogrenci\_id

**✅ Düzeltilmiş Hali:**

sql

SELECT ogrenciler.id, ogrenciler.adi FROM ogrenciler

JOIN notlar ON ogrenciler.id = notlar.ogrenci\_id;

**Örnek 5: Eksik JOIN Koşulu**

sql

SELECT o.adi, n.notu FROM ogrenciler o

JOIN notlar n

**✅ Düzeltilmiş Hali:**

sql

SELECT o.adi, n.notu FROM ogrenciler o

JOIN notlar n ON o.id = n.ogrenci\_id;

**Örnek 6: Yanlış JOIN Türü**

sql

SELECT o.adi, n.notu FROM ogrenciler o

LEFT JOIN notlar n ON o.id = n.ogrenci\_id

WHERE n.notu > 80

**✅ Düzeltilmiş Hali:**

sql

SELECT o.adi, n.notu FROM ogrenciler o

INNER JOIN notlar n ON o.id = n.ogrenci\_id

WHERE n.notu > 80;

**📊 GROUP BY HATALARI**

**Örnek 7: GROUP BY Eksik Sütun**

sql

SELECT sinif, adi, AVG(notu)

FROM ogrenciler

GROUP BY sinif

**✅ Düzeltilmiş Hali:**

sql

SELECT sinif, AVG(notu)

FROM ogrenciler

GROUP BY sinif;

-- VEYA

SELECT sinif, adi, AVG(notu)

FROM ogrenciler

GROUP BY sinif, adi;

**Örnek 8: HAVING'de Aggregate Olmayan Sütun**

sql

SELECT sinif, AVG(notu)

FROM ogrenciler

GROUP BY sinif

HAVING adi LIKE 'A%'

**✅ Düzeltilmiş Hali:**

sql

SELECT sinif, AVG(notu)

FROM ogrenciler

GROUP BY sinif

HAVING AVG(notu) > 70;

-- VEYA WHERE ile

SELECT sinif, AVG(notu)

FROM ogrenciler

WHERE adi LIKE 'A%'

GROUP BY sinif;

**🔢 SUBQUERY HATALARI**

**Örnek 9: Çoklu Sonuç Dönen Subquery**

sql

SELECT adi FROM ogrenciler

WHERE id = (SELECT ogrenci\_id FROM notlar WHERE notu > 80)

**✅ Düzeltilmiş Hali:**

sql

SELECT adi FROM ogrenciler

WHERE id IN (SELECT ogrenci\_id FROM notlar WHERE notu > 80);

**Örnek 10: Correlated Subquery Hatası**

sql

SELECT adi, (SELECT MAX(notu) FROM notlar) as max\_not

FROM ogrenciler

**✅ Düzeltilmiş Hali:**

sql

SELECT o.adi, (SELECT MAX(notu) FROM notlar n WHERE n.ogrenci\_id = o.id) as max\_not

FROM ogrenciler o;

**⏰ TARİH FONKSİYONU HATALARI**

**Örnek 11: Yanlış Tarih Formatı**

sql

SELECT \* FROM ogrenciler

WHERE dogum\_tarihi > 2020-01-01

**✅ Düzeltilmiş Hali:**

sql

SELECT \* FROM ogrenciler

WHERE dogum\_tarihi > '2020-01-01';

**Örnek 12: DATE\_ADD Yanlış Kullanım**

sql

SELECT DATE\_ADD(kayit\_tarihi, 30) FROM ogrenciler

**✅ Düzeltilmiş Hali:**

sql

SELECT DATE\_ADD(kayit\_tarihi, INTERVAL 30 DAY) FROM ogrenciler;

**🚨 MANTIK HATALARI**

**Örnek 13: Çelişkili WHERE Koşulları**

sql

SELECT \* FROM ogrenciler

WHERE yas > 20 AND yas < 18

**✅ Düzeltilmiş Hali:**

sql

SELECT \* FROM ogrenciler

WHERE yas > 18 AND yas < 20;

**Örnek 14: Yanlış LIKE Kullanımı**

sql

SELECT \* FROM ogrenciler

WHERE adi LIKE '%A%' OR adi LIKE '%B%'

**✅ Düzeltilmiş Hali:**

sql

SELECT \* FROM ogrenciler

WHERE adi LIKE '%A%' OR adi LIKE '%B%';

-- VEYA daha iyi:

SELECT \* FROM ogrenciler

WHERE adi REGEXP 'A|B';

**Örnek 15: ORDER BY ve LIMIT Karışıklığı**

sql

SELECT \* FROM ogrenciler

LIMIT 5

ORDER BY yas DESC

**✅ Düzeltilmiş Hali:**

sql

SELECT \* FROM ogrenciler

ORDER BY yas DESC

LIMIT 5;

**🎪 KARMAŞIK HATA SENARYOLARI**

**Örnek 16: Çoklu Hata İçeren Sorgu**

sql

SELECT o.adi, n.notu, COUNT(\*)

FROM ogrenciler o, notlar n

WHERE o.id = n.ogrenci\_id

GROUP BY o.adi

HAVING notu > avg(notu)

ORDER BY 4

**✅ Düzeltilmiş Hali:**

sql

SELECT o.adi, AVG(n.notu) as ortalama

FROM ogrenciler o

JOIN notlar n ON o.id = n.ogrenci\_id

GROUP BY o.id, o.adi

HAVING AVG(n.notu) > (SELECT AVG(notu) FROM notlar)

ORDER BY ortalama DESC;

**Örnek 17: INSERT Hatası**

sql

INSERT INTO ogrenciler (adi, yas, sinif)

VALUES ('Ayşe', 17)

**✅ Düzeltilmiş Hali:**

sql

INSERT INTO ogrenciler (adi, yas, sinif)

VALUES ('Ayşe', 17, '10-A');

-- VEYA sinif NULL izin veriyorsa:

INSERT INTO ogrenciler (adi, yas)

VALUES ('Ayşe', 17);

**Örnek 18: UPDATE ile Yanlış WHERE**

sql

UPDATE ogrenciler

SET notu = 100

WHERE adi = 'Ali'

**✅ Düzeltilmiş Hali:**

sql

UPDATE ogrenciler

SET notu = 100

WHERE adi = 'Ali' AND soyadi = 'Yılmaz';

-- VEYA ID ile

UPDATE ogrenciler

SET notu = 100

WHERE id = 5;

**🏆 YARIŞMA FORMATINDA HATA BULMA SORULARI**

**Soru 1: "Bu sorgu neden çalışmıyor?"**

sql

SELECT adi, AVG(notu) as ortalama

FROM ogrenciler

WHERE ortalama > 70

GROUP BY adi

**🎯 Hata:** WHERE'de aggregate fonksiyon kullanılamaz  
**✅ Çözüm:**

sql

SELECT adi, AVG(notu) as ortalama

FROM ogrenciler

GROUP BY adi

HAVING AVG(notu) > 70

**Soru 2: "Neden sadece 1 kayıt dönüyor?"**

sql

SELECT adi, (SELECT notu FROM notlar) as son\_not

FROM ogrenciler

**🎯 Hata:** Subquery çoklu kayıt dönüyor  
**✅ Çözüm:**

sql

SELECT o.adi,

(SELECT notu FROM notlar n WHERE n.ogrenci\_id = o.id ORDER BY tarih DESC LIMIT 1) as son\_not

FROM ogrenciler o

**Soru 3: "Performance sorunu neden?"**

sql

SELECT \* FROM ogrenciler

WHERE YEAR(dogum\_tarihi) = 2005

**🎯 Hata:** Fonksiyon WHERE'de index kullanımını engelliyor  
**✅ Çözüm:**

sql

SELECT \* FROM ogrenciler

WHERE dogum\_tarihi BETWEEN '2005-01-01' AND '2005-12-31'

**Soru 4: "Neden yanlış sayı dönüyor?"**

sql

SELECT COUNT(\*)

FROM ogrenciler o

LEFT JOIN notlar n ON o.id = n.ogrenci\_id

**🎯 Hata:** LEFT JOIN ile COUNT yanlış sonuç verir  
**✅ Çözüm:**

sql

SELECT COUNT(DISTINCT o.id)

FROM ogrenciler o

LEFT JOIN notlar n ON o.id = n.ogrenci\_id

**🎮 HATA AVCILIĞI OYUN FORMATI**

**Seviye 1: Hızlı Göz (2 dakika)**

sql

-- 5 basit hatalı sorgu - hızlıca bul

SELECT adi yas FROM ogrenciler WHERE adi = Ahmet

INSERT ogrenciler VALUES (1, 'Ali')

**Seviye 2: Dedektif (5 dakika)**

sql

-- 3 orta karmaşıklıkta sorgu

SELECT o.adi, n.notu FROM ogrenciler o, notlar n

SELECT sinif, adi, AVG(notu) FROM ogrenciler GROUP BY sinif

**Seviye 3: Uzman (8 dakika)**

sql

-- 2 karmaşık sorgu - çoklu hata içerir

SELECT adi, (SELECT notu FROM notlar) as notum

FROM ogrenciler WHERE AVG(notu) > 70 GROUP BY adi

**💡 HATA BULMA İPUÇLARI (Gençlere Rehber)**

**Debugging Stratejileri:**

1. **Hatayı okuyun!** - MySQL size ne olduğunu söylüyor
2. **Sorguyu parçalara ayırın** - önce SELECT, sonra WHERE, sonra JOIN
3. **Basit veriyle test edin** - küçük tablolarda deneyin
4. **Adım adım ilerleyin** - bir hatayı düzeltip diğerine geçin

**Sık Hata Mesajları ve Anlamları:**

* **"You have an error in your SQL syntax"** → Noktalı virgül, yazım hatası
* **"Unknown column"** → Sütun adı yanlış veya eksik
* **"Subquery returns more than 1 row"** → = yerine IN kullan
* **"Invalid use of group function"** → WHERE'de aggregate kullanma

**Yarışma Puanlama Sistemi:**

* ✅ Doğru hata tespiti: 10 puan
* ✅ Doğru çözüm önerisi: 15 puan
* ✅ Açıklama yapabilme: 5 puan
* ⏱️ Hızlı çözüm: bonus puan

**Gençlere Motivasyon:** "Unutmayın, en iyi programcılar en çok hata yapanlardır! Her hata sizi bir adım daha ileri götürür! 🚀"

* + - 1. Yapay Zekâ Okur Yazarlık

**Derinlemesine Yapay Zekâ Okuryazarlığı Nedir?**

Temel seviyede "AI nedir?" bilgisinin ötesine geçerek, AI'nın:

* **Nasıl çalıştığını**
* **Sınırlarını ve potansiyelini**
* **Etik boyutlarını**
* **Pratik uygulamalarını**
* **Geleceğini** anlamayı içeren kapsamlı bir beceridir.

