

Web Uygulamaları Mimarileri ve Güvenliđi

METU CCLUB

erbiL Karaman



başlıklar...

- Web uygulama mimarileri
- Uygulama geliştirme ve framework kullanımı
- Güvenli web uygulamaları için öneriler

başlamadan önce...

- Mimari neden önemli?
- Google'in başarısı nerede gizli?
- Webde güvenlik nedir?

ana parametreler...

Güvenlik

Ölçeklenebilirlik

Süreklilik

Kodun Yönetilebilirliği

1. bodos mimari...

- Genellikle fazla heyecanlı gençler tarafından tercih edilen mimaridir(!)
- Görsellik, mantık ve veri iç içedir
- Genellikle tüm sistem tek bir fiziksel makinededir
- Webin %90lık kısmını oluşturur
- Genellikle onlarca potansiyel güvenlik açığı barındırır
- En hızlı başlangıç gibi gözükse de uzun vadede en çok acı çektirendir



index.php

```
<html>
  <head>
    <title>Haberler</title>
  </head>
  <body>
    <?php
      if($_SESSION['kullanici'] == 'admin'){
    ?>
    <h1>Yeni Haber Ekle</h1>
    <form action="haber_ekle.php" method="post">
    <p>
      Başlık:
      <input type="text" name="baslik" value=""><br>
      Haber içeriği:
      <input type="textarea" name="icerik" value=""><br>
      <input type="submit" name="gonder" value="Haber Ekle"><br>
    </p>
    </form>
    <? } else if ($_SESSION['kullanici'] == 'guest') {?>
      HTML biseyler
    <? } ?>
  </body>
</html>
```

