3. Kullanilan Teknolojiler ve Metotlar

3.1. Kullanılan Teknolojiler

3.1.1. Ön Uç(Front-end) Teknolojileri

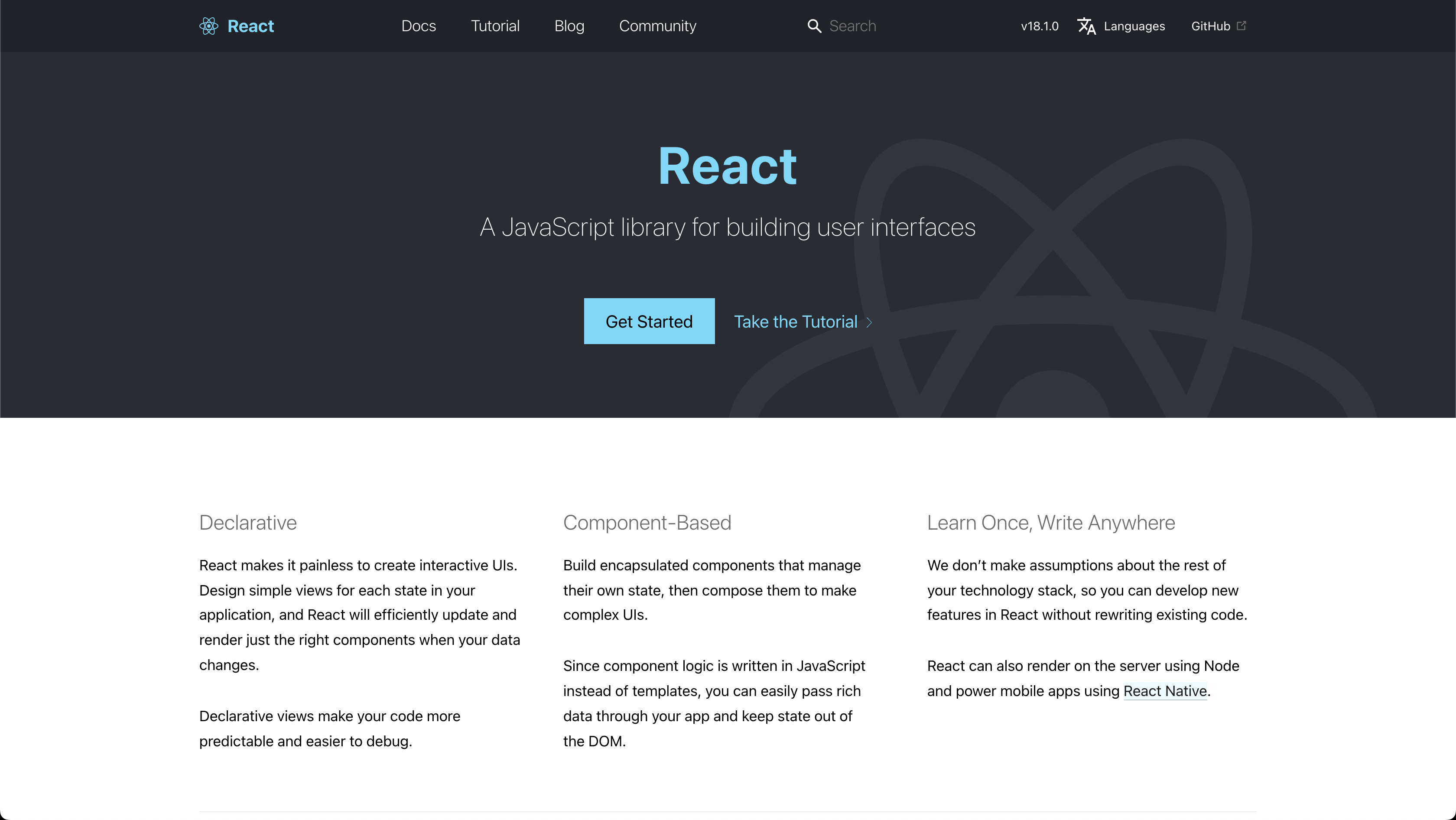
**HTML**

Web sayfalarını oluştururken kullanılan standart bir metin işaretleme dilidir. Son sürümü HTML5’tir ve açılımı “Hyper Text Markup Language”’dir. Temel amacı yazı, görüntü, video gibi değişik verileri ve bunları içeren sayfaları birbirine bağlamak, söz konusu sayfaların web tarayıcıları tarafından düzgün olarak görüntülenmesi için gerekli kuralları ve standartları belirlemektir bundan dolayı aynı HTML, CSS kodu farklı tarayıcılarda farklı sonuç almamız olası bir durumdur. JavaScript ve CSS ile beraber kullanıldığında görsel ve dinamik web siteleri yaratılabilir.

