Introduction to Web Application

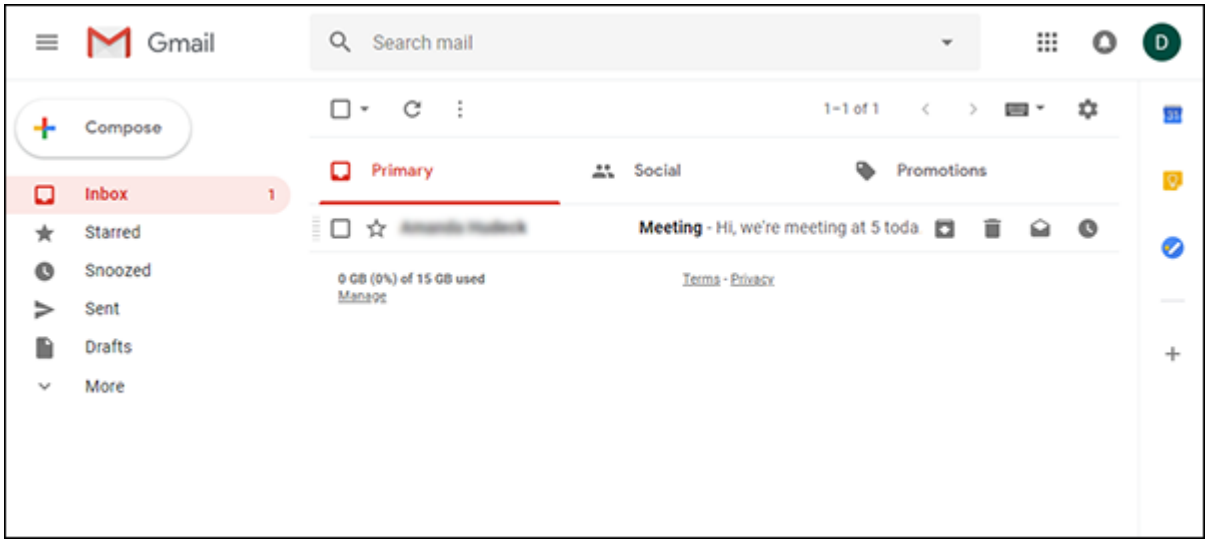
Web (World Wide Web) – internet üzərində mövcud olan, istifadəçilərə istədikləri resurslara çıxış imkanı verən və məlumat mübadiləsini təmin edən bir sistemdir. O, əsasən **HTTP və HTTPS protokolları** vasitəsilə işləyir və veb brauzerlərdən istifadə etməklə resurslara (saytlar, tətbiqlər, məlumat bazaları) daxil olmağa imkan verir.

Veb tətbiqlər – veb brauzerlərdə işləyən interaktiv proqramlardır.
Bu tətbiqlər **müştəri-server arxitekturası** əsasında qurulur.
Onlar iki əsas hissədən ibarətdir:

- **Frontend** (istifadəçi interfeysi – yəni istifadəçinin gördüyü hissə),
- **Backend** (serverdə işləyən hissə – mənbə kodu, verilənlər bazası).

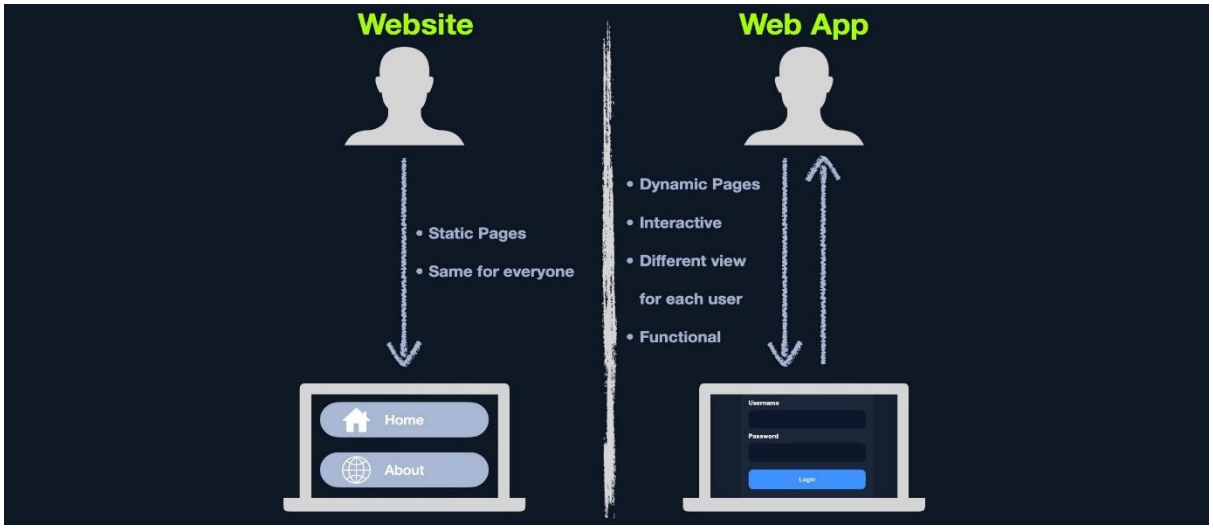
Bu model təşkilatlara tətbiqlərin dizaynı və funksionallığı üzərində **demək olar ki, tam real vaxt nəzarəti** imkanı verir və eyni zamanda onları **dünya üzrə əlçatan** edir. Bir Web tətbiq nümunələri:

- **Gmail (onlayn e-poçt xidməti),**
- **Amazon (onlayn ticarət platforması),**
- **Google Docs (onlayn mətn redaktoru).**



Veb tətbiqlər yalnız **Google** və **Microsoft** kimi nəhəng təminatçılara məxsus deyil, əksinə, istənilən veb tərtibatçı tərəfindən hazırlanıb, geniş istifadə olunan hosting xidmətlərində yerləşdirilə bilər və internetdə istənilən şəxs tərəfindən istifadə oluna bilər. Buna görə də **bu gün internetdə milyonlarla veb tətbiq var** və **milyardlarla istifadəçi** hər gün onlarla qarşılıqlı əlaqə qurur.