**Temel Bileşenleri:**

1. **Teknik Anlayış:**
   * Makine öğrenmesi, derin öğrenme, doğal dil işleme nedir?
   * Model eğitimi nasıl gerçekleşir? Verinin önemi nedir?
   * Türleri: Görüntü işleme, ses tanıma, tahmin modelleri
2. **Pratik Beceriler:**
   * Prompt mühendisliği (AI'ya etkili soru sorma)
   * AI araçlarını verimli kullanma (ChatGPT, Midjourney, vs.)
   * Çıktıları değerlendirme ve doğrulama
3. **Eleştirel Düşünce:**
   * AI'nın yanlılıkları ve sınırlamaları
   * Gizlilik ve güvenlik riskleri
   * Otomasyonun sosyoekonomik etkileri
4. **Etik ve Sorumlu Kullanım:**
   * AI etiği ilkeleri
   * Telif hakları ve fikri mülkiyet
   * Şeffaflık ve sorumluluk

**Neden Bu Kadar Önemli?**

* **Mesleki Gelişim:** Tüm sektörlerde AI entegrasyonu artıyor
* **Günlük Yaşam:** Karar verme süreçlerimizde AI'nın rolü
* **Vatandaşlık:** AI regülasyonları ve politikalarını anlama
* **Eğitim:** Gelecek nesillere doğru rehberlik edebilme

**Başlangıç İçin Öneriler:**

1. **Temel Kavramlar:** YouTube'da açıklayıcı videolar izleyin
2. **Pratik Yapın:** ChatGPT, Deepseek, Copilot, Gemini gibi araçları deneyin
3. **Güncel Kalın:** AI haberlerini takip edin
4. **Topluluklara Katılın:** Yerel veya online AI gruplarına dahil olun

Hangi yönüyle özellikle ilgileniyorsunuz? Örneğin:

* İş hayatında AI kullanımı
* AI etiği
* Teknik detaylar
* Günlük hayatta AI

### **Temel Python Eğitimi**

**Temel Veri Tipleri Ve Değişkenler**

* **Değişken Kavramı**
  + Değişken nedir? Neye benzer? (Kutu metaforu)
  + İsimlendirme kuralları (isim = "Ahmet", yas = 25).
* **Temel Veri Türleri**
  + **String (Metin):** "Merhaba", 'A'
  + **Integer (Tam Sayı):** 10, -5, 1000
  + **Float (Ondalıklı Sayı):** 3.14, -0.5
  + **Boolean (Mantıksal):** True, False
* **Temel İşlemler**
  + Aritmetik operatörler: +, -, \*, /, // (tam bölme), % (mod), \*\* (üs)
  + String birleştirme: + operatörü.

**Kullanıcı Etkileşimi Ve Temel Fonksiyonlar**

* **Veri Girişi:**input()**Fonksiyonu**
  + isim = input("Adınızı girin: ")
  + input()'un her zaman string döndürdüğünün vurgulanması.
* **Tür Dönüşümleri (Type Casting)**
  + int(), float(), str() fonksiyonları.
  + Örnek: yas = int(input("Yaşınız: "))

**Koşul Yapıları (Karar Verme)**

* **Karşılaştırma Operatörleri:** ==, !=, >, <, >=, <=
* if**,**elif**,**else**Yapıları**
  + Basit örnekler: Not hesaplama, yaş kontrolü.

not = 85

if not >= 90:

print("AA")

elif not >= 80:

print("BB")

else:

print("Kaldı")

* **Mantıksal Operatörler:** and, or, not

**Döngüler (Tekrarlı İşlemler)**

* while**Döngüsü**
  + Belirli bir koşul sağlandığı sürece çalışır.

sayac = 1

while sayac <= 5:

print(sayac)

sayac += 1 # Sayacı 1 arttır

for**Döngüsü ve**range()**Fonksiyonu**

* Belirli bir sayıda veya bir koleksiyonun elemanları kadar döner.

for i in range(5): # 0'dan 4'e kadar

print(i)

for i in range(1, 6): # 1'den 5'e kadar

print(i)

**Veri Yapıları (Koleksiyonlar)**

* **Listeler (Lists)**
  + Sıralı, değiştirilebilir koleksiyon: meyveler = ["elma", "armut", "muz"]
  + Eleman ekleme (append), çıkarma (remove, pop), indeksleme.
* **Demetler (Tuples)**
  + Sıralı, **değiştirilemez** koleksiyon: koordinat = (10, 20)
* **Sözlükler (Dictionaries)**
  + Anahtar-Değer (Key-Value) çiftleri: ogrenci = {"ad": "Ayşe", "numara": 123}
  + Değerlere anahtar ile erişim: print(ogrenci["ad"])

**Fonksiyonlar**

* **Neden Fonksiyon Kullanırız?** (Kod tekrarını önleme, modülerlik)
* **Fonksiyon Tanımlama:**def
  + Parametreler ve return ifadesi.

def selamla(isim):

return "Merhaba, " + isim

mesaj = selamla("Mehmet")

print(mesaj)

**Hata Yönetimi (İstisna Kontrolü)**

* try**ve**except**Blokları**
  + Beklenmedik hataları yönetmek.

try:

sayi = int(input("Bir sayı girin: "))

except ValueError:

print("Lütfen geçerli bir sayı girin!")

* + - 1. Örnek Bir Proje Geliştirme

*//Uzun versiyon*

def listSum(words):

result = ""

for item in words:

if type(item) == int:

item = "."

if numbers[0] == item:

item = item.upper()

result += str(item)

return result

myList = ("a", "li", " ata", " bak", 33)

sonuc = listSum(myList)

print(sonuc)

*//Kısa versiyon*

def listSum(words):

items = [str(w) for w in words]

items[0] = items[0].upper()

return "".join(items[:-1]) + "."

myList = ("a", "li", " ata", " bak", 33)

sonuc = listSum(myList)

print(sonuc)

*//farklı versiyon*

def listSum(words):

items = [str(w) for w in words]

items[0] = items[0].capitalize()

return "".join(items[:-1]) + "."

myList = ("ali", " ata", " bak", 33)

sonuc = listSum(myList)

print(sonuc)

*//farklı versiyon 2*

def listSum(words):

items = [str(w) for w in words]

items[0] = items[0][0].upper() + items[0][1:] # İlk harf büyük, gerisi küçük

return "".join(items[:-1]) + "."

myList = ("ali", " ata", " bak", 33)

sonuc = listSum(myList)

print(sonuc)

### **Temel Web Uygulaması Geliştirme**

### **Temel Html Bilgisi**

**HTML Temel Bilgileri Özeti**

Aşağıda HTML'in temellerini içeren kapsamlı bir özet hazırladım:

<!DOCTYPE html>

<html lang="tr">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>HTML Temel Bilgileri Özeti</title>

<style>

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

}

body {

background-color: #f5f7fa;

color: #333;

line-height: 1.6;

padding: 20px;

}

.container {

max-width: 1200px;

margin: 0 auto;

background-color: white;

border-radius: 10px;

box-shadow: 0 0 20px rgba(0, 0, 0, 0.1);

overflow: hidden;

}

header {

background: linear-gradient(135deg, #3498db, #8e44ad);

color: white;

padding: 30px;

text-align: center;

}

h1 {

font-size: 2.5rem;

margin-bottom: 10px;

}

.subtitle {

font-size: 1.2rem;

opacity: 0.9;

}

.content {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(300px, 1fr));

gap: 20px;

padding: 30px;

}

.card {

background-color: #fff;

border-radius: 8px;

padding: 20px;

box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

transition: transform 0.3s ease;

}

.card:hover {

transform: translateY(-5px);

}

.card h2 {

color: #3498db;

margin-bottom: 15px;

padding-bottom: 10px;

border-bottom: 2px solid #f0f0f0;

}

.card h3 {

color: #2c3e50;

margin: 15px 0 10px;

}

.code-block {

background-color: #2c3e50;

color: #ecf0f1;

padding: 15px;

border-radius: 5px;

margin: 10px 0;

overflow-x: auto;

font-family: 'Courier New', monospace;

}

.tag {

display: inline-block;

background-color: #e74c3c;

color: white;

padding: 2px 8px;

border-radius: 4px;

font-size: 0.9rem;

margin-right: 5px;

}

.attribute {

display: inline-block;

background-color: #9b59b6;

color: white;

padding: 2px 8px;

border-radius: 4px;

font-size: 0.9rem;

margin-right: 5px;

}

footer {

text-align: center;

padding: 20px;

background-color: #34495e;

color: white;

margin-top: 20px;

}

@media (max-width: 768px) {

.content {

grid-template-columns: 1fr;

}

}

</style>

</head>

<body>

<div class="container">

<header>

<h1>HTML Temel Bilgileri Özeti</h1>

<p class="subtitle">Web geliştirmeye başlamak için ihtiyacınız olan temel HTML bilgileri</p>

</header>

<div class="content">

<div class="card">

<h2>HTML Nedir?</h2>

<p>HTML (HyperText Markup Language), web sayfalarının yapısını oluşturmak için kullanılan standart işaretleme dilidir. Tarayıcılar HTML kodlarını yorumlayarak görsel web sayfalarına dönüştürür.</p>

<h3>Temel HTML Yapısı</h3>

<div class="code-block">

&lt;!DOCTYPE html&gt;<br>

&lt;html&gt;<br>

&lt;head&gt;<br>

&nbsp;&nbsp;&lt;title&gt;Sayfa Başlığı&lt;/title&gt;<br>

&lt;/head&gt;<br>

&lt;body&gt;<br>

&nbsp;&nbsp;&lt;h1&gt;Başlık&lt;/h1&gt;<br>

&nbsp;&nbsp;&lt;p&gt;Paragraf&lt;/p&gt;<br>

&lt;/body&gt;<br>

&lt;/html&gt;

</div>

</div>

<div class="card">

<h2>Temel HTML Etiketleri</h2>

<h3>Başlık Etiketleri</h3>

<p><span class="tag">&lt;h1&gt;</span> - <span class="tag">&lt;h6&gt;</span>: Başlık etiketleri (h1 en büyük, h6 en küçük)</p>

<h3>Metin Etiketleri</h3>

<p><span class="tag">&lt;p&gt;</span>: Paragraf</p>

<p><span class="tag">&lt;strong&gt;</span> veya <span class="tag">&lt;b&gt;</span>: Kalın metin</p>

<p><span class="tag">&lt;em&gt;</span> veya <span class="tag">&lt;i&gt;</span>: İtalik metin</p>

<p><span class="tag">&lt;u&gt;</span>: Altı çizili metin</p>

<h3>Bağlantı ve Görsel</h3>

<p><span class="tag">&lt;a&gt;</span>: Bağlantı (link)</p>

<p><span class="tag">&lt;img&gt;</span>: Resim</p>

</div>

<div class="card">

<h2>Listeler ve Tablolar</h2>

<h3>Liste Etiketleri</h3>

<p><span class="tag">&lt;ul&gt;</span>: Sırasız liste</p>

<p><span class="tag">&lt;ol&gt;</span>: Sıralı liste</p>

<p><span class="tag">&lt;li&gt;</span>: Liste öğesi</p>

<h3>Tablo Etiketleri</h3>

<p><span class="tag">&lt;table&gt;</span>: Tablo</p>

<p><span class="tag">&lt;tr&gt;</span>: Tablo satırı</p>

<p><span class="tag">&lt;td&gt;</span>: Tablo hücresi</p>

<p><span class="tag">&lt;th&gt;</span>: Tablo başlık hücresi</p>

<div class="code-block">

&lt;table&gt;<br>

&nbsp;&nbsp;&lt;tr&gt;<br>

&nbsp;&nbsp;&nbsp;&nbsp;&lt;th&gt;İsim&lt;/th&gt;<br>

&nbsp;&nbsp;&nbsp;&nbsp;&lt;th&gt;Yaş&lt;/th&gt;<br>

&nbsp;&nbsp;&lt;/tr&gt;<br>

&nbsp;&nbsp;&lt;tr&gt;<br>

&nbsp;&nbsp;&nbsp;&nbsp;&lt;td&gt;Ahmet&lt;/td&gt;<br>

&nbsp;&nbsp;&nbsp;&nbsp;&lt;td&gt;25&lt;/td&gt;<br>

&nbsp;&nbsp;&lt;/tr&gt;<br>

&lt;/table&gt;