**CSS**

CSS, bir web sayfasını biçimlendirmek ve stillendirmek için kullandığımız dildir. Açılımı Cascading Style Sheets’dir. HTML gibi bir biçimlendirme dilinde yazılmış bir belgenin sunumunu açıklamak için kullanılan bir stil sayfası dilidir. CSS, HTML ve JavaScript'in yanı sıra World Wide Web'in temel taşı teknolojisidir. CSS, düzen, renkler ve yazı tipleri dahil olmak üzere sunum ve içeriğin ayrılmasını sağlamak için tasarlanmıştır. Bu ayrım, içerik erişilebilirliğini iyileştirebilir; sunum özelliklerinin belirlenmesinde daha fazla esneklik ve kontrol sağlamaktadır. Projemizin arayüzünü stillendirmek ve kullanıcı deneyimini artırmakta yoğunlukla kullandığımız teknolojidir.

**React.js**React bir JavaScript kütüphanesi olup web uygulamaları için hızlı ve interaktif kullanıcı arayüzleri oluşturmada kullanılır. Açık kaynak kodlu, bileşen bazlı ve front-end bir kütüphane olan React, sadece uygulamanın görünüm katmanından sorumludur. Projemizin arayüzünü oluşturmamızda ve kullanıcının tamamen etkileşim halinde olduğu için projemizde React.js’i uygun gördük. Tercih etmemizin birden çok nedeni bulunmakta, bunlardan en önemlisi takımdaki herkesin React.js Framework’unu bilmesi diğer ikinci nedeni ise React.js ile yapılmış bol miktarda kütüphane bulunmasıdır. Bizde projede React.js ile yazılmış ve React.js’in kütüphanelerden birçok kütüphanesini kullandık. Ayrıca React.js’i yapan şirket Facebook olması bizim bu Framework’u seçmemizde bir etken olmuştur.



**Reactstrap**

Reactstrap, tepkiler için bir bileşen kütüphanesidir. Esneklik ve yerleşik doğrulamalar sağlayan bağımsız bileşenleriyle UI oluşturmayı kolaylaştıran yerleşik Bootstrap bileşenleri sağlar. Reactstrap, Bootstrap'a benzer, ancak kendi kendine yeten bileşenlere sahiptir. Bootstrap 4'ü kullanmak ve desteklemek kolaydır.

**MetarialUI**

Material-UI, Google'ın Materyal Tasarımını uygulayan React bileşenlerini içeren açık kaynaklı bir projedir. React'in halka açılmasından kısa bir süre sonra 2014'te başladı ve o zamandan beri popülaritesi arttı.

**Socket.io**

Socket.IO gerçek zamanlı ve karşılıklı veri alışverişine ihtiyaç duyan uygulamalar geliştirmek için kullanılan Node.js modülüdür. Socket.IO karşılıklı veri alışverişi yapmak için tarayıcı desteğine göre WebSocket, AJAX, Flash Socket gibi çeşitli yöntemleri kullanarak eski tarayıcılarda da karşılıklı veri alışverişine imkan verir.

Websocket Nedir ?

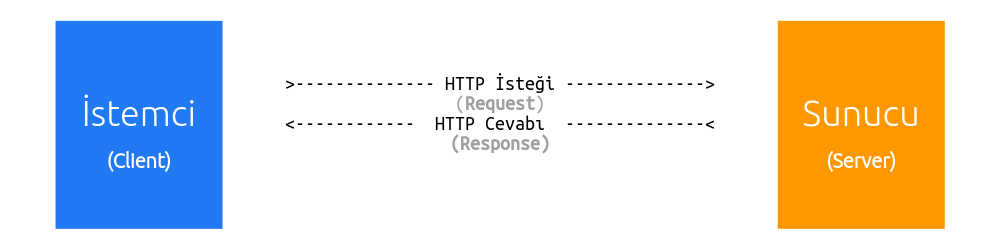
Websocket IETF tarafından RFC 6455 ile kuralları belirlenmiş sunucu ve istemci arasında karşılıklı, çift taraflı veya terminolojideki adıyla full duplex veri alışverişine imkan veren protokoldür.

AJAX Nedir ?

