Backend Development Nedir?

- Backend development, sunucu tarafı mantığının geliştirilmesi anlamına gelir; bu, veritabanları, sunucular ve uygulamalarla ilgili kodları içerir.
- Frontend ve backend teknolojileri arasındaki farkları anlamak önemlidir; frontend, istemci tarafında çalışırken, backend sunucu tarafında çalışır.

Backend Bileşenleri

- Backend teknolojileri, sunucular, veritabanları, web API'leri, programlama dilleri, çerçeveler ve çalışma zamanlarını içerir.
- Sunucular, istemcilere işlevsellik sağlamak için iletişim kuran donanım ve yazılımlardır; veritabanı sunucuları, verileri depolayan ve sunan makineler olarak tanımlanır.

Önemli Backend Teknolojileri

- Backend programlama dilleri arasında JavaScript, PHP, Python, Ruby, Java ve C# bulunur.
- Çerçeveler, kod için yapı sağlar; örneğin, Django (Python), Ruby on Rails (Ruby) ve Express.js (JavaScript) gibi popüler çerçeveler vardır.

Performans ve Ölçeklenebilirlik

- Backend, uygulamanın performansını ve ölçeklenebilirliğini sağlamakla sorumludur; ölçeklenebilirlik, uygulamanın kullanıcı sayısı ve veri yükü arttıkça performansını korumasını ifade eder.
- Güvenlik, kimlik doğrulama ve kötü amaçlı yazılım saldırılarını önleme gibi diğer backend sorumlulukları da vardır.

Arka Uç Geliştirmenin Ana Bileşenleri:

1. **Sunucu:** Bir ağa bağlı istemcilere kaynak, veri, hizmet veya program sağlayan donanım veya yazılım.
2. **Veritabanı:** Kolayca erişilebilen, yönetilebilen ve güncellenebilen yapılandırılmış bir veri koleksiyonu.
    - **SQL Veritabanları:** MySQL, PostgreSQL ve SQLite gibi ilişkisel veritabanları.
    - **NoSQL Veritabanları:** MongoDB, Cassandra ve Redis gibi ilişkisel olmayan veritabanları.
3. **Sunucu Tarafı Dilleri:** Arka uç mantığını oluşturmak için kullanılan programlama dilleri.

- **Node.js:** Chrome'un V8 JavaScript motoru üzerine inşa edilmiş JavaScript çalışma zamanı.
- **Python:** Okunabilirliği ve verimliliği ile bilinen yüksek seviyeli bir programlama dili.
- **Ruby:** Basitliğe odaklanan dinamik, açık kaynaklı bir programlama dili.
- **Java:** Büyük ölçekli uygulamalar için kullanılan çok yönlü ve güçlü bir programlama dili.

4. **Çerçeveler ve Kütüphaneler:**
    - **Express:** Minimal ve esnek bir Node.js web uygulama çerçevesi.
    - **Django:** Hızlı geliştirmeyi teşvik eden yüksek seviyeli bir Python web çerçevesi.
    - **Ruby on Rails:** Ruby ile yazılmış bir sunucu tarafı web uygulama çerçevesi.
    - **Spring:** Kurumsal Java geliştirme için kapsamlı bir çerçeve.

Sağlam, Ölçeklenebilir Bir Arka Uçun Önemi:

Sağlam bir arka uç, veri güvenliğini, performansı, ölçeklenebilirliği ve sorunsuz uygulama mantığını garanti eder. Kullanıcı etkileşimlerini destekler, iş mantığını yönetir ve ön uç sistemleriyle entegrasyon sağlar.

Ön Uç Geliştirme

Ön uç geliştirme, istemci tarafı geliştirme olarak da bilinir, kullanıcıların doğrudan etkileşimde bulunduğu bir web sitesi veya uygulamanın parçasını oluşturmayı içerir. Bu, kullanıcının görsel olarak ve etkileşimler yoluyla deneyimlediği her şeyi kapsar.

Ön Uç Geliştirmenin Temel Bileşenleri:

1. **HTML (Hiper Metin İşaretleme Dili):** Web sayfalarının yapısı, başlıklar, paragraflar, resimler ve bağlantılar gibi öğeleri tanımlar.
2. **CSS (Kaskad Stil Sayfaları):** Web sayfalarının stilini, düzenini, renklerini, yazı tiplerini ve duyarlılığını içerir.
3. **JavaScript:** Web sayfalarının davranışını, formlar, animasyonlar ve dinamik içerik güncellemeleri gibi etkileşimli özellikleri sağlar.
4. **Frameworkler ve Kütüphaneler:**
    - **React:** Kullanıcı arayüzleri oluşturmak için bir JavaScript kütüphanesi.
    - **Angular:** Web uygulamaları oluşturmak için TypeScript tabanlı bir framework.
    - **Vue.js:** Kullanıcı arayüzleri oluşturmak için ilerici bir JavaScript framework'ü.

Sonuç:

Arka uç ve ön uç geliştirme, işlevsel ve kullanıcı dostu web uygulamaları oluşturmak için esastır. Ön uç geliştirme, etkileşimli kullanıcı arayüzleri oluşturmak için HTML, CSS ve JavaScript gibi teknolojileri içerir. Arka uç geliştirme, Node.js, Python ve Java gibi dilleri kullanarak sunucuları, veritabanlarını ve uygulama mantığını yönetir. Sağlam bir arka uç, veri güvenliği, performans ve ölçeklenebilirlik sağlar, böylece ön uç ile sorunsuz entegrasyonu destekleyerek akıcı bir kullanıcı deneyimi sunar. Her ikisi de herhangi bir web uygulamasının başarısı için kritik öneme sahiptir.