</div>

</div>

<div class="card">

<h2>Formlar</h2>

<p>HTML formları kullanıcıdan veri toplamak için kullanılır.</p>

<h3>Temel Form Etiketleri</h3>

<p><span class="tag">&lt;form&gt;</span>: Form kapsayıcısı</p>

<p><span class="tag">&lt;input&gt;</span>: Giriş alanı</p>

<p><span class="tag">&lt;textarea&gt;</span>: Çok satırlı metin alanı</p>

<p><span class="tag">&lt;select&gt;</span> ve <span class="tag">&lt;option&gt;</span>: Açılır liste</p>

<p><span class="tag">&lt;button&gt;</span>: Buton</p>

<h3>Input Türleri</h3>

<p><span class="attribute">type="text"</span>: Metin girişi</p>

<p><span class="attribute">type="password"</span>: Şifre girişi</p>

<p><span class="attribute">type="email"</span>: E-posta girişi</p>

<p><span class="attribute">type="submit"</span>: Gönder butonu</p>

</div>

<div class="card">

<h2>Önemli Özellikler</h2>

<h3>id ve class</h3>

<p><span class="attribute">id</span>: Bir öğeye benzersiz kimlik verir</p>

<p><span class="attribute">class</span>: Öğeleri gruplamak için kullanılır</p>

<h3>Diğer Önemli Özellikler</h3>

<p><span class="attribute">href</span>: Bağlantı adresi</p>

<p><span class="attribute">src</span>: Kaynak dosya yolu</p>

<p><span class="attribute">alt</span>: Alternatif metin (görseller için)</p>

<p><span class="attribute">style</span>: Satır içi CSS</p>

<div class="code-block">

&lt;a href="https://example.com"&gt;Örnek Site&lt;/a&gt;<br>

&lt;img src="resim.jpg" alt="Açıklama"&gt;<br>

&lt;p id="benzersiz" class="kirmizi"&gt;Metin&lt;/p&gt;

</div>

</div>

<div class="card">

<h2>Semantik HTML</h2>

<p>Semantik HTML, içeriğin anlamını daha iyi tanımlamak için kullanılan etiketlerdir.</p>

<h3>Semantik Etiketler</h3>

<p><span class="tag">&lt;header&gt;</span>: Sayfa veya bölüm başlığı</p>

<p><span class="tag">&lt;nav&gt;</span>: Navigasyon menüsü</p>

<p><span class="tag">&lt;main&gt;</span>: Ana içerik</p>

<p><span class="tag">&lt;section&gt;</span>: İçerik bölümü</p>

<p><span class="tag">&lt;article&gt;</span>: Bağımsız içerik</p>

<p><span class="tag">&lt;aside&gt;</span>: Yan içerik</p>

<p><span class="tag">&lt;footer&gt;</span>: Sayfa veya bölüm altbilgisi</p>

<p>Semantik etiketler, hem arama motorları hem de ekran okuyucular için daha anlamlı bir yapı oluşturur.</p>

</div>

</div>

<footer>

<p>HTML Temel Bilgileri Özeti &copy; 2023</p>

<p>Bu özet, HTML öğrenmeye başlayanlar için temel bilgileri içermektedir.</p>

</footer>

</div>

</body>

</html>

Bu HTML dosyası, temel HTML bilgilerini kapsamlı bir şekilde özetleyen modern ve responsive bir web sayfası oluşturur. Sayfa aşağıdaki bölümleri içerir:

1. **HTML Nedir?** - HTML'in tanımı ve temel yapısı
2. **Temel HTML Etiketleri** - Başlık, metin, bağlantı ve görsel etiketleri
3. **Listeler ve Tablolar** - Liste ve tablo oluşturma etiketleri
4. **Formlar** - Form oluşturma ve input türleri
5. **Önemli Özellikler** - id, class, href, src gibi önemli HTML özellikleri
6. **Semantik HTML** - Anlamsal HTML etiketleri

Sayfa, modern CSS teknikleri kullanılarak tasarlanmıştır ve tüm cihazlarda düzgün görüntülenecek şekilde responsive'dir.