Keçmişdə istifadə etdiyimiz veb saytlar **statik** idi və **real vaxtda dəyişdirilə bilmirdi**. Yəni ənənəvi veb saytlar sadəcə müəyyən məlumatı göstərmək üçün hazırlanırdı və bu məlumat istifadəçi ilə qarşılıqlı əlaqə zamanı dəyişmirdi. Saytın məzmununu dəyişmək üçün isə həmin səhifə **developer tərəfindən əl ilə redaktə olunmalı idi**. Bu tip statik səhifələr funksionallığa malik deyildi və real vaxt dəyişiklikləri mümkün olmurdu. Belə saytlar **Web 1.0** adlanır.





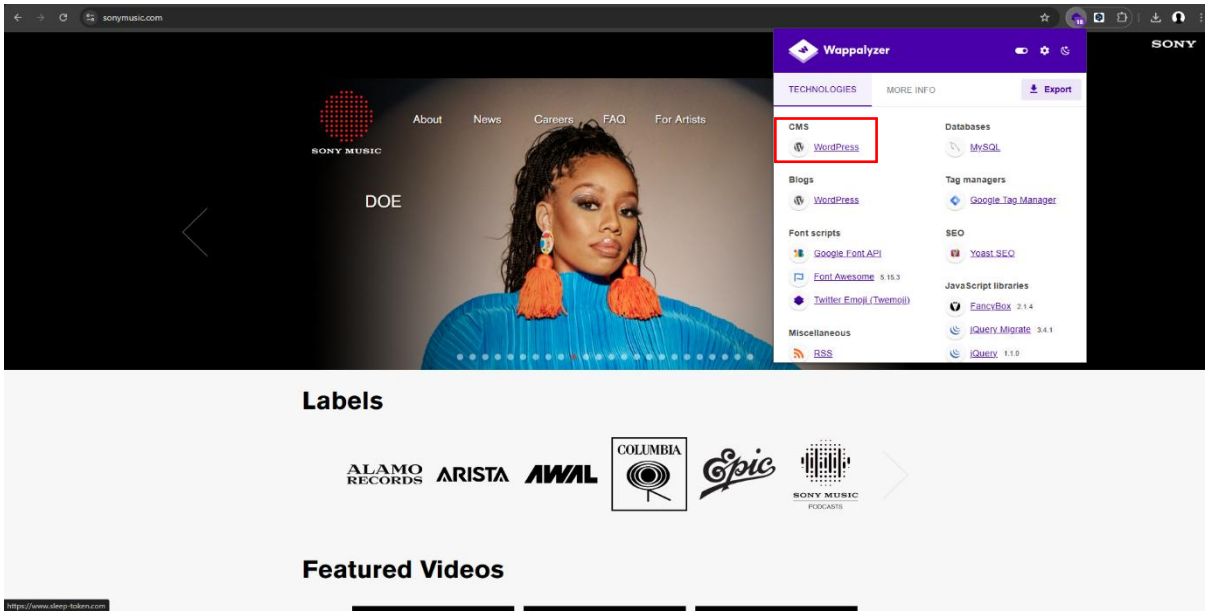
Müasir dövrdə isə əksər saytlar **veb tətbiqlər** üzərində qurulub və **Web 2.0** texnologiyasını təmsil edir. Bu tip saytlar dinamik məzmun təqdim edir və istifadəçinin hərəkətinə uyğun olaraq dəyişir. Ən böyük fərq ondan ibarətdir ki, veb tətbiqlər **tam funksionaldır** və istifadəçi üçün müxtəlif əməliyyatlar yerinə yetirə bilir. Ənənəvi veb saytlar isə bu imkanlara malik deyil.

Veb Tətbiqlər və Yerli Əməliyyat Sistemi Tətbiqləri

Veb tətbiqlər, **yerli əməliyyat sistemi (native OS) tətbiqlərindən fərqli olaraq**, platformadan asılı deyil və **istənilən əməliyyat sistemində brauzer vasitəsilə işləyə** bilir. Bu tətbiqləri istifadəçinin sisteminə quraşdırmaq lazım deyil, çünki onların funksionallığı **uzaq serverdə işləyir** və beləliklə, istifadəçinin **hard diskində yer tutmur**. Veb tətbiqlərin bir digər üstünlüyü isə **versiya vahidliyidir**. Bütün istifadəçilər eyni veb tətbiqin eyni versiyasını istifadə edir və tətbiq **davamlı olaraq yenilənə** bilər. Bu zaman hər bir istifadəçiyə ayrıca yeniləmə göndərmək lazım olmur. Veb tətbiqləri **tək bir serverdə yeniləmək** kifayətdir, fərqli platformalar üçün ayrıca versiyalar yaratmaq tələb olunmur. Bu isə **təmir və dəstək xərclərini əhəmiyyətli dərəcədə azaldır**. Digər tərəfdən, **yerli OS tətbiqləri** bəzi üstünlüklərə malikdir. Əsas üstünlüklər: onların **işləmə sürəti** və **yerli əməliyyat sistemi kitabxanalarından və lokal hardware-dən istifadə imkanları**dır. Yerli tətbiqlər **OS kitabxanalarını tam istifadə etdikləri üçün** daha sürətli yüklənir və qarşılıqlı əlaqə daha effektiv olur. Bundan əlavə, yerli tətbiqlər adətən **daha funksionaldır**, çünki əməliyyat sistemi ilə daha dərin inteqrasiya olunublar və yalnız brauzerin imkanları ilə məhdudlaşmırlar.