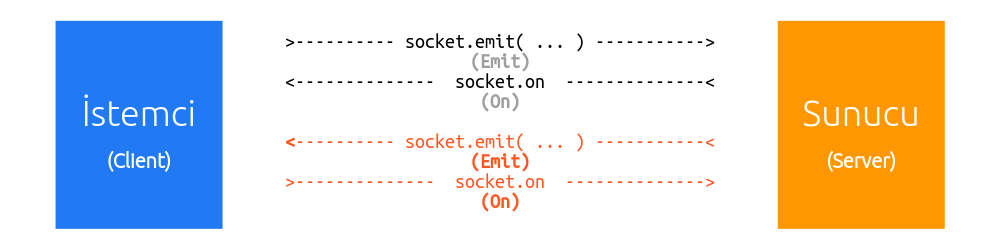
Ajax, asenkron web uygulamaları oluşturmak için istemci tarafında çeşitli web teknolojilerini kullanan bir dizi web geliştirme tekniğidir. Ajax ile web uygulamaları, mevcut sayfanın görüntülenmesine ve davranışına müdahale etmeden bir sunucudan asenkron olarak veri gönderip alabilir.

**Socket.io Client**

Socket.io Client ise Socket.io’un tarayıcılarımızda socket bağlantısı yapmak ve server ile haberleşmek için browser’da çalışan socket.io kütüphanesidir.