### **Temel CSS Bilgisi**

# Temel CSS Bilgileri Özeti

Aşağıda CSS'in temellerini kapsamlı şekilde özetleyen modern bir sayfa hazırladım:

```html

<!DOCTYPE html>

<html lang="tr">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Temel CSS Bilgileri Özeti</title>

<style>

/\* Reset ve Temel Stiller \*/

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

}

body {

background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

color: #333;

line-height: 1.6;

min-height: 100vh;

padding: 20px;

}

.container {

max-width: 1200px;

margin: 0 auto;

background: white;

border-radius: 15px;

box-shadow: 0 15px 30px rgba(0, 0, 0, 0.2);

overflow: hidden;

}

/\* Header Stilleri \*/

header {

background: linear-gradient(135deg, #2c3e50, #3498db);

color: white;

padding: 40px 30px;

text-align: center;

position: relative;

}

header::after {

content: '';

position: absolute;

bottom: 0;

left: 0;

right: 0;

height: 4px;

background: linear-gradient(90deg, #e74c3c, #f39c12, #2ecc71, #3498db, #9b59b6);

}

h1 {

font-size: 2.8rem;

margin-bottom: 15px;

text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.3);

}

.subtitle {

font-size: 1.3rem;

opacity: 0.9;

max-width: 600px;

margin: 0 auto;

}

/\* İçerik Düzeni \*/

.content {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(350px, 1fr));

gap: 25px;

padding: 40px 30px;

}

.card {

background: white;

border-radius: 12px;

padding: 25px;

box-shadow: 0 8px 20px rgba(0, 0, 0, 0.1);

transition: all 0.3s ease;

border-left: 5px solid #3498db;

position: relative;

overflow: hidden;

}

.card::before {

content: '';

position: absolute;

top: 0;

left: 0;

width: 100%;

height: 4px;

background: linear-gradient(90deg, #3498db, #2ecc71);

}

.card:hover {

transform: translateY(-8px);

box-shadow: 0 15px 30px rgba(0, 0, 0, 0.15);

}

.card h2 {

color: #2c3e50;

margin-bottom: 20px;

padding-bottom: 12px;

border-bottom: 2px solid #ecf0f1;

display: flex;

align-items: center;

}

.card h2 i {

margin-right: 10px;

color: #3498db;

}

.card h3 {

color: #34495e;

margin: 18px 0 12px;

font-size: 1.2rem;

}

/\* Kod Blokları \*/

.code-block {

background: #2c3e50;

color: #ecf0f1;

padding: 18px;

border-radius: 8px;

margin: 15px 0;

overflow-x: auto;

font-family: 'Courier New', monospace;

font-size: 0.95rem;

line-height: 1.5;

border-left: 4px solid #e74c3c;

}

.code-block .comment {

color: #7f8c8d;

}

.code-block .property {

color: #3498db;

}

.code-block .value {

color: #2ecc71;

}

.code-block .selector {

color: #e74c3c;

}

/\* Etiket Stilleri \*/

.tag {

display: inline-block;

background: linear-gradient(135deg, #e74c3c, #c0392b);

color: white;

padding: 4px 12px;

border-radius: 20px;

font-size: 0.9rem;

margin: 5px 8px 5px 0;

font-weight: 600;

box-shadow: 0 2px 5px rgba(0, 0, 0, 0.2);

}

.attribute {

display: inline-block;

background: linear-gradient(135deg, #9b59b6, #8e44ad);

color: white;

padding: 4px 12px;

border-radius: 20px;

font-size: 0.9rem;

margin: 5px 8px 5px 0;

font-weight: 600;

box-shadow: 0 2px 5px rgba(0, 0, 0, 0.2);

}

.value-tag {

display: inline-block;

background: linear-gradient(135deg, #2ecc71, #27ae60);

color: white;

padding: 4px 12px;

border-radius: 20px;

font-size: 0.9rem;

margin: 5px 8px 5px 0;

font-weight: 600;

box-shadow: 0 2px 5px rgba(0, 0, 0, 0.2);

}

/\* Örnek Kutu \*/

.example-box {

background: #f8f9fa;

border: 2px dashed #bdc3c7;

border-radius: 8px;

padding: 20px;

margin: 15px 0;

}

.demo-element {

background: #3498db;

color: white;

padding: 15px;

border-radius: 5px;

margin: 10px 0;

text-align: center;

transition: all 0.3s ease;

}

.demo-element:hover {

background: #2980b9;

transform: scale(1.02);

}

/\* Footer \*/

footer {

text-align: center;

padding: 30px;

background: #2c3e50;

color: white;

margin-top: 20px;

}

footer p {

margin: 8px 0;

}

/\* Responsive Tasarım \*/

@media (max-width: 768px) {

.content {

grid-template-columns: 1fr;

padding: 25px 20px;

}

h1 {

font-size: 2.2rem;

}

.subtitle {

font-size: 1.1rem;

}

}

/\* Özel Stil Örnekleri \*/

.flex-demo {

display: flex;

gap: 10px;

margin: 15px 0;

}

.flex-item {

background: #e74c3c;

color: white;

padding: 15px;

border-radius: 5px;

flex: 1;

text-align: center;

}

.grid-demo {

display: grid;

grid-template-columns: repeat(2, 1fr);

gap: 10px;

margin: 15px 0;

}

.grid-item {

background: #2ecc71;

color: white;

padding: 15px;

border-radius: 5px;

text-align: center;

}

</style>

</head>

<body>

<div class="container">

<header>

<h1>Temel CSS Bilgileri Özeti</h1>

<p class="subtitle">Stil sayfaları ile web tasarımını öğrenmek için ihtiyacınız olan temel CSS bilgileri</p>

</header>

<div class="content">

<div class="card">

<h2><i>📖</i> CSS Nedir?</h2>

<p>CSS (Cascading Style Sheets), HTML elementlerinin görünümünü ve düzenini kontrol etmek için kullanılan stil dilidir.</p>

<h3>CSS Söz Dizimi</h3>

<div class="code-block">

<span class="selector">seçici</span> {<br>

&nbsp;&nbsp;<span class="property">özellik</span>: <span class="value">değer</span>;<br>

}

</div>

<h3>CSS Ekleme Yöntemleri</h3>

<p><span class="tag">Inline CSS</span>: HTML elementi içinde style attribute'u ile</p>

<p><span class="tag">Internal CSS</span>: &lt;style&gt; etiketleri arasında</p>

<p><span class="tag">External CSS</span>: Harici .css dosyası ile</p>

</div>

<div class="card">

<h2><i>🎯</i> Seçiciler (Selectors)</h2>

<h3>Temel Seçiciler</h3>

<p><span class="tag">Element</span>: p, h1, div gibi HTML etiketleri</p>

<p><span class="tag">Class</span>: .className şeklinde (birden fazla element)</p>

<p><span class="tag">ID</span>: #idName şeklinde (tek bir element)</p>

<h3>Gelişmiş Seçiciler</h3>

<p><span class="tag">Gruplama</span>: h1, p, div (virgül ile ayırma)</p>

<p><span class="tag">Çocuk</span>: div > p (direkt çocuk element)</p>

<p><span class="tag">Torun</span>: div p (tüm torun elementler)</p>

<div class="code-block">

<span class="comment">/\* Element seçici \*/</span><br>

<span class="selector">p</span> { <span class="property">color</span>: <span class="value">blue</span>; }<br><br>

<span class="comment">/\* Class seçici \*/</span><br>

<span class="selector">.kirmizi</span> { <span class="property">color</span>: <span class="value">red</span>; }<br><br>

<span class="comment">/\* ID seçici \*/</span><br>

<span class="selector">#baslik</span> { <span class="property">font-size</span>: <span class="value">24px</span>; }

</div>

</div>

<div class="card">

<h2><i>🎨</i> Renk ve Tipografi</h2>

<h3>Renk Tanımlama</h3>

<p><span class="tag">İsim</span>: red, blue, green</p>

<p><span class="tag">HEX</span>: #FF0000, #0000FF</p>

<p><span class="tag">RGB</span>: rgb(255, 0, 0)</p>

<p><span class="tag">RGBA</span>: rgba(255, 0, 0, 0.5) (alpha: şeffaflık)</p>

<h3>Yazı Stilleri</h3>

<p><span class="attribute">font-family</span>: Yazı tipi</p>

<p><span class="attribute">font-size</span>: Yazı boyutu</p>

<p><span class="attribute">font-weight</span>: Yazı kalınlığı</p>

<p><span class="attribute">text-align</span>: Metin hizalaması</p>

<div class="code-block">

<span class="selector">h1</span> {<br>

&nbsp;&nbsp;<span class="property">color</span>: <span class="value">#2c3e50</span>;<br>

&nbsp;&nbsp;<span class="property">font-family</span>: <span class="value">Arial, sans-serif</span>;<br>

&nbsp;&nbsp;<span class="property">font-size</span>: <span class="value">2rem</span>;<br>

&nbsp;&nbsp;<span class="property">text-align</span>: <span class="value">center</span>;<br>

}

</div>

</div>

<div class="card">

<h2><i>📐</i> Kutu Modeli (Box Model)</h2>

<p>Her HTML elementi bir kutu olarak düşünülür:</p>

<div class="example-box">

<div style="text-align: center; margin: 15px 0;">

<div style="background: #3498db; color: white; padding: 30px; margin: 20px; border: 5px solid #2980b9; border-radius: 5px;">

İçerik (Content)

</div>

</div>

</div>

<h3>Kutu Model Özellikleri</h3>

<p><span class="attribute">width</span>/<span class="attribute">height</span>: Genişlik/Yükseklik</p>

<p><span class="attribute">padding</span>: İç boşluk</p>

<p><span class="attribute">border</span>: Kenarlık</p>

<p><span class="attribute">margin</span>: Dış boşluk</p>

<div class="code-block">

<span class="selector">.kutu</span> {<br>

&nbsp;&nbsp;<span class="property">width</span>: <span class="value">300px</span>;<br>

&nbsp;&nbsp;<span class="property">padding</span>: <span class="value">20px</span>;<br>

&nbsp;&nbsp;<span class="property">border</span>: <span class="value">2px solid #333</span>;<br>

&nbsp;&nbsp;<span class="property">margin</span>: <span class="value">10px</span>;<br>

}

</div>

</div>

<div class="card">

<h2><i>🎪</i> Flexbox ve Grid</h2>

<h3>Flexbox (Esnek Kutu)</h3>

<p>Tek boyutlu düzenler için</p>

<div class="flex-demo">

<div class="flex-item">1</div>

<div class="flex-item">2</div>

<div class="flex-item">3</div>

</div>

<h3>CSS Grid</h3>

<p>İki boyutlu düzenler için</p>

<div class="grid-demo">

<div class="grid-item">A</div>

<div class="grid-item">B</div>

<div class="grid-item">C</div>

<div class="grid-item">D</div>

</div>

<div class="code-block">

<span class="comment">/\* Flexbox \*/</span><br>

<span class="selector">.container</span> {<br>

&nbsp;&nbsp;<span class="property">display</span>: <span class="value">flex</span>;<br>

&nbsp;&nbsp;<span class="property">justify-content</span>: <span class="value">space-between</span>;<br>

}<br><br>

<span class="comment">/\* Grid \*/</span><br>

<span class="selector">.container</span> {<br>

&nbsp;&nbsp;<span class="property">display</span>: <span class="value">grid</span>;<br>

&nbsp;&nbsp;<span class="property">grid-template-columns</span>: <span class="value">1fr 1fr</span>;<br>

}

</div>

</div>

<div class="card">

<h2><i>✨</i> Animasyon ve Geçişler</h2>

<h3>Transition (Geçiş)</h3>

<p>CSS özelliklerindeki değişikliklere yumuşak geçiş ekler</p>

<h3>Animation (Animasyon)</h3>

<p>Karmaşık animasyonlar için keyframes kullanır</p>

<div class="demo-element" style="animation: colorChange 3s infinite alternate;">

Animasyon Örneği

</div>

<div class="code-block">

<span class="comment">/\* Transition \*/</span><br>

<span class="selector">.button</span> {<br>

&nbsp;&nbsp;<span class="property">transition</span>: <span class="value">all 0.3s ease</span>;<br>

}<br><br>

<span class="comment">/\* Animation \*/</span><br>

<span class="selector">@keyframes colorChange</span> {<br>

&nbsp;&nbsp;<span class="selector">from</span> { <span class="property">background</span>: <span class="value">#3498db</span>; }<br>

&nbsp;&nbsp;<span class="selector">to</span> { <span class="property">background</span>: <span class="value">#e74c3c</span>; }<br>

}

</div>

<style>

@keyframes colorChange {

from { background: #3498db; }

to { background: #e74c3c; }

}

</style>

</div>

</div>

<footer>

<p><strong>Temel CSS Bilgileri Özeti</strong> &copy; 2023</p>

<p>CSS öğrenmeye başlamak için bu temel bilgilerle iyi bir başlangıç yapabilirsiniz.</p>

</footer>

</div>

</body>

</html>

```

Bu CSS özet sayfası şu temel konuları kapsamaktadır:

## 📋 İçerdiği Konular:

1. \*\*CSS Nedir?\*\* - Tanım ve temel söz dizimi

2. \*\*Seçiciler\*\* - Element, class, ID ve diğer seçici türleri

3. \*\*Renk ve Tipografi\*\* - Yazı stilleri ve renk tanımlama yöntemleri

4. \*\*Kutu Modeli\*\* - Padding, border, margin kavramları

5. \*\*Flexbox ve Grid\*\* - Modern layout sistemleri

6. \*\*Animasyonlar\*\* - Transition ve animation özellikleri

## 🎨 Tasarım Özellikleri:

- Modern gradient arka planlar

- Responsive grid layout

- Etkileşimli kartlar (hover efektleri)

- Renk kodlu syntax vurgulama

- Canlı demo örnekleri

- Mobil uyumlu tasarım

Sayfa, CSS'in temel kavramlarını hem teorik hem de pratik örneklerle açıklayarak öğrenmeyi kolaylaştırmaktadır.