Veb Tətbiqlərin distroları

Dünyada bir çox təşkilatlar **açıq mənbəli (open-source) veb tətbiqlər**dən istifadə edir. Bu tətbiqlər **hər bir təşkilatın ehtiyacına uyğun olaraq tənzimlənə bilər**. Ən məşhur açıq mənbəli veb tətbiqlərə **WordPress**, **OpenCart** və **Joomla** daxildir.



Bundan əlavə, **müəllif hüquqları ilə qorunan (closed source) veb tətbiqlər** də mövcuddur. Bu tətbiqlər adətən müəyyən bir təşkilat tərəfindən hazırlanır və ya **başqa təşkilatlara satılır**, ya da **abunə modeli ilə istifadə olunur**. Ən məşhur closed source veb tətbiqlərə **Wix**, **Shopify** və **DotNetNuke** daxildir.

Web Application Layout-u (dizaynı)

Heç bir iki veb tətbiq bir-birinə bənzəmir. Təşkilatlar veb tətbiqləri **müxtəlif məqsədlər və auditoriyalar üçün** yaradır. Veb tətbiqlər **fərqli şəkildə dizayn olunur və proqramlaşdırılır**, arxa plan infrastrukturu isə **bir çox fərqli üsulla quraşdırıla bilər**. Əhəmiyyətlidir ki, veb tətbiqlərin **arxa planda necə işlədiyini, strukturunu, komponentlərini və şirkət infrastrukturu daxilində necə qurula biləcəyini** başa düşək.

Veb tətbiqlərin **layout-u bir neçə fərqli qat**dan ibarətdir və bunları əsasən üç kateqoriyada toplamaq olar:

1. **Veb Tətbiq İnfrastrukturu**
   Bu, veb tətbiqin düzgün işləməsi üçün tələb olunan komponentlərin, məsələn **verilənlər bazasının**, strukturunu izah edir. Çünki veb tətbiq **ayrı serverdə işləyə** bilər, buna görə hansı **database serverinə** ehtiyac olduğunu bilmək vacibdir.
2. **Veb Tətbiq Komponentləri**
   Veb tətbiqi təşkil edən komponentlər onun qarşılıqlı əlaqədə olduğu bütün elementləri əhatə edir. Bu komponentlər üç əsas sahəyə bölünür: **UI/UX**, **Client** və **Server komponentləri**.
3. **Veb Tətbiq Arxitekturası**
   Arxitektura veb tətbiq komponentləri arasındakı bütün **əlaqələri və qarşılıqlı təsirləri** əhatə edir.

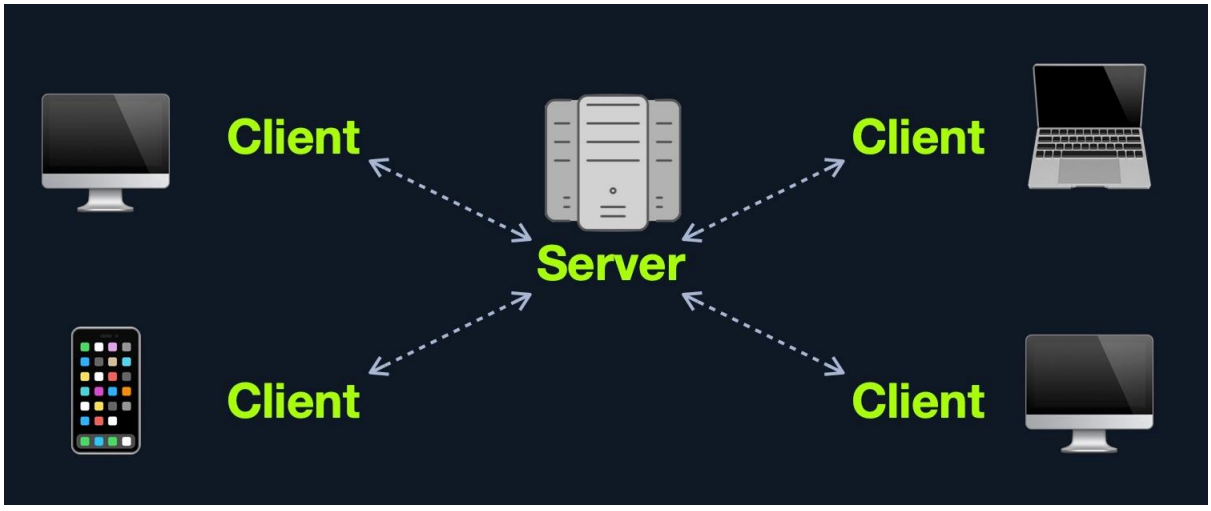
Veb Tətbiq İnfrastrukturu

Veb tətbiqlər müxtəlif **infrastruktur quruluşlarından** istifadə edə bilər. Bu quruluşlar bəzən **modellər** adlandırılır. Ən çox istifadə olunan modellər isə dörd əsas növə bölünür:

1. **Client-Server**
2. **Bir Server (One Server)**
3. **Çox Server – Bir Verilənlər Bazası (Many Servers – One Database)**
4. **Çox Server – Çox Verilənlər Bazası (Many Servers – Many Databases)**