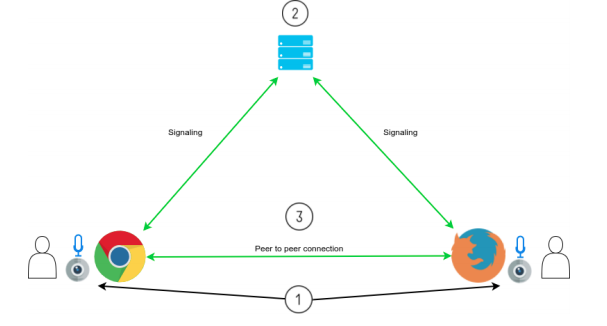


HTTP Çalışma Sistemi

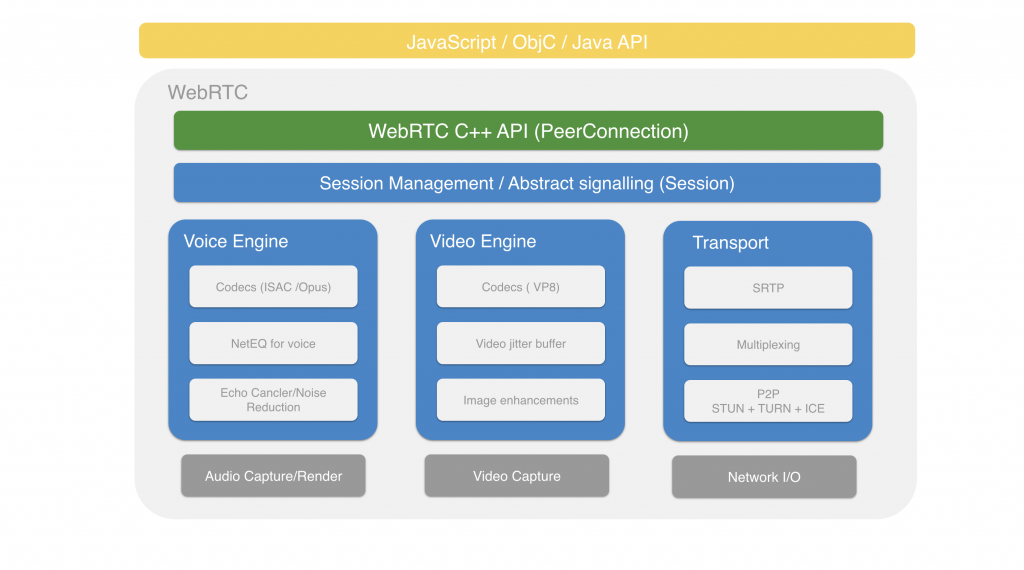


Socket.io Çalışma Sistemi

**webRTC**

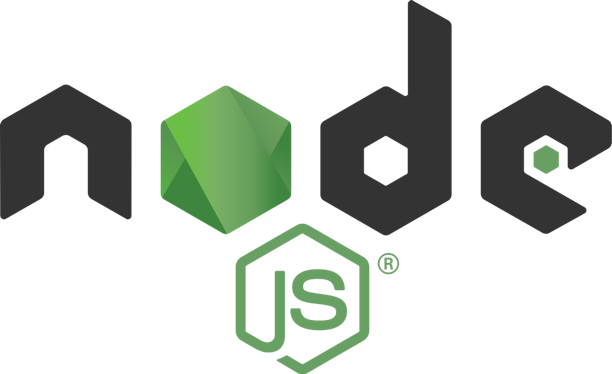


WebRTC (Web Gerçek Zamanlı İletişim), tarayıcılar arasında eşler arası çift yönlü gerçek zamanlı iletişim için geliştirilmiş bir dizi teknolojidir. Adından da anlaşılacağı gibi Web ile uyumludur ve W3C'de bir standarttır WebRTC'nin önemli özelliklerinden biri de NAT adreslerinin arkasından bile çalışmasıdır. WebRTC ( Web Gerçek Zamanlı İletişimi ), web tarayıcılarına ve mobil uygulamalara basit uygulama geliştirme arayüzü (API'ler) aracılığıyla gerçek zamanlı iletişim (RTC) sağlayan ücretsiz, açık kaynaklı bir projedir. Direkt olarak eşler arası iletişime izin vermesi ile, eklenti yükleme veya uygulama indirme ihtiyacını ortadan kaldırarak, ses ve video iletişiminin web sayfalarında kolaylıkla kullanılmasını sağlar. webRTC API’si 3 ana başlık altında toplayabiliriz bunlar: MediaStream, PeerConnection, DataChannel



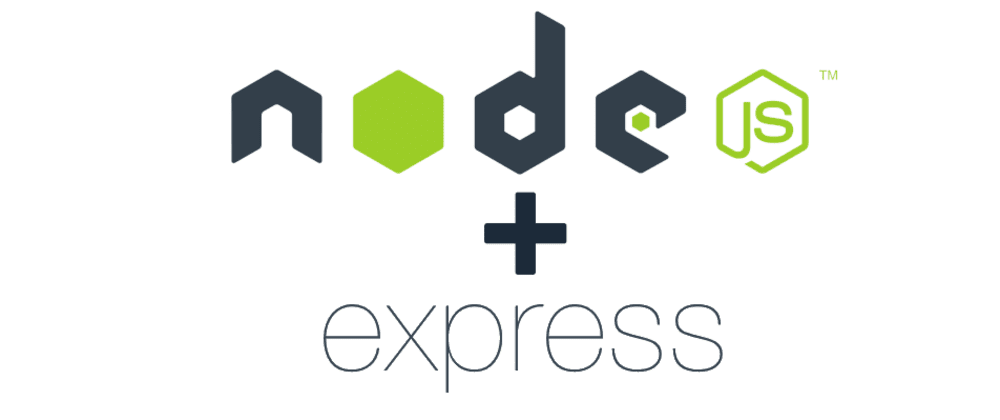
**Node.js**

Node.js, V8 motorunda çalışan ve JavaScript kodunu bir web tarayıcısının dışında yürüten açık kaynaklı, platformlar arası, arka uç bir JavaScript çalışma zamanı ortamıdır. Node.js, geliştiricilerin komut satırı araçları yazmak ve sunucu tarafı komut dosyası oluşturmak için JavaScript'i kullanmalarına olanak tanır; sayfa kullanıcının web tarayıcısına gönderilmeden önce dinamik web sayfası içeriği oluşturmak için sunucu tarafında komut dosyaları çalıştırır. Sonuç olarak, Node.js, sunucu tarafı ve istemci tarafı komut dosyaları için farklı diller yerine web uygulaması geliştirmeyi tek bir programlama dili etrafında birleştiren bir "JavaScript her yerde"paradigmasını temsil eder. Node.js, eşzamansız G/Ç yapabilen, olaya dayalı bir mimariye sahiptir. Bu tasarım seçenekleri, birçok giriş/çıkış işlemine sahip web uygulamalarında ve gerçek zamanlı Web uygulamalarında (örneğin, gerçek zamanlı iletişim programları ve tarayıcı oyunları) verim ve ölçeklenebilirliği optimize etmeyi amaçlar.