### **Web Teknolojisi Bilgisi**

# Temel Web Teknolojileri Rehberi

Aşağıda modern web geliştirme için gerekli tüm temel teknolojileri kapsayan kapsamlı bir rehber hazırladım:

```html

<!DOCTYPE html>

<html lang="tr">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Temel Web Teknolojileri Rehberi</title>

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">

<style>

/\* Reset ve Temel Stiller \*/

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

}

:root {

--primary: #3498db;

--secondary: #2c3e50;

--success: #2ecc71;

--danger: #e74c3c;

--warning: #f39c12;

--info: #9b59b6;

--light: #ecf0f1;

--dark: #34495e;

}

body {

background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

color: #333;

line-height: 1.6;

min-height: 100vh;

padding: 20px;

}

.container {

max-width: 1400px;

margin: 0 auto;

background: white;

border-radius: 20px;

box-shadow: 0 20px 40px rgba(0, 0, 0, 0.2);

overflow: hidden;

}

/\* Header Stilleri \*/

header {

background: linear-gradient(135deg, var(--secondary), var(--primary));

color: white;

padding: 50px 40px;

text-align: center;

position: relative;

}

header::after {

content: '';

position: absolute;

bottom: 0;

left: 0;

right: 0;

height: 5px;

background: linear-gradient(90deg, var(--danger), var(--warning), var(--success), var(--primary), var(--info));

}

h1 {

font-size: 3.2rem;

margin-bottom: 20px;

text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.3);

}

.subtitle {

font-size: 1.4rem;

opacity: 0.9;

max-width: 700px;

margin: 0 auto 25px;

}

.tech-stats {

display: flex;

justify-content: center;

gap: 30px;

margin-top: 30px;

flex-wrap: wrap;

}

.stat {

background: rgba(255, 255, 255, 0.2);

padding: 15px 25px;

border-radius: 50px;

backdrop-filter: blur(10px);

}

/\* Navigasyon \*/

nav {

background: var(--dark);

padding: 0 40px;

}

.nav-links {

display: flex;

list-style: none;

flex-wrap: wrap;

}

.nav-links li {

margin: 0;

}

.nav-links a {

color: white;

text-decoration: none;

padding: 18px 25px;

display: block;

transition: all 0.3s ease;

border-bottom: 3px solid transparent;

}

.nav-links a:hover {

background: rgba(255, 255, 255, 0.1);

border-bottom: 3px solid var(--primary);

}

/\* İçerik Düzeni \*/

.content {

display: grid;

grid-template-columns: 1fr 300px;

gap: 30px;

padding: 40px;

}

.main-content {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(350px, 1fr));

gap: 30px;

}

.sidebar {

background: var(--light);

border-radius: 15px;

padding: 25px;

height: fit-content;

box-shadow: 0 5px 15px rgba(0, 0, 0, 0.1);

}

/\* Kart Stilleri \*/

.card {

background: white;

border-radius: 15px;

padding: 30px;

box-shadow: 0 10px 25px rgba(0, 0, 0, 0.1);

transition: all 0.3s ease;

border-top: 5px solid var(--primary);

position: relative;

overflow: hidden;

}

.card::before {

content: '';

position: absolute;

top: 0;

left: 0;

width: 100%;

height: 5px;

background: linear-gradient(90deg, var(--primary), var(--info));

}

.card:hover {

transform: translateY(-10px);

box-shadow: 0 15px 35px rgba(0, 0, 0, 0.15);

}

.card h2 {

color: var(--secondary);

margin-bottom: 20px;

padding-bottom: 15px;

border-bottom: 2px solid var(--light);

display: flex;

align-items: center;

font-size: 1.5rem;

}

.card h2 i {

margin-right: 12px;

font-size: 1.8rem;

}

.card h3 {

color: var(--dark);

margin: 20px 0 12px;

font-size: 1.3rem;

}

/\* Kod Blokları \*/

.code-block {

background: var(--dark);

color: var(--light);

padding: 20px;

border-radius: 10px;

margin: 15px 0;

overflow-x: auto;

font-family: 'Courier New', monospace;

font-size: 0.95rem;

line-height: 1.5;

border-left: 5px solid var(--danger);

}

.code-block .comment {

color: #7f8c8d;

}

.code-block .tag {

color: #e74c3c;

}

.code-block .attribute {

color: #3498db;

}

.code-block .value {

color: #2ecc71;

}

/\* Etiket Stilleri \*/

.tech-tag {

display: inline-block;

background: linear-gradient(135deg, var(--primary), var(--info));

color: white;

padding: 5px 15px;

border-radius: 20px;

font-size: 0.9rem;

margin: 5px 8px 5px 0;

font-weight: 600;

box-shadow: 0 3px 6px rgba(0, 0, 0, 0.2);

}

.level-tag {

display: inline-block;

background: var(--success);

color: white;

padding: 3px 10px;

border-radius: 15px;

font-size: 0.8rem;

margin-left: 10px;

}

/\* Örnek Kutu \*/

.example-box {

background: #f8f9fa;

border: 2px dashed #bdc3c7;

border-radius: 10px;

padding: 20px;

margin: 15px 0;

}

.demo-element {

background: var(--primary);

color: white;

padding: 15px;

border-radius: 8px;

margin: 10px 0;

text-align: center;

transition: all 0.3s ease;

}

/\* İlerleme Çubukları \*/

.skill-bar {

margin: 15px 0;

}

.skill-info {

display: flex;

justify-content: between;

margin-bottom: 5px;

}

.skill-level {

height: 10px;

background: #ecf0f1;

border-radius: 5px;

overflow: hidden;

}

.skill-progress {

height: 100%;

border-radius: 5px;

}

/\* Footer \*/

footer {

text-align: center;

padding: 40px;

background: var(--secondary);

color: white;

margin-top: 40px;

}

.footer-links {

display: flex;

justify-content: center;

gap: 30px;

margin: 20px 0;

flex-wrap: wrap;

}

.footer-links a {

color: var(--light);

text-decoration: none;

transition: color 0.3s ease;

}

.footer-links a:hover {

color: var(--primary);

}

/\* Responsive Tasarım \*/

@media (max-width: 1100px) {

.content {

grid-template-columns: 1fr;

}

.sidebar {

order: -1;

}

}

@media (max-width: 768px) {

.main-content {

grid-template-columns: 1fr;

}

h1 {

font-size: 2.5rem;

}

.subtitle {

font-size: 1.2rem;

}

.nav-links {

flex-direction: column;

}

.tech-stats {

flex-direction: column;

align-items: center;

gap: 15px;

}

}

/\* Özel Bileşenler \*/

.tech-roadmap {

margin: 20px 0;

}

.roadmap-item {

display: flex;

align-items: center;

margin: 15px 0;

padding: 10px;

border-radius: 8px;

background: #f8f9fa;

}

.roadmap-icon {

width: 40px;

height: 40px;

background: var(--primary);

border-radius: 50%;

display: flex;

align-items: center;

justify-content: center;

color: white;

margin-right: 15px;

flex-shrink: 0;

}

.comparison-table {

width: 100%;

border-collapse: collapse;

margin: 20px 0;

}

.comparison-table th, .comparison-table td {

padding: 12px 15px;

text-align: left;

border-bottom: 1px solid #ddd;

}

.comparison-table th {

background: var(--light);

color: var(--dark);

}

.comparison-table tr:hover {

background: #f5f5f5;

}

</style>

</head>

<body>

<div class="container">

<header>

<h1>Temel Web Teknolojileri Rehberi</h1>

<p class="subtitle">Modern web geliştirme için ihtiyacınız olan tüm temel teknolojiler ve araçlar</p>

<div class="tech-stats">

<div class="stat"><i class="fas fa-code"></i> 10+ Teknoloji</div>

<div class="stat"><i class="fas fa-layer-group"></i> 3 Katman</div>

<div class="stat"><i class="fas fa-rocket"></i> Modern Araçlar</div>

</div>

</header>

<nav>

<ul class="nav-links">

<li><a href="#frontend"><i class="fas fa-paint-brush"></i> Frontend</a></li>

<li><a href="#backend"><i class="fas fa-server"></i> Backend</a></li>

<li><a href="#database"><i class="fas fa-database"></i> Veritabanı</a></li>

<li><a href="#tools"><i class="fas fa-tools"></i> Araçlar</a></li>

<li><a href="#trends"><i class="fas fa-chart-line"></i> Trendler</a></li>

</ul>

</nav>

<div class="content">

<div class="main-content">

<!-- Frontend Teknolojileri -->

<div class="card" id="frontend">

<h2><i class="fas fa-paint-brush"></i> Frontend Teknolojileri</h2>

<p>Kullanıcıların gördüğü ve etkileşimde bulunduğu arayüzü oluşturan teknolojiler.</p>

<h3>Çekirdek Teknolojiler</h3>

<p><span class="tech-tag">HTML5</span> <span class="level-tag">Temel</span></p>

<p>Web sayfalarının yapısını ve içeriğini oluşturur.</p>

<p><span class="tech-tag">CSS3</span> <span class="level-tag">Temel</span></p>

<p>Web sayfalarının görünümünü ve düzenini kontrol eder.</p>

<p><span class="tech-tag">JavaScript</span> <span class="level-tag">Temel</span></p>

<p>Web sayfalarına etkileşim ve dinamik davranış ekler.</p>

<h3>Modern Framework'ler</h3>

<p><span class="tech-tag">React</span> <span class="level-tag">İleri</span></p>

<p>Facebook tarafından geliştirilen component tabanlı kütüphane.</p>

<p><span class="tech-tag">Vue.js</span> <span class="level-tag">İleri</span></p>

<p>Progressive framework - kademeli olarak benimsenebilir.</p>

<p><span class="tech-tag">Angular</span> <span class="level-tag">İleri</span></p>

<p>Google tarafından geliştirilen tam özellikli framework.</p>

<div class="code-block">

<span class="comment">// Modern React Bileşeni Örneği</span><br>

<span class="tag">import</span> React, { useState } <span class="tag">from</span> <span class="value">'react'</span>;<br><br>

<span class="tag">function</span> <span class="attribute">Counter</span>() {<br>

&nbsp;&nbsp;<span class="tag">const</span> [count, setCount] = <span class="attribute">useState</span>(0);<br><br>

&nbsp;&nbsp;<span class="tag">return</span> (<br>

&nbsp;&nbsp;&nbsp;&nbsp;<span class="tag">&lt;div&gt;</span><br>

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;<span class="tag">&lt;button</span> <span class="attribute">onClick</span>={() => setCount(count - 1)}<span class="tag">&gt;</span>-<span class="tag">&lt;/button&gt;</span><br>

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;<span class="tag">&lt;span&gt;</span>{count}<span class="tag">&lt;/span&gt;</span><br>

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;<span class="tag">&lt;button</span> <span class="attribute">onClick</span>={() => setCount(count + 1)}<span class="tag">&gt;</span>+<span class="tag">&lt;/button&gt;</span><br>

&nbsp;&nbsp;&nbsp;&nbsp;<span class="tag">&lt;/div&gt;</span><br>

&nbsp;&nbsp;);<br>

}

</div>

</div>

<!-- Backend Teknolojileri -->

<div class="card" id="backend">

<h2><i class="fas fa-server"></i> Backend Teknolojileri</h2>

<p>Sunucu tarafında çalışan, veri işleme ve iş mantığından sorumlu teknolojiler.</p>

<h3>Sunucu Taraflı Diller</h3>

<p><span class="tech-tag">Node.js</span> <span class="level-tag">Orta</span></p>

<p>JavaScript'i sunucu tarafında çalıştırmayı sağlar.</p>

<p><span class="tech-tag">Python</span> <span class="level-tag">Orta</span></p>

<p>Django ve Flask framework'leri ile popüler.</p>

<p><span class="tech-tag">PHP</span> <span class="level-tag">Orta</span></p>

<p>WordPress ve Laravel ile yaygın kullanım.</p>

<p><span class="tech-tag">Java</span> <span class="level-tag">İleri</span></p>

<p>Spring framework'ü ile kurumsal uygulamalar.</p>

<h3>Backend Framework'ler</h3>

<p><span class="tech-tag">Express.js</span> <span class="level-tag">Orta</span></p>

<p>Node.js için minimal web framework'ü.</p>

<p><span class="tech-tag">Django</span> <span class="level-tag">Orta</span></p>

<p>Python için "batteries included" framework.</p>

<p><span class="tech-tag">Laravel</span> <span class="level-tag">Orta</span></p>

<p>PHP için elegant syntax'a sahip framework.</p>

<div class="code-block">

<span class="comment">// Express.js Route Örneği</span><br>

<span class="tag">const</span> express = <span class="attribute">require</span>(<span class="value">'express'</span>);<br>

<span class="tag">const</span> app = <span class="attribute">express</span>();<br><br>

<span class="attribute">app</span>.<span class="attribute">get</span>(<span class="value">'/api/users'</span>, (req, res) => {<br>

&nbsp;&nbsp;<span class="tag">const</span> users = [<br>

&nbsp;&nbsp;&nbsp;&nbsp;{ id: 1, name: <span class="value">'Ahmet'</span> },<br>

&nbsp;&nbsp;&nbsp;&nbsp;{ id: 2, name: <span class="value">'Mehmet'</span> }<br>

&nbsp;&nbsp;];<br>

&nbsp;&nbsp;res.<span class="attribute">json</span>(users);<br>

});<br><br>

<span class="attribute">app</span>.<span class="attribute">listen</span>(3000, () => {<br>

&nbsp;&nbsp;console.<span class="attribute">log</span>(<span class="value">'Sunucu 3000 portunda çalışıyor'</span>);<br>

});

</div>

</div>

<!-- Veritabanı Teknolojileri -->

<div class="card" id="database">

<h2><i class="fas fa-database"></i> Veritabanı Teknolojileri</h2>

<p>Uygulama verilerinin saklandığı ve yönetildiği sistemler.</p>

<h3>İlişkisel Veritabanları</h3>

<p><span class="tech-tag">MySQL</span> <span class="level-tag">Orta</span></p>

<p>Açık kaynak, yaygın kullanılan RDBMS.</p>

<p><span class="tech-tag">PostgreSQL</span> <span class="level-tag">Orta</span></p>

<p>Gelişmiş özelliklere sahip açık kaynak RDBMS.</p>

<p><span class="tech-tag">SQLite</span> <span class="level-tag">Temel</span></p>

<p>Hafif, dosya tabanlı veritabanı.</p>

<h3>NoSQL Veritabanları</h3>

<p><span class="tech-tag">MongoDB</span> <span class="level-tag">Orta</span></p>

<p>Belge tabanlı NoSQL veritabanı.</p>

<p><span class="tech-tag">Redis</span> <span class="level-tag">Orta</span></p>

<p>Bellek içi anahtar-değer deposu.</p>

<p><span class="tech-tag">Firebase</span> <span class="level-tag">Orta</span></p>

<p>Google'ın realtime veritabanı çözümü.</p>

<div class="code-block">

<span class="comment">-- SQL Sorgu Örnekleri</span><br>

<span class="tag">CREATE TABLE</span> users (<br>

&nbsp;&nbsp;id <span class="attribute">INT</span> <span class="tag">PRIMARY KEY</span> <span class="tag">AUTO\_INCREMENT</span>,<br>

&nbsp;&nbsp;name <span class="attribute">VARCHAR</span>(50) <span class="tag">NOT NULL</span>,<br>

&nbsp;&nbsp;email <span class="attribute">VARCHAR</span>(100) <span class="tag">UNIQUE</span>,<br>

&nbsp;&nbsp;created\_at <span class="attribute">TIMESTAMP</span> <span class="tag">DEFAULT</span> <span class="attribute">CURRENT\_TIMESTAMP</span><br>

);<br><br>

<span class="tag">INSERT INTO</span> users (name, email) <br>

<span class="tag">VALUES</span> (<span class="value">'Ali'</span>, <span class="value">'ali@example.com'</span>);<br><br>

<span class="tag">SELECT</span> \* <span class="tag">FROM</span> users <span class="tag">WHERE</span> name = <span class="value">'Ali'</span>;

</div>

</div>

<!-- Geliştirme Araçları -->

<div class="card" id="tools">

<h2><i class="fas fa-tools"></i> Geliştirme Araçları</h2>

<p>Web geliştirme sürecini kolaylaştıran araçlar ve teknolojiler.</p>

<h3>Editörler ve IDE'ler</h3>

<p><span class="tech-tag">Visual Studio Code</span> <span class="level-tag">Temel</span></p>

<p>Microsoft'un popüler, hafif kod editörü.</p>

<p><span class="tech-tag">WebStorm</span> <span class="level-tag">Orta</span></p>

<p>JetBrains'in JavaScript IDE'si.</p>

<p><span class="tech-tag">Sublime Text</span> <span class="level-tag">Temel</span></p>

<p>Hızlı ve hafif text editörü.</p>

<h3>Versiyon Kontrol</h3>

<p><span class="tech-tag">Git</span> <span class="level-tag">Temel</span></p>

<p>Dağıtık versiyon kontrol sistemi.</p>

<p><span class="tech-tag">GitHub</span> <span class="level-tag">Temel</span></p>

<p>Git repositor'larını barındıran platform.</p>

<p><span class="tech-tag">GitLab</span> <span class="level-tag">Orta</span></p>

<p>CI/CD özellikleri ile kapsamlı platform.</p>

<h3>Paket Yöneticileri</h3>

<p><span class="tech-tag">npm</span> <span class="level-tag">Temel</span></p>

<p>Node.js paket yöneticisi.</p>

<p><span class="tech-tag">yarn</span> <span class="level-tag">Temel</span></p>

<p>Facebook'un npm alternatifi.</p>

</div>

<!-- Web Teknolojisi Trendleri -->

<div class="card" id="trends">

<h2><i class="fas fa-chart-line"></i> Teknoloji Trendleri</h2>

<p>Günümüzde popüler olan ve geleceği şekillendiren web teknolojileri.</p>

<h3>Modern Geliştirme</h3>

<p><span class="tech-tag">JAMstack</span> <span class="level-tag">İleri</span></p>

<p>JavaScript, APIs ve Markup ile modern web mimarisi.</p>

<p><span class="tech-tag">Serverless</span> <span class="level-tag">İleri</span></p>

<p>Sunucu yönetimi gerektirmeyen mimari.</p>

<p><span class="tech-tag">Micro Frontends</span> <span class="level-tag">İleri</span></p>

<p>Frontend uygulamalarını mikroservis olarak yapılandırma.</p>

<h3>Yükselen Teknolojiler</h3>

<p><span class="tech-tag">WebAssembly</span> <span class="level-tag">İleri</span></p>

<p>Web'de yüksek performanslı kod çalıştırma.</p>

<p><span class="tech-tag">Progressive Web Apps</span> <span class="level-tag">Orta</span></p>

<p>Native app benzeri web uygulamaları.</p>

<p><span class="tech-tag">GraphQL</span> <span class="level-tag">Orta</span></p>

<p>REST API'lere alternatif veri sorgulama dili.</p>

<table class="comparison-table">

<thead>

<tr>

<th>Teknoloji</th>

<th>Popülerlik</th>

<th>Öğrenme Zorluğu</th>

<th>İş Pazarı</th>

</tr>

</thead>

<tbody>

<tr>

<td>React</td>

<td>Çok Yüksek</td>

<td>Orta</td>

<td>Çok İyi</td>

</tr>

<tr>

<td>Vue.js</td>

<td>Yüksek</td>

<td>Düşük</td>

<td>İyi</td>

</tr>

<tr>

<td>Angular</td>

<td>Yüksek</td>

<td>Yüksek</td>

<td>İyi</td>

</tr>

<tr>

<td>Node.js</td>

<td>Çok Yüksek</td>

<td>Orta</td>

<td>Çok İyi</td>

</tr>

</tbody>

</table>

</div>

</div>

<!-- Sidebar -->

<div class="sidebar">

<h2><i class="fas fa-road"></i> Öğrenme Yol Haritası</h2>

<div class="tech-roadmap">

<div class="roadmap-item">

<div class="roadmap-icon">1</div>

<div>

<strong>Temel Web Teknolojileri</strong>

<p>HTML, CSS, JavaScript</p>

</div>

</div>

<div class="roadmap-item">

<div class="roadmap-icon">2</div>

<div>

<strong>Frontend Framework</strong>

<p>React, Vue veya Angular</p>

</div>

</div>

<div class="roadmap-item">

<div class="roadmap-icon">3</div>

<div>

<strong>Backend Geliştirme</strong>

<p>Node.js, Express, Veritabanı</p>

</div>

</div>

<div class="roadmap-item">

<div class="roadmap-icon">4</div>

<div>

<strong>Araçlar & DevOps</strong>

<p>Git, Docker, CI/CD</p>

</div>

</div>

</div>

<h2><i class="fas fa-lightbulb"></i> Hızlı İpuçları</h2>

<div class="example-box">

<p><strong>Responsive Tasarım:</strong> Mobile-first yaklaşımını benimseyin</p>

<p><strong>Performans:</strong> Görselleri optimize edin, lazy loading kullanın</p>

<p><strong>SEO:</strong> Semantik HTML ve meta tag'leri kullanın</p>

<p><strong>Güvenlik:</strong> Input validation ve HTTPS kullanın</p>

</div>

<h2><i class="fas fa-rocket"></i> Kaynaklar</h2>

<ul style="padding-left: 20px; margin-top: 15px;">

<li style="margin-bottom: 10px;">MDN Web Docs</li>

<li style="margin-bottom: 10px;">freeCodeCamp</li>

<li style="margin-bottom: 10px;">W3Schools</li>

<li style="margin-bottom: 10px;">Stack Overflow</li>

</ul>

</div>

</div>

<footer>

<h3>Web Teknolojileri Rehberi</h3>

<p>Modern web geliştirme dünyasında başarılı olmak için gerekli tüm temel bilgiler</p>

<div class="footer-links">

<a href="#"><i class="fab fa-github"></i> GitHub</a>

<a href="#"><i class="fab fa-codepen"></i> CodePen</a>

<a href="#"><i class="fab fa-dev"></i> Dev.to</a>

<a href="#"><i class="fab fa-free-code-camp"></i> freeCodeCamp</a>

</div>

<p style="margin-top: 20px; opacity: 0.8;">&copy; 2023 Web Teknolojileri Rehberi. Tüm hakları saklıdır.</p>

</footer>

</div>

</body>

</html>