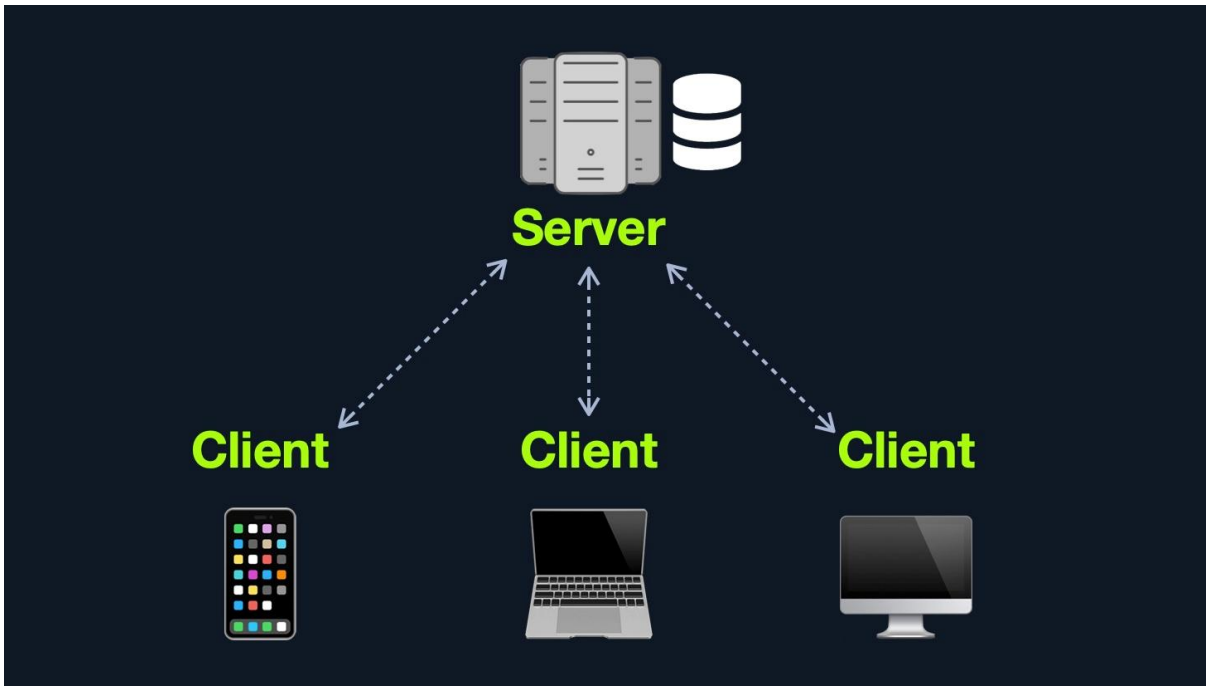
Client-Server modeli

Veb tətbiqlər tez-tez **client-server (müştəri-server) modeli**ni istifadə edir. Bu modeldə **server veb tətbiqi yerləşdirir** və onu **onu əldə etmək istəyən müştərilərə (clients)** təqdim edir. Məsələn, istifadəçi veb tətbiqin URL-ə daxil olduqda (məsələn, https://example.com), server **əsas istifadəçi interfeysini (UI)** göstərir. İstifadəçi düyməyə kliklədikdə və ya müəyyən funksiyanı istədikdə, **brauzer serverə HTTP sorğu göndərir**. Server bu sorğunu **təhlil edir** və tələb olunan əməliyyatları yerinə yetirir (məsələn, istifadəçinin daxil olması, məhsulun səbətə əlavə olunması, başqa səhifəyə keçid və s.). Server tələb olunan məlumatları əldə etdikdən sonra nəticəni **müştərinin brauzerinə göndərir** və insan oxuya biləcəyi şəkildə göstərir.



Bir Server (One Server) modeli

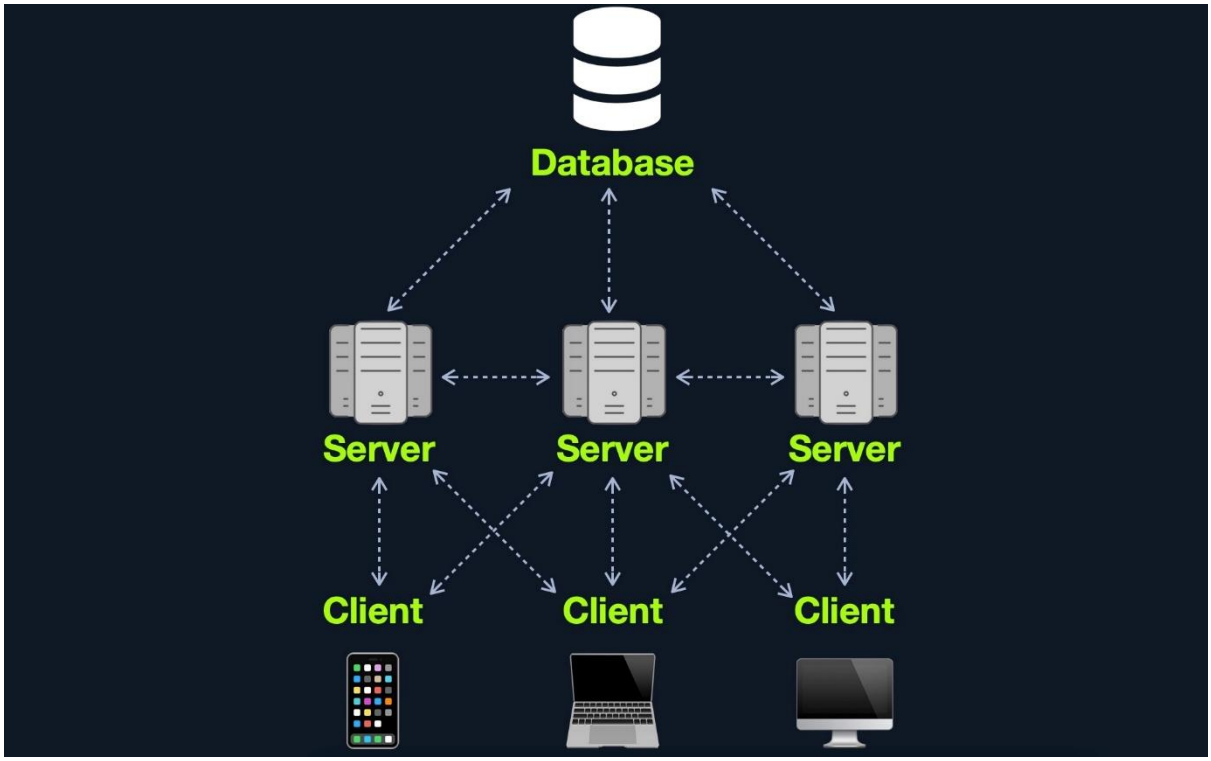
Bu arxitekturda **bütün veb tətbiq və ya bir neçə veb tətbiq** və onların komponentləri, o cümlədən **verilənlər bazası**, **tək bir serverdə yerləşdirilir**. Bu dizayn **sadə və tətbiqi asandır**, amma eyni zamanda **ən riskli dizayn** sayılır. Əgər bu serverdə yerləşən hər hansı bir veb tətbiq **hücuma məruz qalarsa**, bütün veb tətbiqlərin **məlumatları təhlükə altına düşür**. Bu dizayn **"bütün yumurtaları bir səbətdə saxlamaq"** kimi qiymətləndirilir, çünki serverdəki hər hansı bir zəiflik bütün serveri riskə atır. Bundan əlavə, əgər server **hər hansı səbəbdən işləməzsə**, bütün yerləşdirilmiş veb tətbiqlər **tamamilə əlçatmaz olur** və problem həll edilənə qədər istifadəçilər onlara daxil ola bilməz.