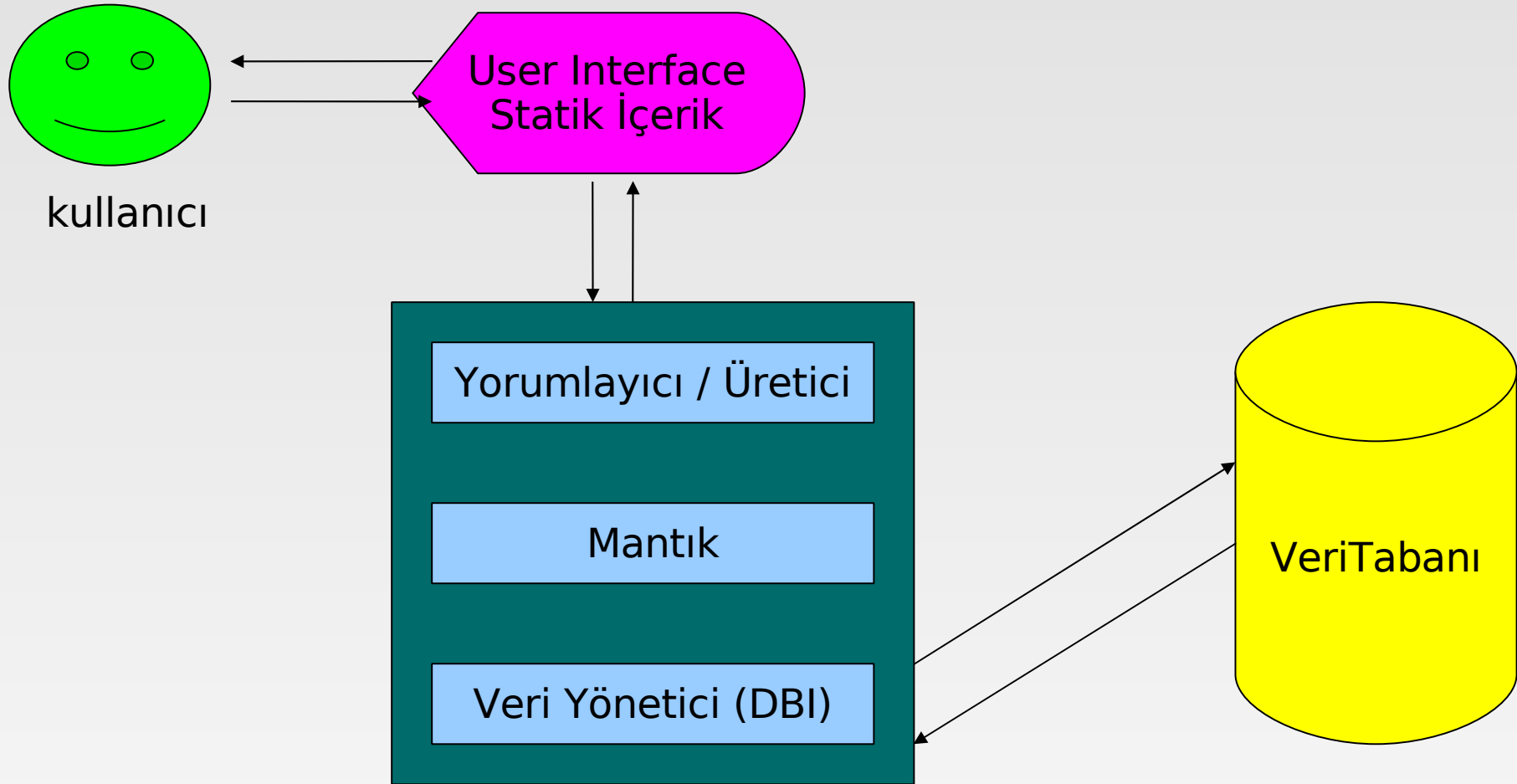
ana parametrelerin değerlendirilmesi

Güvenli Değil

Ölçeklenemez (Sıkışık Mimari)

Üretimi zahmetli ve sürekliliğe uygun değil

Three-Tier



Three-Tier

Presentation tier

The top-most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.



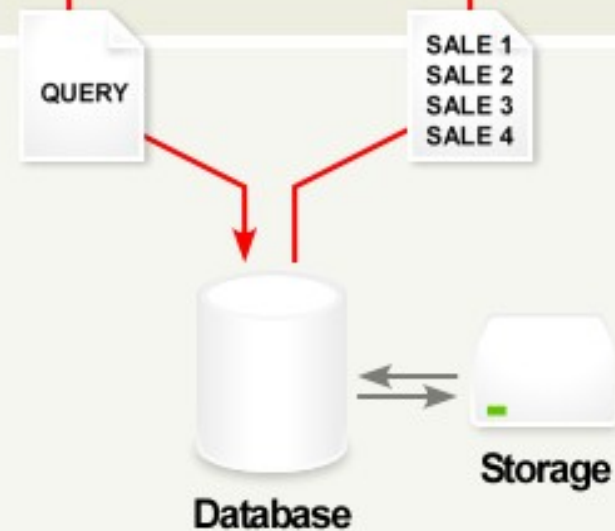
Logic tier

This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.