```

## 🌟 Öne Çıkan Özellikler:

### 📚 Kapsamlı İçerik:

- \*\*Frontend Teknolojileri\*\*: HTML5, CSS3, JavaScript, React, Vue, Angular

- \*\*Backend Teknolojileri\*\*: Node.js, Python, PHP, Java ve framework'leri

- \*\*Veritabanları\*\*: MySQL, PostgreSQL, MongoDB, Redis

- \*\*Geliştirme Araçları\*\*: VS Code, Git, GitHub, npm

- \*\*Trend Teknolojiler\*\*: JAMstack, Serverless, PWA, GraphQL

### 🎨 Modern Tasarım:

- Responsive grid layout

- Gradient renkler ve animasyonlar

- Etkileşimli kartlar ve navigasyon

- Kod blokları ile örnekler

- İlerleme çubukları ve karşılaştırma tabloları

### 📱 Kullanıcı Deneyimi:

- Smooth scrolling navigasyon

- Hover efektleri ve geçişler

- Mobil uyumlu tasarım

- Öğrenme yol haritası

- Pratik ipuçları ve kaynaklar

Bu rehber, web geliştirme yolculuğuna başlayanlar için tüm temel teknolojileri kapsamlı şekilde açıklamakta ve modern web geliştirme ekosistemini anlamalarına yardımcı olmaktadır.