Çox Server – Bir Verilənlər Bazası (Many Servers – One Database)

Bu modeldə **verilənlər bazası ayrı bir database serverdə yerləşdirilir**, veb tətbiqlərin host edildiyi server isə bu database serverə **məlumat saxlamaq və oxumaq üçün daxil olur**. Bu, **çox server – bir database** və ya **bir server – bir database** modeli kimi qəbul edilə bilər, əsas odur ki, **verilənlər bazası öz serverində yerləşir**.

Bu model bir neçə veb tətbiqin **eyni verilənlər bazasına çıxışını təmin edir**, beləliklə onlar məlumatları **sinxronizasiya etmədən** istifadə edə bilirlər. Veb tətbiqlər ya **əsas tətbiqin nüsxələri (primary/backup)** ola bilər, ya da **müstəqil veb tətbiqlər** ola bilər ki, ortaq məlumatları paylaşırlar.

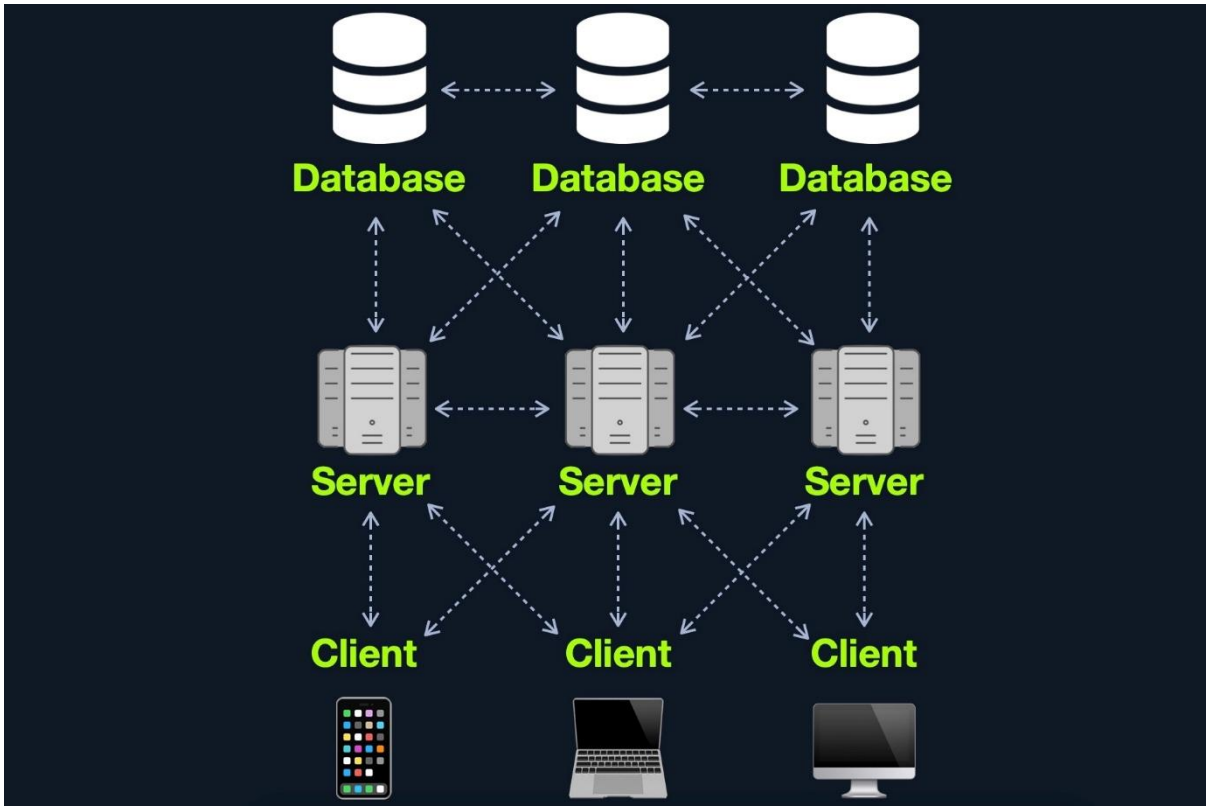


Bu modelin əsas üstünlüyü, **təhlükəsizlik baxımından**, **segmentasiyadır**: veb tətbiqin əsas komponentləri **ayrı-ayrı yerləşdirilir və host edilir**. Əgər bir veb server **hücuma məruz qalarsa**, digər serverlər birbaşa təsirlənmir. Eyni şəkildə, əgər **verilənlər bazası kompromit edilərsə** (məsələn, SQL injection zəifliyi vasitəsilə), veb tətbiqin özü **birbaşa təsirlənmir**.