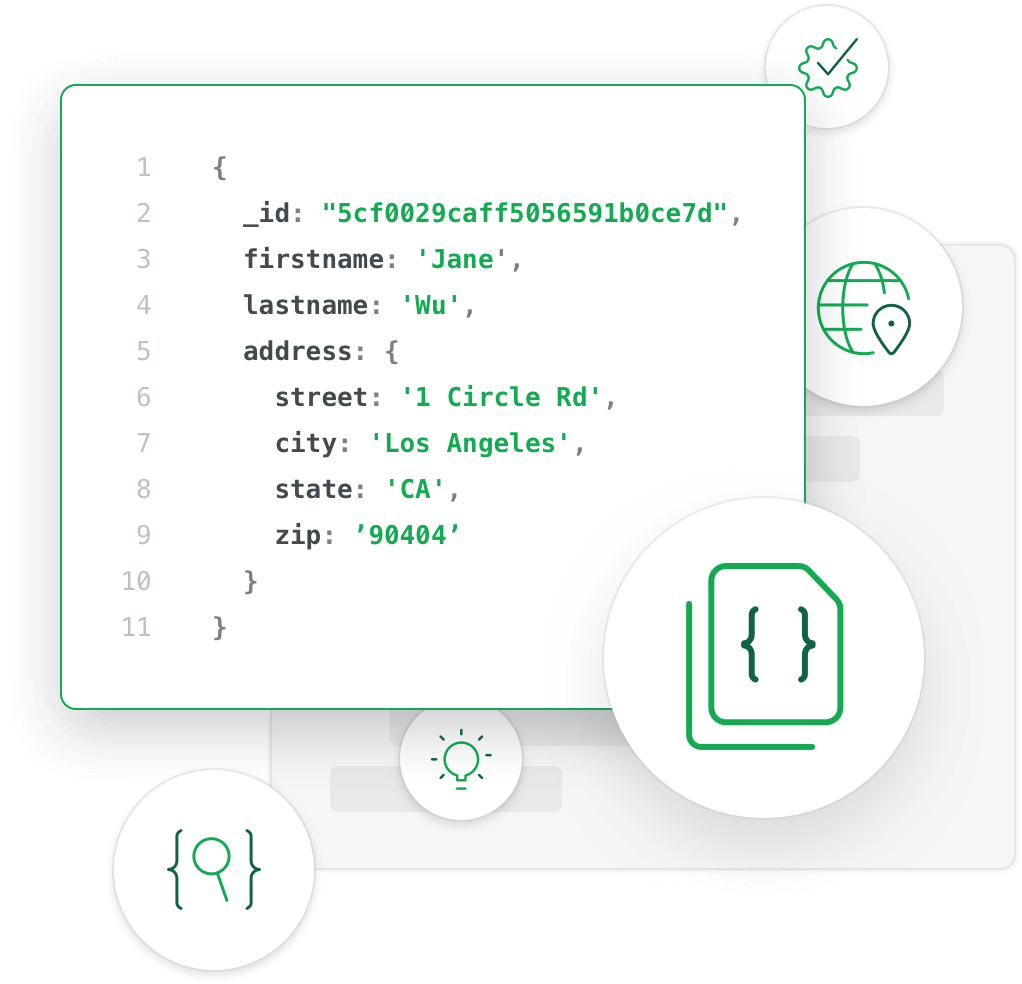


**Express.js**

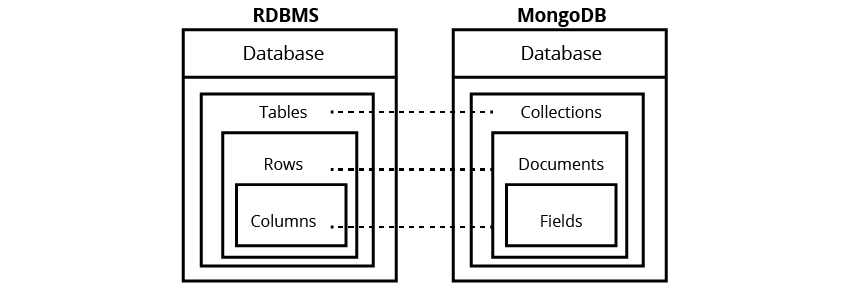
Express.js veya basitçe Express, MIT Lisansı altında ücretsiz ve açık kaynaklı yazılım olarak yayınlanan Node.js için bir arka uç web uygulama çerçevesidir. Web uygulamaları ve API'ler oluşturmak için tasarlanmıştır. Node.js için de facto standart sunucu framework’u olarak adlandırılmıştır.