Data tier

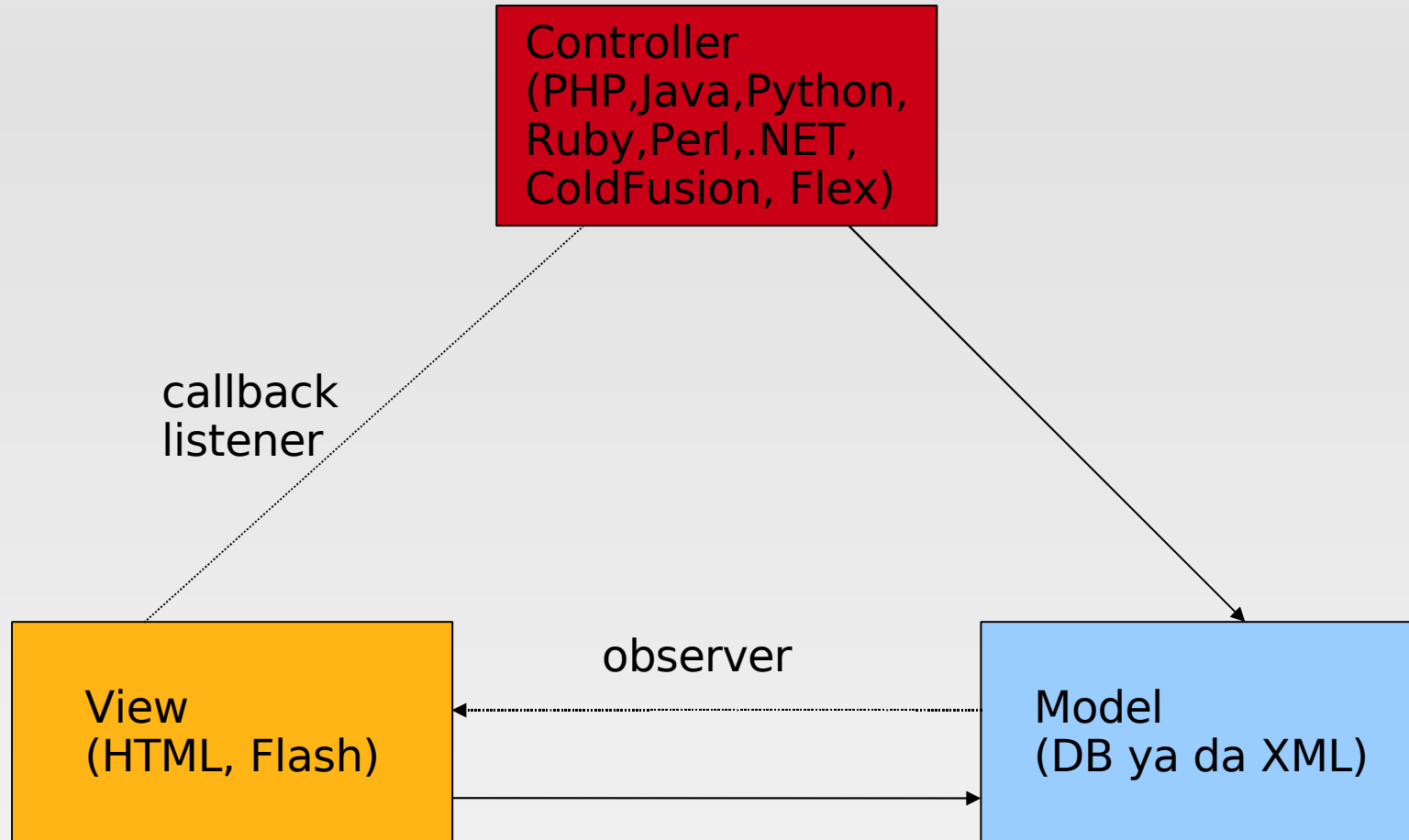
Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.



Three-Tier

- Katmanlar genellikle ayrı fiziksel makinelerde
- Uygulaması en basit (lineer) ve en çok tercih edilen dağıtık mimari
- Kullanıcı sadece arayüz katmanı ile etkileşimde.
- Katmanlar birbirine bağımlı değil, birindeki hata diğerini etkilemiyor.
- Kolay yönetilebilir ve sürekliliğe uygun
- En hızlı genişletilebilir mimari
- Genellikle tek bir iş için özelleşmiş yapı, doğal olarak en iyi performansı sunuyor

MVC(Model-View-Controller)



View

- Kullanıcının hareketlerinden ve görsel içerikten sorumludur.
- Kullanıcının hareketlerini Model katmanına haber verir. (Event handler – callback)
- Modelden aldığı veriyi kullanarak arayüzü oluşturur.

Controller

- View katmanından event(callback) bekler, bu nedenle katmana direkt bağlıdır.
- Her event için ne yapacağı önceden tanımlanmıştır, event ile gelen veriyi alır, işler, sonucunu modele iletir.

Model

- Controllerdan gelen mesaja göre Model sahip olduğu veriyi gözden geçirip gerekli bilgiyi üretir / düzenler.
- Genellikle bir observer viewa değişiklikleri haber verir.

MVC

- Yeni nesil web uygulamalarına (Web2.0-Web3.0) en uygun mimaridir.
- View genellikle güncel veri barındırır.
- Güvenlidir çünkü kullanıcı taraflı etkileşimi minimize eder.
- Katmanlar genellikle tek bir Framework içerisinde yer alır.
- Geliştirilmesi ve alışması zor olsa da en kullanışlı yapıyı sunar.
- Dağıtılabilir yapıya uyarlanması Three-Tiere göre daha zahmetlidir.