Çox Server – Çox Verilənlər Bazası (Many Servers – Many Databases)

Bu model, **Çox Server – Bir Verilənlər Bazası** modelinin təkmilləşdirilmiş versiyasıdır. Burada hər veb tətbiqin məlumatları **ayrı verilənlər bazasında** saxlanılır. Veb tətbiq yalnız **özəl məlumatlara** və **digər veb tətbiqlərlə paylaşılmış ümumi məlumatlara** çıxış əldə edə bilər. Həmçinin, hər veb tətbiqin verilənlər bazası **ayrı database serverdə** də yerləşdirilə bilər.

Bu dizayn **redundans (ehtiyat nüsxə) məqsədilə** də geniş istifadə olunur: əgər hər hansı bir veb server və ya verilənlər bazası işləməzsə, **backup** sistemi onun yerinə keçərək fasiləni minimuma endirir.



Veb Tətbiq Komponentləri

Hər veb tətbiq fərqli komponentlərə sahib ola bilər, amma əsasən bütün komponentlər aşağıdakılara bölünür:

1) **Client (Müştəri)** – İstifadəçi veb tətbiqə brauzer vasitəsilə daxil olur.
2) **Server** – Veb tətbiqin işlədiyi arxa plan maşınıdır.
   a) **Webserver**
   b) **Database**
3) **Services (Microservices)**
   a) **3rd Party Integrations**
   b) **Web App Integrations**

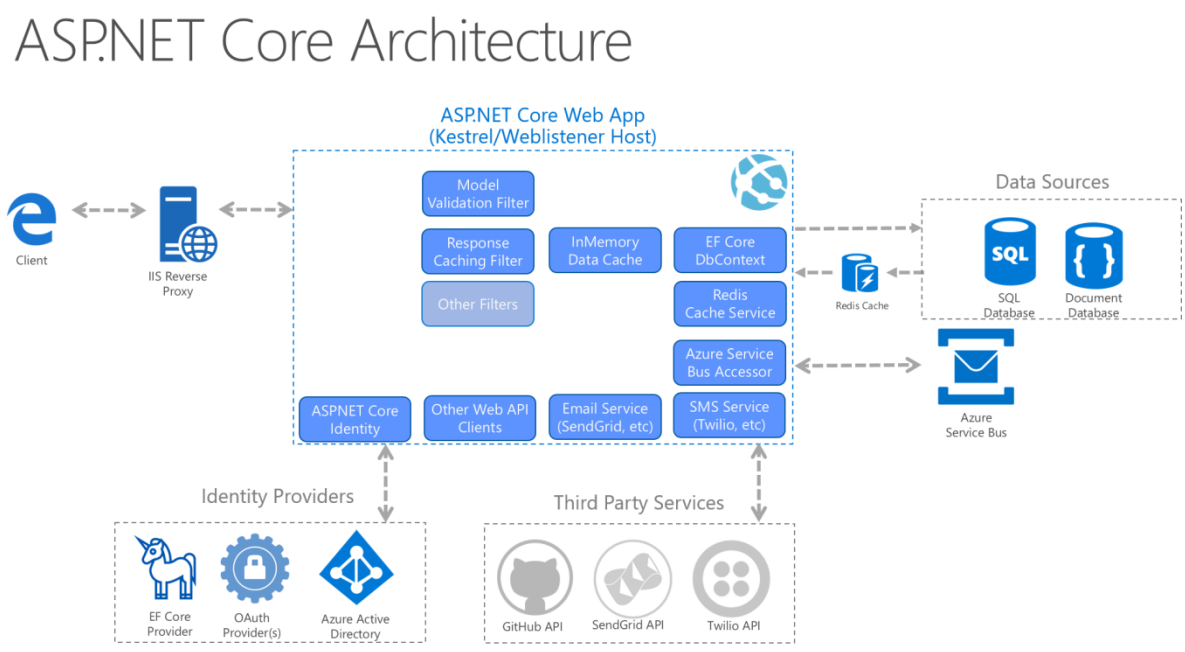
Sadə desək, **veb tətbiq bir neçə "hissədən" ibarətdir**, hər hissə öz rolunu oynayır: istifadəçi ilə əlaqə qurmaq, məlumatları saxlamaq, funksiyaları icra etmək və digər sistemlərlə inteqrasiya olmaq.

Veb Tətbiq Arxitekturası

Veb tətbiqin komponentləri **üç qatlı struktura (Three Tier Architecture)** bölünür:

1. **Presentation Layer (Təqdimat Qatı)**
   - UI komponentlərindən ibarətdir və **istifadəçi ilə tətbiq arasında əlaqəni təmin edir**.
   - Müştəri bu qatla **brauzer vasitəsilə** əlaqə qurur və nəticələr **HTML, JavaScript və CSS** şəklində göstərilir.
2. **Application Layer (Tətbiq Qatı)**
   - Bu qat **bütün müştəri sorğularını düzgün emal edir**.
   - Burada **səlahiyyətlər, icazələr və müştəriyə göndərilən məlumatlar** yoxlanılır.
3. **Data Layer (Verilənlər Qatı)**
   - Application Layer ilə sıx işləyir və **tələb olunan məlumatın harada saxlandığını və necə əldə ediləcəyini müəyyənləşdirir**.

Sadə desək, **Presentation Layer = istifadəçi interfeysi**, **Application Layer = əməliyyatların idarə olunması**, **Data Layer = məlumatların saxlanması və çıxarılması** rolunu oynayır.