### **Doküman Versiyon Takibi**

### **Git Eğitimi**

# Temel Git Eğitimi ve Komutları Rehberi

Aşağıda Git versiyon kontrol sistemi için kapsamlı bir eğitim rehberi hazırladım:

```html

<!DOCTYPE html>

<html lang="tr">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Temel Git Eğitimi - Versiyon Kontrol Sistemi</title>

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">

<style>

/\* Reset ve Temel Stiller \*/

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

}

:root {

--primary: #f05133;

--secondary: #2c3e50;

--success: #28a745;

--warning: #ffc107;

--info: #17a2b8;

--light: #f8f9fa;

--dark: #343a40;

--git-orange: #f14e32;

--git-dark: #1e1e1e;

}

body {

background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

color: #333;

line-height: 1.6;

min-height: 100vh;

padding: 20px;

}

.container {

max-width: 1400px;

margin: 0 auto;

background: white;

border-radius: 20px;

box-shadow: 0 20px 40px rgba(0, 0, 0, 0.2);

overflow: hidden;

}

/\* Header Stilleri \*/

header {

background: linear-gradient(135deg, var(--git-dark), var(--git-orange));

color: white;

padding: 50px 40px;

text-align: center;

position: relative;

}

header::after {

content: '';

position: absolute;

bottom: 0;

left: 0;

right: 0;

height: 5px;

background: linear-gradient(90deg, var(--success), var(--warning), var(--info), var(--git-orange));

}

.git-logo {

font-size: 4rem;

margin-bottom: 20px;

text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.3);

}

h1 {

font-size: 3.2rem;

margin-bottom: 20px;

text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.3);

}

.subtitle {

font-size: 1.4rem;

opacity: 0.9;

max-width: 700px;

margin: 0 auto 25px;

}

.git-stats {

display: flex;

justify-content: center;

gap: 30px;

margin-top: 30px;

flex-wrap: wrap;

}

.stat {

background: rgba(255, 255, 255, 0.2);

padding: 15px 25px;

border-radius: 50px;

backdrop-filter: blur(10px);

}

/\* Navigasyon \*/

nav {

background: var(--git-dark);

padding: 0 40px;

}

.nav-links {

display: flex;

list-style: none;

flex-wrap: wrap;

}

.nav-links li {

margin: 0;

}

.nav-links a {

color: white;

text-decoration: none;

padding: 18px 25px;

display: block;

transition: all 0.3s ease;

border-bottom: 3px solid transparent;

}

.nav-links a:hover {

background: rgba(255, 255, 255, 0.1);

border-bottom: 3px solid var(--git-orange);

}

/\* İçerik Düzeni \*/

.content {

display: grid;

grid-template-columns: 1fr 350px;

gap: 30px;

padding: 40px;

}

.main-content {

display: flex;

flex-direction: column;

gap: 30px;

}

.sidebar {

background: var(--light);

border-radius: 15px;

padding: 25px;

height: fit-content;

box-shadow: 0 5px 15px rgba(0, 0, 0, 0.1);

border-left: 5px solid var(--git-orange);

}

/\* Kart Stilleri \*/

.card {

background: white;

border-radius: 15px;

padding: 30px;

box-shadow: 0 10px 25px rgba(0, 0, 0, 0.1);

transition: all 0.3s ease;

border-top: 5px solid var(--git-orange);

position: relative;

overflow: hidden;

}

.card::before {

content: '';

position: absolute;

top: 0;

left: 0;

width: 100%;

height: 5px;

background: linear-gradient(90deg, var(--git-orange), var(--info));

}

.card:hover {

transform: translateY(-5px);

box-shadow: 0 15px 35px rgba(0, 0, 0, 0.15);

}

.card h2 {

color: var(--secondary);

margin-bottom: 20px;

padding-bottom: 15px;

border-bottom: 2px solid var(--light);

display: flex;

align-items: center;

font-size: 1.5rem;

}

.card h2 i {

margin-right: 12px;

font-size: 1.8rem;

color: var(--git-orange);

}

.card h3 {

color: var(--dark);

margin: 20px 0 12px;

font-size: 1.3rem;

}

/\* Kod Blokları \*/

.code-block {

background: var(--dark);

color: var(--light);

padding: 20px;

border-radius: 10px;

margin: 15px 0;

overflow-x: auto;

font-family: 'Courier New', monospace;

font-size: 0.95rem;

line-height: 1.5;

border-left: 5px solid var(--success);

}

.code-block .comment {

color: #6c757d;

}

.code-block .command {

color: #e83e8c;

font-weight: bold;

}

.code-block .option {

color: #20c997;

}

.code-block .path {

color: #fd7e14;

}

/\* Etiket Stilleri \*/

.git-tag {

display: inline-block;

background: linear-gradient(135deg, var(--git-orange), #e44d26);

color: white;

padding: 5px 15px;

border-radius: 20px;

font-size: 0.9rem;

margin: 5px 8px 5px 0;

font-weight: 600;

box-shadow: 0 3px 6px rgba(0, 0, 0, 0.2);

}

.level-tag {

display: inline-block;

background: var(--success);

color: white;

padding: 3px 10px;

border-radius: 15px;

font-size: 0.8rem;

margin-left: 10px;

}

/\* İşlem Adımları \*/

.workflow-step {

display: flex;

align-items: flex-start;

margin: 20px 0;

padding: 15px;

background: #f8f9fa;

border-radius: 10px;

border-left: 4px solid var(--git-orange);

}

.step-number {

background: var(--git-orange);

color: white;

width: 30px;

height: 30px;

border-radius: 50%;

display: flex;

align-items: center;

justify-content: center;

margin-right: 15px;

flex-shrink: 0;

font-weight: bold;

}

/\* Diyagram \*/

.git-flow {

background: white;

border-radius: 10px;

padding: 20px;

margin: 20px 0;

border: 2px solid #e9ecef;

}

.flow-item {

padding: 15px;

margin: 10px 0;

border-radius: 8px;

text-align: center;

font-weight: bold;

}

.working-directory { background: #e3f2fd; border-left: 4px solid #2196f3; }

.staging-area { background: #fff3e0; border-left: 4px solid #ff9800; }

.local-repo { background: #e8f5e8; border-left: 4px solid #4caf50; }

.remote-repo { background: #fce4ec; border-left: 4px solid #e91e63; }

/\* Footer \*/

footer {

text-align: center;

padding: 40px;

background: var(--git-dark);

color: white;

margin-top: 40px;

}

.footer-links {

display: flex;

justify-content: center;

gap: 30px;

margin: 20px 0;

flex-wrap: wrap;

}

.footer-links a {

color: var(--light);

text-decoration: none;

transition: color 0.3s ease;

}

.footer-links a:hover {

color: var(--git-orange);

}

/\* Responsive Tasarım \*/

@media (max-width: 1100px) {

.content {

grid-template-columns: 1fr;

}

.sidebar {

order: -1;

}

}

@media (max-width: 768px) {

.main-content {

grid-template-columns: 1fr;

}

h1 {

font-size: 2.5rem;

}

.subtitle {

font-size: 1.2rem;

}

.nav-links {

flex-direction: column;

}

.git-stats {

flex-direction: column;

align-items: center;

gap: 15px;

}

}

/\* Özel Bileşenler \*/

.command-grid {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(300px, 1fr));

gap: 15px;

margin: 20px 0;

}

.command-item {

background: #f8f9fa;

padding: 15px;

border-radius: 8px;

border-left: 4px solid var(--info);

}

.command-name {

font-weight: bold;

color: var(--dark);

margin-bottom: 5px;

}

.command-desc {

font-size: 0.9rem;

color: #6c757d;

}

.branch-visual {

background: #2c3e50;

color: white;

padding: 20px;

border-radius: 10px;

margin: 20px 0;

font-family: 'Courier New', monospace;

}

.commit {

padding: 8px 15px;

margin: 5px 0;

background: #34495e;

border-radius: 5px;

}

.branch {

color: #2ecc71;

font-weight: bold;

}

</style>

</head>

<body>

<div class="container">

<header>

<div class="git-logo">

<i class="fab fa-git-alt"></i>

</div>

<h1>Temel Git Eğitimi</h1>

<p class="subtitle">Versiyon kontrol sistemini öğrenmek için kapsamlı rehber</p>

<div class="git-stats">

<div class="stat"><i class="fas fa-code-branch"></i> 20+ Komut</div>

<div class="stat"><i class="fas fa-project-diagram"></i> Branch Yönetimi</div>

<div class="stat"><i class="fas fa-users"></i> Ekip Çalışması</div>

</div>

</header>

<nav>

<ul class="nav-links">

<li><a href="#baslangic"><i class="fas fa-play"></i> Başlangıç</a></li>

<li><a href="#temel-komutlar"><i class="fas fa-terminal"></i> Temel Komutlar</a></li>

<li><a href="#branching"><i class="fas fa-code-branch"></i> Branching</a></li>

<li><a href="#remote"><i class="fas fa-cloud"></i> Remote Repo</a></li>

<li><a href="#advanced"><i class="fas fa-rocket"></i> İleri Seviye</a></li>

</ul>

</nav>

<div class="content">

<div class="main-content">

<!-- Git Başlangıç -->

<div class="card" id="baslangic">

<h2><i class="fas fa-play"></i> Git'e Başlangıç</h2>

<p>Git, yazılım geliştirme sürecinde dosyaların değişikliklerini takip eden dağıtık bir versiyon kontrol sistemidir.</p>

<h3>Git Kurulumu</h3>

<div class="code-block">

<span class="comment"># Windows için</span><br>

<span class="comment"># https://git-scm.com/ adresinden indirin</span><br><br>

<span class="comment"># macOS için</span><br>

<span class="command">brew</span> <span class="option">install</span> git<br><br>

<span class="comment"># Linux (Debian/Ubuntu) için</span><br>

<span class="command">sudo</span> <span class="command">apt</span> <span class="option">install</span> git

</div>

<h3>İlk Yapılandırma</h3>

<div class="code-block">

<span class="comment"># Kullanıcı bilgilerini ayarlama</span><br>

<span class="command">git</span> <span class="option">config</span> <span class="option">--global</span> user.name <span class="path">"Adınız Soyadınız"</span><br>

<span class="command">git</span> <span class="option">config</span> <span class="option">--global</span> user.email <span class="path">"email@adresiniz.com"</span><br><br>

<span class="comment"># Ayarları kontrol etme</span><br>

<span class="command">git</span> <span class="option">config</span> <span class="option">--list</span>

</div>

<h3>Git Çalışma Mantığı</h3>

<div class="git-flow">

<div class="flow-item working-directory">

<i class="fas fa-folder-open"></i> Working Directory<br>

<small>Çalışma dizininiz</small>

</div>

<div class="flow-item staging-area">

<i class="fas fa-layer-group"></i> Staging Area<br>

<small>Commit için hazırlanan dosyalar</small>

</div>

<div class="flow-item local-repo">

<i class="fas fa-database"></i> Local Repository<br>

<small>Yerel deponuz</small>

</div>

<div class="flow-item remote-repo">

<i class="fas fa-cloud"></i> Remote Repository<br>

<small>Uzak depo (GitHub, GitLab)</small>

</div>

</div>

</div>

<!-- Temel Komutlar -->

<div class="card" id="temel-komutlar">

<h2><i class="fas fa-terminal"></i> Temel Git Komutları</h2>

<p>Günlük kullanımda en sık ihtiyaç duyacağınız Git komutları</p>

<div class="command-grid">

<div class="command-item">

<div class="command-name">git init</div>

<div class="command-desc">Yeni bir Git deposu oluşturur</div>

</div>

<div class="command-item">

<div class="command-name">git clone [url]</div>

<div class="command-desc">Uzak depoyu yereline kopyalar</div>

</div>

<div class="command-item">

<div class="command-name">git status</div>

<div class="command-desc">Dosya durumlarını gösterir</div>

</div>

<div class="command-item">

<div class="command-name">git add [file]</div>

<div class="command-desc">Dosyaları staging area'ya ekler</div>

</div>

<div class="command-item">

<div class="command-name">git commit -m "mesaj"</div>

<div class="command-desc">Değişiklikleri kaydeder</div>

</div>

<div class="command-item">

<div class="command-name">git log</div>

<div class="command-desc">Commit geçmişini gösterir</div>

</div>

</div>

<h3>Günlük Çalışma Akışı</h3>

<div class="workflow-step">

<div class="step-number">1</div>

<div>

<strong>Değişiklikleri Kontrol Et</strong>

<div class="code-block">

<span class="command">git</span> <span class="option">status</span>

</div>

</div>

</div>

<div class="workflow-step">

<div class="step-number">2</div>

<div>

<strong>Değişiklikleri Staging'e Ekle</strong>

<div class="code-block">

<span class="comment"># Tüm değişiklikleri ekle</span><br>

<span class="command">git</span> <span class="option">add</span> <span class="path">.</span><br><br>

<span class="comment"># Belirli dosyaları ekle</span><br>

<span class="command">git</span> <span class="option">add</span> <span class="path">dosya1.js dosya2.css</span>

</div>

</div>

</div>

<div class="workflow-step">

<div class="step-number">3</div>

<div>

<strong>Değişiklikleri Commit Et</strong>

<div class="code-block">

<span class="command">git</span> <span class="option">commit</span> <span class="option">-m</span> <span class="path">"Kullanıcı arayüzü güncellemeleri"</span>

</div>

</div>

</div>

</div>

<!-- Branching -->

<div class="card" id="branching">

<h2><i class="fas fa-code-branch"></i> Branch (Dal) Yönetimi</h2>

<p>Branch'ler, projenizin farklı versiyonlarını yönetmenizi sağlar.</p>

<h3>Temel Branch Komutları</h3>

<div class="code-block">

<span class="comment"># Branch listesini görüntüle</span><br>

<span class="command">git</span> <span class="option">branch</span><br><br>

<span class="comment"># Yeni branch oluştur</span><br>

<span class="command">git</span> <span class="option">branch</span> <span class="path">yeni-ozellik</span><br><br>

<span class="comment"># Branch değiştir</span><br>

<span class="command">git</span> <span class="option">checkout</span> <span class="path">yeni-ozellik</span><br><br>

<span class="comment"># Branch oluştur ve geçiş yap (kısa yol)</span><br>

<span class="command">git</span> <span class="option">checkout</span> <span class="option">-b</span> <span class="path">yeni-ozellik</span>

</div>

<h3>Branch Görselleştirme</h3>

<div class="branch-visual">

<div class="commit">main\* (initial commit)</div>

<div style="margin-left: 20px;">

<div class="commit">feature/login (<span class="branch">branch from main</span>)</div>

<div class="commit">feature/payment (<span class="branch">branch from main</span>)</div>

</div>

</div>

<h3>Merge (Birleştirme) İşlemleri</h3>

<div class="code-block">

<span class="comment"># Önce main branch'ine geç</span><br>

<span class="command">git</span> <span class="option">checkout</span> <span class="path">main</span><br><br>

<span class="comment"># Yeni özellik branch'ini main'e birleştir</span><br>

<span class="command">git</span> <span class="option">merge</span> <span class="path">yeni-ozellik</span>

</div>

</div>

<!-- Remote Repo -->

<div class="card" id="remote">

<h2><i class="fas fa-cloud"></i> Uzak Depo (Remote) İşlemleri</h2>

<p>GitHub, GitLab veya Bitbucket gibi platformlarla çalışma</p>

<h3>Remote Komutları</h3>

<div class="code-block">

<span class="comment"># Uzak depo ekleme</span><br>

<span class="command">git</span> <span class="option">remote</span> <span class="option">add</span> <span class="path">origin https://github.com/kullanici/repo.git</span><br><br>

<span class="comment"># Uzak depoya gönderme (push)</span><br>

<span class="command">git</span> <span class="option">push</span> <span class="option">-u</span> <span class="path">origin main</span><br><br>

<span class="comment"># Uzak depodan çekme (pull)</span><br>

<span class="command">git</span> <span class="option">pull</span> <span class="path">origin main</span><br><br>

<span class="comment"># Uzak depoyu görüntüle</span><br>

<span class="command">git</span> <span class="option">remote</span> <span class="option">-v</span>

</div>

<h3>Ekip Çalışması Akışı</h3>

<div class="workflow-step">

<div class="step-number">1</div>

<div>

<strong>Son Değişiklikleri Al</strong>

<div class="code-block">

<span class="command">git</span> <span class="option">pull</span> <span class="path">origin main</span>

</div>

</div>

</div>

<div class="workflow-step">

<div class="step-number">2</div>

<div>

<strong>Değişiklikleri Yap ve Commit Et</strong>

<div class="code-block">

<span class="command">git</span> <span class="option">add</span> <span class="path">.</span><br>

<span class="command">git</span> <span class="option">commit</span> <span class="option">-m</span> <span class="path">"Yeni özellik eklendi"</span>

</div>

</div>

</div>

<div class="workflow-step">

<div class="step-number">3</div>

<div>

<strong>Değişiklikleri Gönder</strong>

<div class="code-block">

<span class="command">git</span> <span class="option">push</span> <span class="path">origin main</span>

</div>

</div>

</div>

</div>

<!-- İleri Seviye -->

<div class="card" id="advanced">

<h2><i class="fas fa-rocket"></i> İleri Seviye Git Komutları</h2>

<p>Daha gelişmiş senaryolar için kullanışlı Git komutları</p>

<h3>Değişiklikleri Geri Alma</h3>

<div class="code-block">

<span class="comment"># Son commit'i geri al (soft - değişiklikler kalır)</span><br>

<span class="command">git</span> <span class="option">reset</span> <span class="option">--soft</span> <span class="path">HEAD~1</span><br><br>

<span class="comment"># Son commit'i geri al (hard - değişiklikler silinir)</span><br>

<span class="command">git</span> <span class="option">reset</span> <span class="option">--hard</span> <span class="path">HEAD~1</span><br><br>

<span class="comment"># Belirli bir dosyayı geri al</span><br>

<span class="command">git</span> <span class="option">checkout</span> <span class="path">-- dosyaadi.js</span>

</div>

<h3>Stash (Geçici Saklama)</h3>

<div class="code-block">

<span class="comment"># Geçerli değişiklikleri geçici olarak sakla</span><br>

<span class="command">git</span> <span class="option">stash</span><br><br>

<span class="comment"># Saklanan değişiklikleri listele</span><br>

<span class="command">git</span> <span class="option">stash</span> <span class="option">list</span><br><br>

<span class="comment"># Son saklanan değişiklikleri geri yükle</span><br>

<span class="command">git</span> <span class="option">stash</span> <span class="option">pop</span>

</div>

<h3>Hata Ayıklama ve Log</h3>

<div class="code-block">

<span class="comment"># Detaylı log görüntüleme</span><br>

<span class="command">git</span> <span class="option">log</span> <span class="option">--oneline</span> <span class="option">--graph</span> <span class="option">--all</span><br><br>

<span class="comment"># Belirli bir dosyanın geçmişi</span><br>

<span class="command">git</span> <span class="option">log</span> <span class="option">-p</span> <span class="path">dosyaadi.js</span><br><br>

<span class="comment"># Kim, ne zaman, hangi dosyayı değiştirdi</span><br>

<span class="command">git</span> <span class="option">blame</span> <span class="path">dosyaadi.js</span>

</div>

</div>

</div>

<!-- Sidebar -->

<div class="sidebar">

<h2><i class="fas fa-road"></i> Git Öğrenme Yol Haritası</h2>

<div class="workflow-step">

<div class="step-number">1</div>

<div>

<strong>Temel Kavramlar</strong>

<p>Repository, Commit, Branch</p>

</div>

</div>

<div class="workflow-step">

<div class="step-number">2</div>

<div>

<strong>Günlük Komutlar</strong>

<p>add, commit, push, pull</p>

</div>

</div>

<div class="workflow-step">

<div class="step-number">3</div>

<div>

<strong>Branch Yönetimi</strong>

<p>merge, rebase, conflict</p>

</div>

</div>

<div class="workflow-step">

<div class="step-number">4</div>

<div>

<strong>Ekip Çalışması</strong>

<p>Pull Request, Code Review</p>

</div>

</div>

<h2><i class="fas fa-lightbulb"></i> En İyi Uygulamalar</h2>

<div class="code-block">

<span class="comment"># Commit mesajları anlamlı olsun</span><br>

<span class="path">"feat: kullanıcı giriş sayfası eklendi"</span><br>

<span class="path">"fix: güvenlik açığı kapatıldı"</span><br>

<span class="path">"docs: readme güncellendi"</span>

</div>

<ul style="padding-left: 20px; margin-top: 15px;">

<li style="margin-bottom: 10px;">Küçük ve sık commit yapın</li>

<li style="margin-bottom: 10px;">Branch'lerde çalışın</li>

<li style="margin-bottom: 10px;">Commit'lerden önce test edin</li>

<li style="margin-bottom: 10px;">Anlamlı commit mesajları yazın</li>

</ul>

<h2><i class="fas fa-exclamation-triangle"></i> Sık Yapılan Hatalar</h2>

<ul style="padding-left: 20px; margin-top: 15px;">

<li style="margin-bottom: 10px;">Doğrudan main branch'inde çalışmak</li>

<li style="margin-bottom: 10px;">Commit mesajı olarak "update" yazmak</li>

<li style="margin-bottom: 10px;">Pull yapmadan push denemek</li>

<li style="margin-bottom: 10px;">.gitignore dosyasını kullanmamak</li>

</ul>

</div>

</div>

<footer>

<h3>Git Versiyon Kontrol Sistemi</h3>

<p>Profesyonel yazılım geliştirme için temel araç</p>

<div class="footer-links">

<a href="#"><i class="fab fa-github"></i> GitHub Docs</a>

<a href="#"><i class="fas fa-book"></i> Git Book</a>

<a href="#"><i class="fas fa-graduation-cap"></i> Git Tutorials</a>

<a href="#"><i class="fas fa-question-circle"></i> Git Cheat Sheet</a>

</div>

<p style="margin-top: 20px; opacity: 0.8;">&copy; 2023 Git Eğitim Rehberi. Linus Torvalds'a teşekkürler!</p>

</footer>

</div>

</body>

</html>