Tam yığın uygulaması:

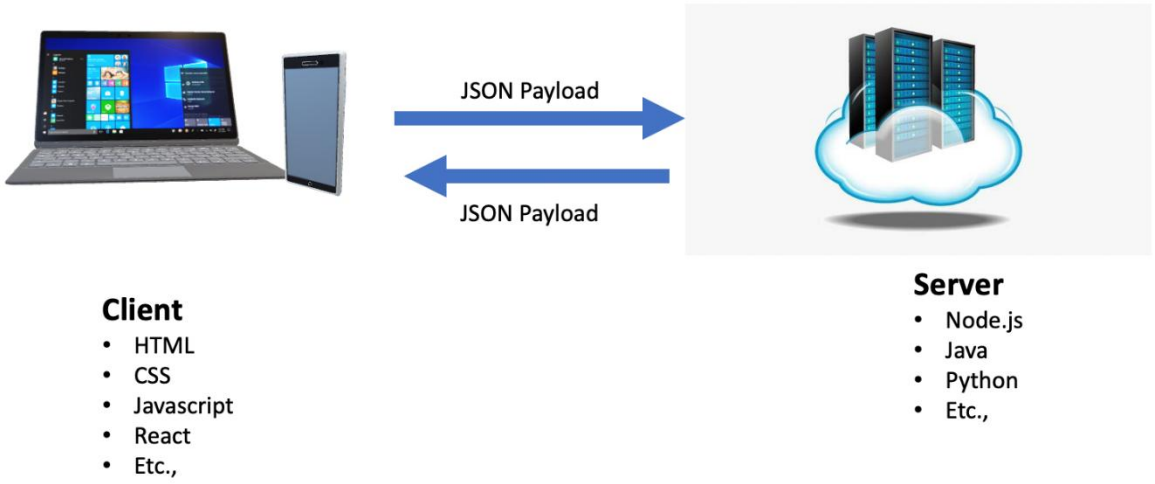
Tam yığın uygulaması aşağıdaki bileşenleri içerir:

İstemci Tarafı:

Kullanıcıya yönelik web sitesi ve mobil uygulama.

Sunucu Tarafı:

İstemci tarafından gelen her türlü isteği işler ve yanıtları istemciye geri gönderir. Günümüzde, bulut web sunucusunu, uygulama sunucusunu ve veritabanını barındırır.



Açık Kaynak ve Çoklu Platform:

JavaScript, HTML sayfalarının istemci tarafında doğrulanması için ideal bir seçimdir çünkü çok yönlüdür ve çoklu platform uyumluluğuna sahiptir. Kullanım kolaylığının fark edilmesiyle, JavaScript sunucu tarafı kodlaması için de uyarlanmıştır ve bu da Node.js'i ortaya çıkarmıştır. Node.js, bir çalışma zamanı ortamı olarak işlev görür ve ayrıca kullanım için özel lisans gerektirmez. Ayrıca, açık kaynak doğası sayesinde Node.js ile kullanılmak üzere birçok paket ve kütüphane geliştirilmiştir. Dahası, Node.js kodu, Linux, Windows ve Mac OSX gibi çeşitli işletim sistemlerinde sorunsuz bir şekilde çalışabilir.

V8 Motoru:

Yazdığınız her bir kod parçası, işlenmesi ve makine tarafından okunabilir bir forma dönüştürülmesi gereken bir süreçten geçmelidir. Node.js alanında, JavaScript kodu Google V8 motoru kullanılarak yürütülür. Yüksek performansıyla tanınan V8, Google tarafından geliştirilen açık kaynak bir motordur ve tüm Google Chrome tarayıcılarına entegre edilmiştir. Microsoft Edge gibi modern tarayıcılar JavaScript için V8 motorunu kullanırken, Node.js sunucu tarafında V8'i kullanır.

Olay Tabanlı, Asenkron, Engellemeyen, Tek İşlemci:

Sunucu süreçleri "tek iş parçacıklı" veya "çoklu iş parçacıklı" olarak kategorize edilebilir. Tek iş parçacıklı bir ortamda, yalnızca bir komut belirli bir anda işlenirken, çoklu iş parçacıklı bir ortamda birden fazla komut aynı anda işlenebilir. Tek iş parçacıklı olmasına rağmen, Node.js asenkron ve engellemeyen doğası sayesinde performans açısından öne çıkar. Bu, bir süreç yürütülürken programın tamamlanmasını beklemesi gerekmediği anlamına gelir. Node.js olay tabanlıdır; bu, ağdan okuma veya bir veritabanına veya dosya sistemine erişim gibi bir girdi/çıktı (I/O) işlemi gerçekleştirdiğinde bir olayın tetiklendiği anlamına gelir. İş parçacığını engellemek ve yanıtı beklerken işlemci zamanını tüketmek yerine, Node.js yanıt alındığında veya ilgili olay gerçekleştiğinde işlemleri yeniden başlatır. Bu engellemeyen davranış, sunucunun yanıt verebilir durumda kalmasını ve çoklu görevleri eşzamanlı olarak yönetmesini sağlar, tıpkı çoklu iş parçacıklı bir ortam gibi.

JSON Yükü

JSON, "anahtar-değer çiftleri" olarak biçimlendirilmiş JavaScript Nesne Notasyonu'nu ifade eder. Yük, istemci ile sunucu arasında iletilen veriyi temsil eder. İstemci sunucuya veri göndermesi gerektiğinde, bunu aşağıdaki örnekte gösterildiği gibi bir JSON nesnesi şeklinde yapar:

Express Framework

Node.js, bir sunucu oluşturmak için paketler sağlarken, Express framework'ü API'ler ve uç noktalar oluşturma sürecini basitleştirir. Bir API uç noktası, istemciden sunucuya yapılan bir isteğin belirli bir giriş noktasıdır.