Microservices (Mikroxidmətlər)

Microservices, veb tətbiqin **müstəqil komponentləri** kimi düşünülə bilər və əksər hallarda **yalnız bir tapşırıq üçün proqramlaşdırılır**.

Məsələn: Onlayn mağaza üçün əsas tapşırıqları belə mikroxidmətlərə bölmək olar:

- **Registration (Qeydiyyat)** – istifadəçi hesabının yaradılması
- **Search (Axtarış)** – məhsulları axtarma funksiyası
- **Payments (Ödənişlər)** – ödəniş əməliyyatları
- **Ratings (Qiymətləndirmə)** – məhsul reytinqləri
- **Reviews (Rəy yazmaq)** – istifadəçi rəyləri

Bu mikroxidmətlər **bir-biri ilə və müştəri ilə əlaqə saxlayır**, amma **stateless** işləyir. Yəni **sorğu və cavab müstəqildir**, çünki məlumatlar hər mikroxidmətin özündən ayrı saxlanılır.

Serverless (Server Olmadan Arxitektura)

Serverless arxitektura bulud xidmətləri tərəfindən təklif olunur, məsələn: **AWS, GCP, Azure** və s.

Bu platformalar veb tətbiqləri **serverləri idarə etmədən qurmağa və işlətməyə imkan verir**. Veb tətbiqlər **stateless computing container-lərdə** (məsələn, Docker) işləyir.

Üstünlükləri:

- Şirkət tətbiqləri və xidmətləri **asanca qurur və yerləşdirir**, amma **server infrastrukturunu idarə etməyə ehtiyac yoxdur**.
- **Serverlərin təminatı, miqyaslanması və texniki baxımı** bulud təminatçısı tərəfindən həyata keçirilir.
- Beləliklə, **tətbiqlər və verilənlər bazaları işləməsi üçün server idarəsi ilə məşğul olmaq lazım deyil**.

Sadə desək, **serverless = tətbiq işləyir, amma serverlərlə sən məşğul olmursan, bütün idarəetmə bulud provayder tərəfindən həyata keçirilir**.

Front End vs. Back End

Biz bəzən **front end və back end** web inkişafı terminlərini eşitmişik. Həmçinin **Full Stack web development** termini də var ki, bu həm front end, həm də back end-i əhatə edir. Bu terminlər veb tətbiq inkişafı ilə bağlıdır, çünki **web inkişaf dövrünün böyük hissəsini təşkil edir**.

Lakin **front end və back end fərqli sahələrdir**: hər biri veb tətbiqin bir tərəfini əhatə edir və **müxtəlif funksiyaları yerinə yetirir, fərqli sahələrdə işləyir**. **Front end**, veb tətbiqin **istifadəçi ilə birbaşa qarşılıqlı əlaqədə olan hissəsidir** (client-side, yəni brauzerdə). Bu komponentlər **veb səhifənin görünən hissəsini** təşkil edir və adətən **HTML, CSS və JavaScript**-dən ibarətdir. Bu kod **brauzerimiz tərəfindən real vaxtda işlənir və göstərilir**, yəni istifadəçi gördüyü interfeys buradan yaranır. Müasir veb tətbiqlərdə **front-end hissəsi** bütün ekran ölçülərinə uyğunlaşmalı və istənilən cihazda, istənilən brauzerdə problemsiz işləməlidir. Əgər front-end hissəsi yaxşı optimallaşdırılmayıbsa, bütün veb tətbiq ləng və istifadə üçün çətin görünə bilər. Belə hallarda istifadəçilər düşünə bilər ki, problem serverdə və ya onların internet bağlantısındadır, amma əslində problem **istifadəçinin brauzerində, yəni client tərəfdə** olur. Buna görə də front-end hissəsi mümkün qədər çox platforma, cihaz (mobil də daxil olmaqla) və ekran ölçüləri üçün optimallaşdırılmalıdır. Front-end kodlaşdırmasını öyrənmək və təcrübə etmək üçün bir çox onlayn saytlar mövcuddur. Məsələn, belə bir platformada biz kod redaktoru ilə işləyə, yazdığımız və formatladığımız mətni görə, həmçinin onun üçün yaradılan **HTML**, **CSS** və **JavaScript** kodlarını izləyə bilərik.