```

## 🚀 Git Eğitim Rehberi Özellikleri:

### 📚 Kapsamlı İçerik:

- \*\*Git Temelleri\*\*: Kurulum, yapılandırma, çalışma mantığı

- \*\*Temel Komutlar\*\*: init, add, commit, status, log

- \*\*Branch Yönetimi\*\*: checkout, merge, branch oluşturma

- \*\*Remote İşlemleri\*\*: push, pull, clone, remote yönetimi

- \*\*İleri Seviye\*\*: reset, stash, log incelemeleri

### 🎨 Görsel Öğeler:

- Git workflow görselleştirmesi

- Branch diyagramları

- Kod blokları ile gerçek komut örnekleri

- Adım adım çalışma akışları

- Renk kodlu komut gösterimi

### 💡 Pratik Bilgiler:

- Günlük kullanım için en iyi uygulamalar

- Sık yapılan hatalar ve çözümleri

- Ekip çalışması için ipuçları

- Commit mesajı standartları

### 🛠️ Kullanıcı Dostu:

- Responsive tasarım

- Kolay navigasyon

- Mobil uyumluluk

- Hızlı erişim için sidebar

- Görsel ipuçları ve ikonlar

Bu rehber, Git öğrenmeye yeni başlayanlar için temel kavramları açıklarken, deneyimli kullanıcılar için de referans olarak kullanılabilecek kapsamlı bir kaynak sunmaktadır.

|  |  |  |  |
| --- | --- | --- | --- |
| **Hafta** | **Konu / Modül** | **Uygulama** | **Çıktı** |
| **1** | Program Tanıtımı, Temel Bilgisayar Bilimleri (BTD01-i) | İşletim sistemi, dosya yönetimi | Dosya klasör yapısı, temel araç kullanımı |
| **2** | Temel Yazılım Rehberi (BTD01-ii) | Kod editörü (VS Code) kurulumu, ilk script | Basit Python/HTML örneği |
| **3** | Temel Web Uygulaması Geliştirme (BTD01-iii) | HTML & CSS ile basit sayfa | Kişisel tanıtım web sayfası |
| **4** | Doküman Versiyon Takibi (BTD01-iv) | Git/GitHub kullanımı | İlk commit & repo |