Yaygın MVC frameworkleri:

PHP için:

Zend Framework
Symfony

Python için:

Django

Perl için:

Catalyst

Ruby için:

Ruby on Rails

Java için:

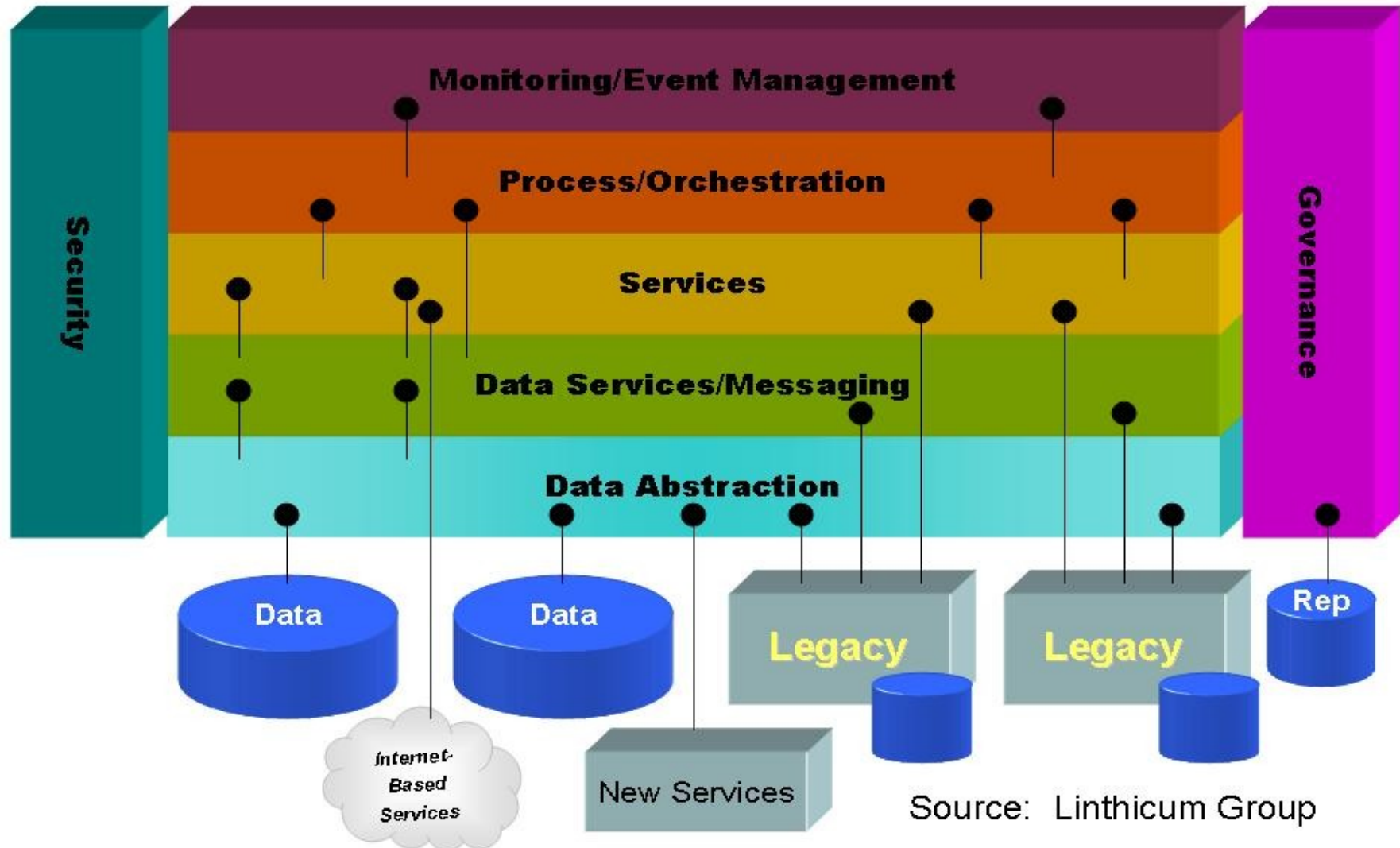
Spring Framework
Struts

.NET için:

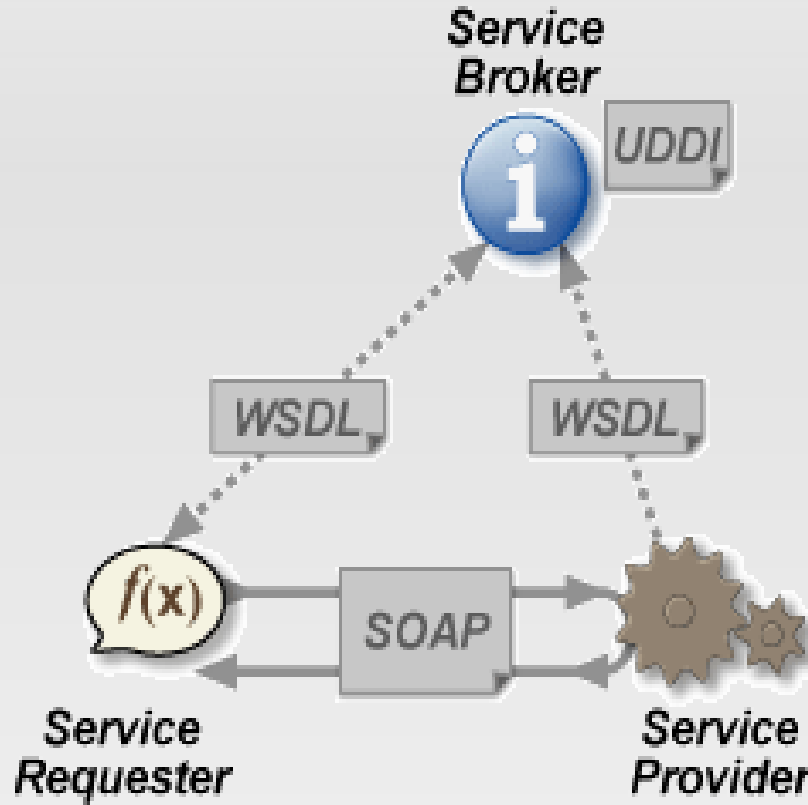
MonoRail
The Spring.NET

SOA (Service Oriented Architecture)

SOA Meta-Model



SOA ve Web Hizmetleri (Web Services)



- En kompleks yapıdır.
- Uygulamalar genellikle servislerin yetenekleriyle (API kapsamı ile) sınırlıdır.
- Aıştıđımız protokoller enkapsüle edilmiştir. (HTTP GET,POST vs. yoktur)
- Bütün veri XML(SOAP) olarak servis edilir ve WSDL olarak sunulur.

SOA

- Servis sağlayıcı için en güvenli yapıdır.
- Yapı itibari ile dağıtıktır (ölçeklenebilir), her türlü alt/üst sisteme entegre edilebilir.
- Sürekliliği ve mantık katmanını büyük ölçüde servis sağlayıcıya bırakır.
- Geliştirici için uygulamayı tek katmana indirger.
- Servis ve uygulama katmanı genellikle farklı lokasyonlardadır.

Ör: Yahoo Web Services

Web Uygulaması Frameworkleri ve Kütüphaneler

**TEKERLİĞİ YENİDEN İCAT
ETMEYİN !!!**

Web Uygulaması Frameworkleri ve Kütüphaneler

- Kararlı çalışan web uygulaması geliştirmek tecrübe ve zaman ister.
- Yaptığınız işlerin çoğu daha önce başkaları tarafından da onlarca kez yapılmıştır ve test edilmiştir. O halde ne diye yeniden yapasınız.
- Yaygın kullanılan frameworkler sık sık güncellenir ve yenilenir.
- Size zaman ve hız kazandırır.
- Kodunuz yalnızca 'yapmak istediğiniz şeyi' içerir.
- Uygulama katmanlarını birbirinden bağımsız kılar.

Web Uygulaması Frameworkleri ve Kütüphaneler

Kullanıcı yetkilendirmesi (auth bypass), session yönetimi (XSS), veri tabanı etkileşimi (SQL injection), uygulama güvenliği açısından en kritik noktalardır. Yeterli tecrübeniz yok ise hata yapmanız (açık bırakmanız) an meselesidir.

Web Uygulaması Frameworkleri ve Kütüphaneler

PHP için:

PEAR (PHP Extension and Application Repository)

<http://pear.php.net/>

(DB(MDB2), HTML_QuickForm, Auth)

Perl için:

CPAN kütüphaneleri

AJAX için:

GoogleWebToolkit(Java)

Dojo Toolkit

Apache XAP

CleanAJAX