Sadə kod:

```
<p><strong>Welcome to Hack World</strong><strong></strong></p>

<p></p>

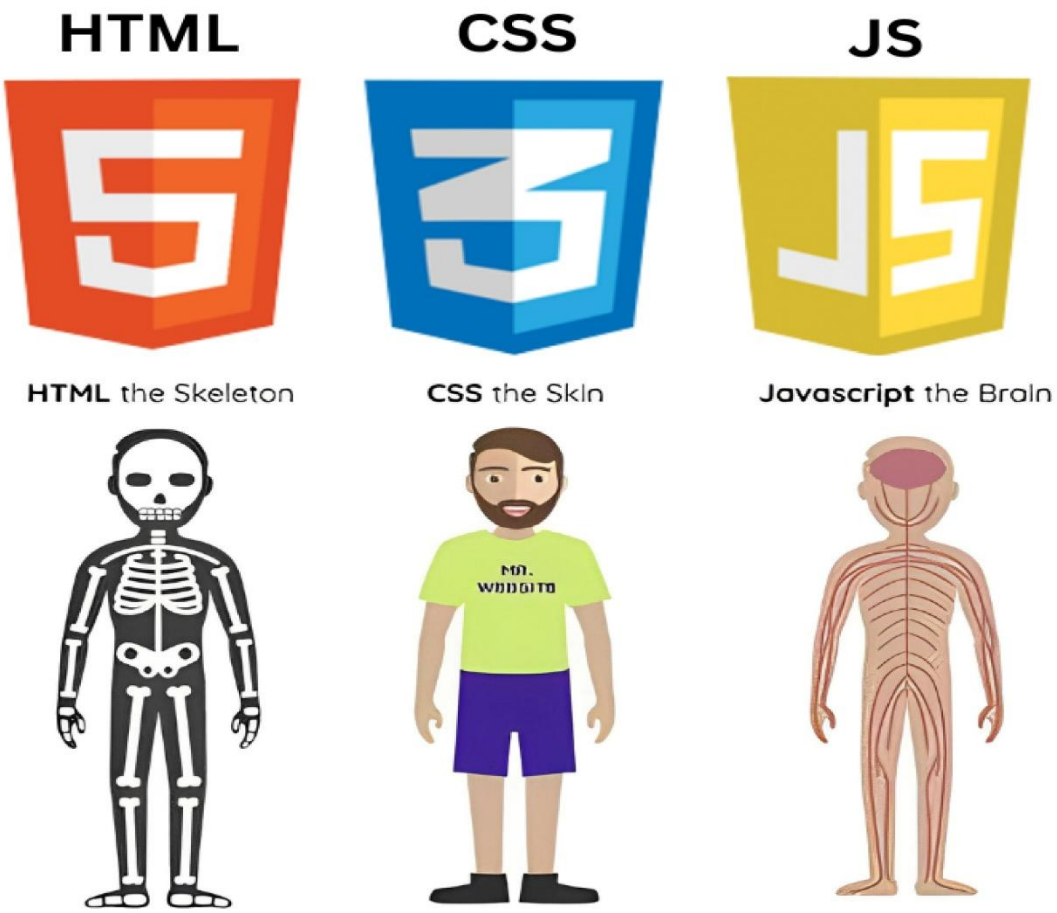
<p><em>This is some italic text.</em></p>

<p></p>

<p><span style="color: #0000ff;">This is some blue text.</span></p>

<p></p>

<p></p>
```



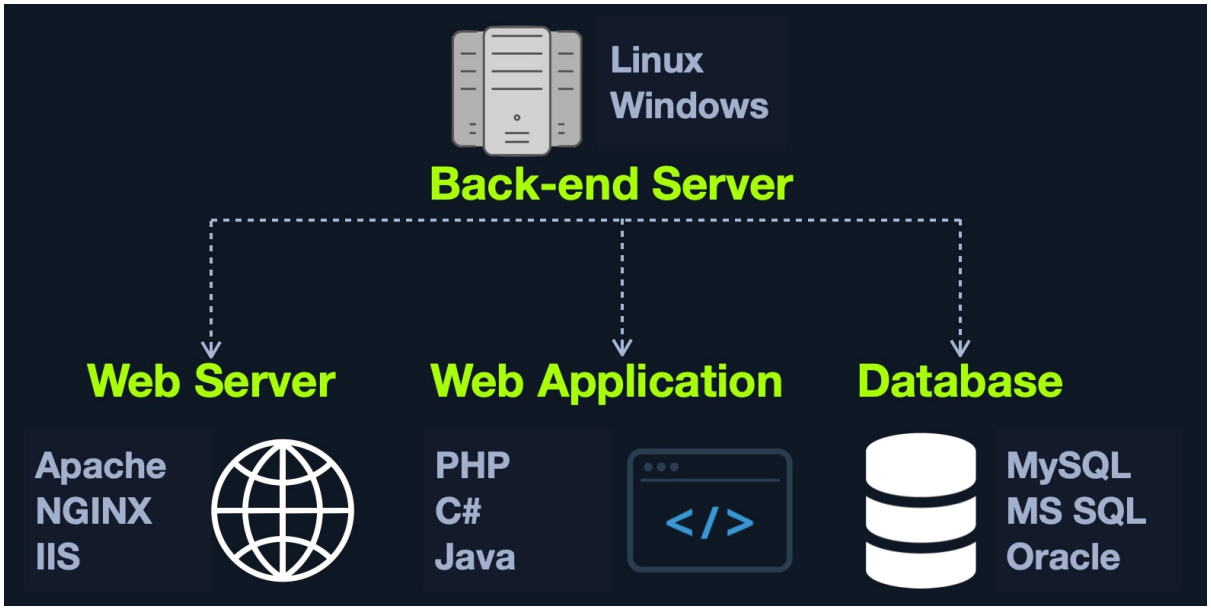
Back End

Veb tətbiqin arxa hissəsi **bütün əsas funksionallığı** təmin edir. Bu funksiyaların hamısı arxa serverdə icra olunur və tətbiqin düzgün işləməsi üçün lazım olan hər şey burada işlənir. Bu hissəni biz heç vaxt birbaşa görmürük və ya onunla qarşılıqlı əlaqədə olmuruq, amma arxa hissə olmadan bir vebsayt sadəcə statik veb səhifələr toplusundan ibarət olar. Veb tətbiqlər üçün dörd əsas arxa hissə komponenti mövcuddur:

Back end Serverlər - Digər bütün komponentləri saxlayan aparat və əməliyyat sistemi. Adətən Linux, Windows kimi əməliyyat sistemlərində və ya konteynerlərdə işləyir.

Məlumat bazaları (Databases) - Veb tətbiqin məlumatlarını saxlayan və əldə edən sistemlərdir. Əlaqəli (relational) bazalara misallar**: MySQL, MSSQL, Oracle, PostgreSQL**. Əlaqəsiz (non-relational) bazalara misallar: **NoSQL, MongoDB**.

İnkişaf Çərçivələri (Development Frameworks) - Veb tətbiqin əsas hissəsini hazırlamaq üçün istifadə olunan alətlərdir. Tanınmış çərçivələrə misallar: **Laravel (PHP), ASP.NET (C#), Spring (Java), Django (Python), Express (NodeJS JavaScript).**


Linux
Windows
Back-end Server
Web Server
Web Application
Database
Apache
NGINX
IIS
PHP
C#
Java
</>
MySQL
MS SQL
Oracle