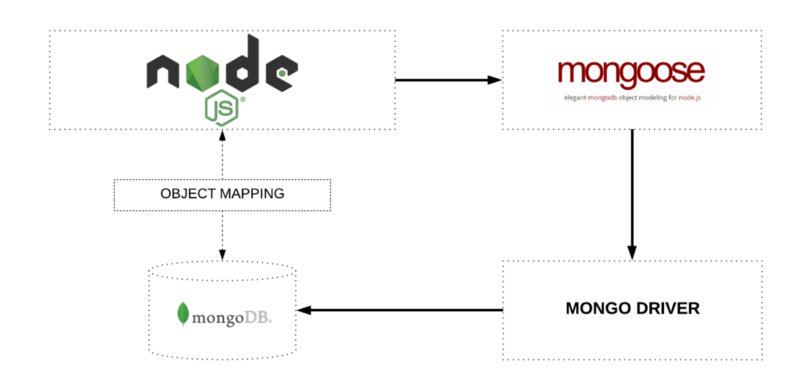
**MongoDB**



MongoDB, yüksek hacimli veri depolama için kullanılan belge odaklı bir NoSQL veritabanıdır. MongoDB, geleneksel ilişkisel veritabanlarında olduğu gibi tabloları ve satırları kullanmak yerine koleksiyonları ve belgeleri kullanır. Belgeler, MongoDB'deki temel veri birimi olan anahtar/değer çiftlerinden oluşur. Collection’lar, ilişkisel veritabanı tablolarının eşdeğeri olan belge ve işlev kümelerini içerir. MongoDB, 2000'lerin ortalarında ortaya çıkan bir veritabanıdır. JSON benzeri depolama için destek sağlayan ilişkisel olmayan bir belge veritabanıdır. MongoDB veritabanı, yapılandırılmamış verileri depolamanıza olanak tanıyan esnek bir veri modeline sahiptir ve zengin ve sezgisel API'ler ile tam indeksleme desteği ve replikasyon sağlar. Her veritabanı, sırayla belgeleri içeren koleksiyonları içerir. Her belge, değişen sayıda alanla farklı olabilir. Her belgenin boyutu ve içeriği birbirinden farklı olabilir. Belge yapısı, geliştiricilerin sınıflarını ve nesnelerini ilgili programlama dillerinde nasıl oluşturdukları ile daha uyumludur. Geliştiriciler genellikle sınıflarının satır ve sütun olmadığını, anahtar/değer çiftleriyle net bir yapıya sahip olduğunu söyleyecektir. Satırların (veya MongoDB'de çağrıldığı şekliyle belgelerin) önceden tanımlanmış bir şemaya sahip olması gerekmez. Bunun yerine, alanlar anında oluşturulabilir. MongoDB'de bulunan veri modeli, hiyerarşik ilişkileri temsil etmenize, dizileri ve diğer daha karmaşık yapıları daha kolay saklamanıza olanak tanır. MongoDB ortamları çok rahat bir şekilde ölçeklenebilir.



**Mongoose**Mongoose, MongoDB ve Node.js için bir Nesne Veri Modelleme (ODM) kütüphanesidir. Veriler arasındaki ilişkileri yönetir, şema doğrulaması sağlar ve koddaki nesneler ve bu nesnelerin MongoDB'deki temsili arasında çeviri yapmak için kullanılır.



MongoDB, şemasız bir NoSQL belge veritabanıdır. Bu, JSON belgelerini içinde saklayabileceğiniz anlamına gelir ve bu belgelerin yapısı, SQL veritabanları gibi zorunlu olmadığı için değişebilir. Bu, uygulama geliştirmeyi hızlandırdığı ve dağıtımların karmaşıklığını azalttığı için NoSQL kullanmanın avantajlarından biridir.

En başından itibaren bizi yarı katı bir şemaya zorlayan Mongoose gibi ODM'lerimiz var. Mongoose ile, uygulama kodunuzda MongoDB veritabanınızdaki bir collection’la eşleşen bir Schema nesnesi tanımlarsınız. Schema nesnesi, koleksiyonunuzdaki belgelerin yapısını tanımlar. Ardından, şemadan bir Model nesnesi oluşturmanız gerekir. Model, collection’la etkileşim kurmak için kullanılır.

HTML

CSS

SASS

React.js

Socket.io Client

webRTC

Reactstrap

MetarialUI

Jest

3.1.2. Arka Uc Teknolojileri(Backend)

Backend

Node.js

Express.js

Socket.io

MongoDB

Mongoose

3.2. Metotlar

3.2.1. Algoritmalar

3.2.2. Kütüphane ve paket kullanımları

3.2.3. Uygulama sunum platformları