Farah Zaib Khan

INFO90002
Database Systems & Information Modelling

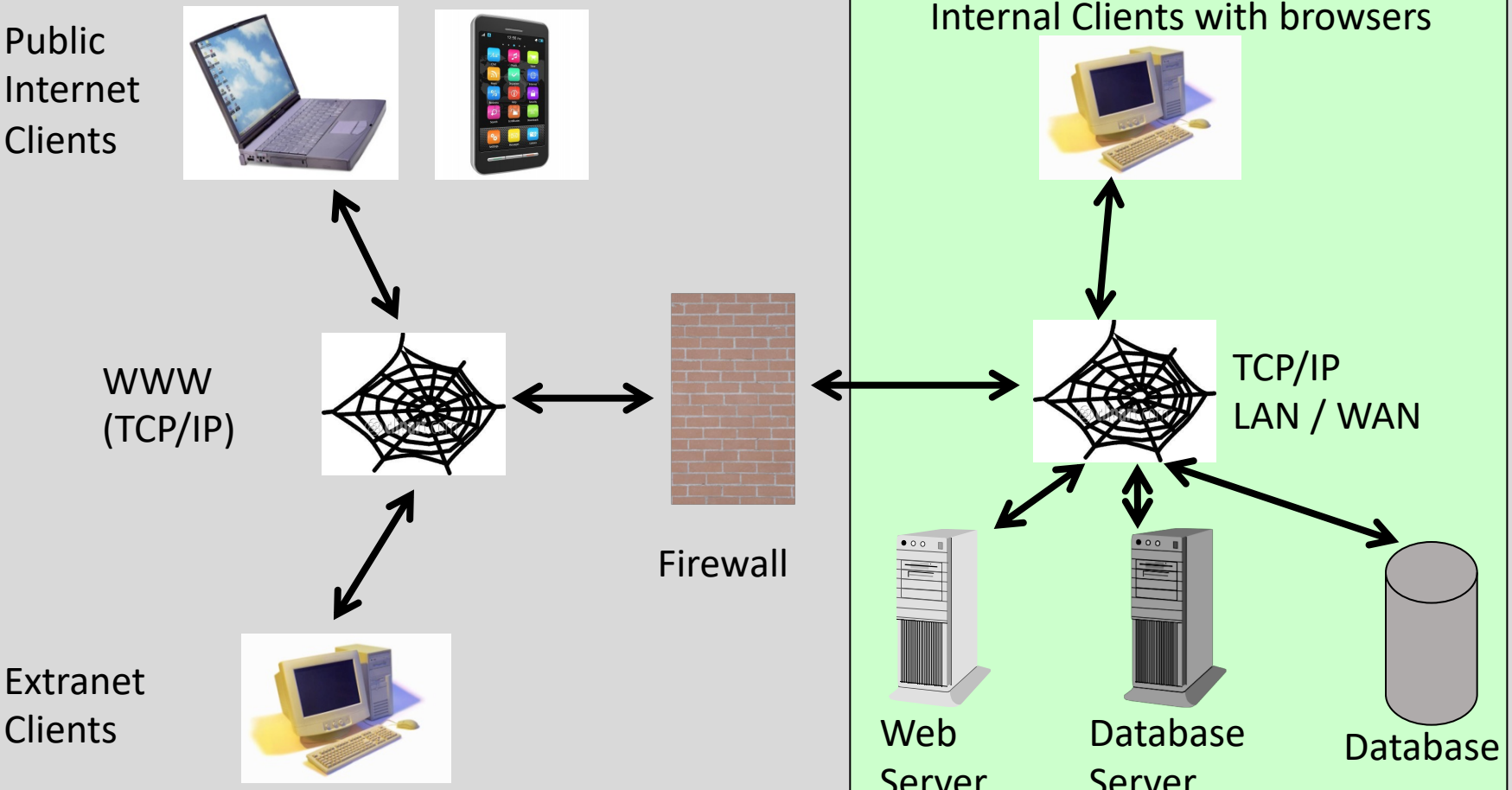
Week 07
Web Apps

Today's Session…

- Why web apps?
- How web apps work
- Making an HTML document
- Connecting to the DB
- Demo web app
- Web services

THE UNIVERSITY OF MELBOURNE

Architecture of a web app



Why create web applications?

- Web browsers are ubiquitous
- No need to install client software for external customers
- Simple communication protocols
- Platform and Operating System independent
- Reduction in development time and cost
- Has enabled eGov, eBusiness, eCommerce, B2B, B2C

Web infrastructure

Browser

- Software that retrieves and displays HTML documents

Web Server

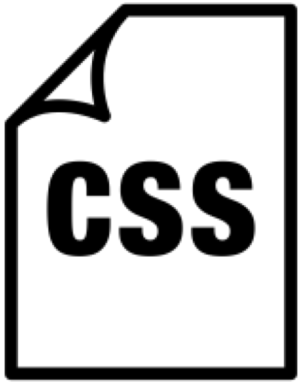
- Software that responds to requests from browsers by transmitting HTML and other documents to browsers

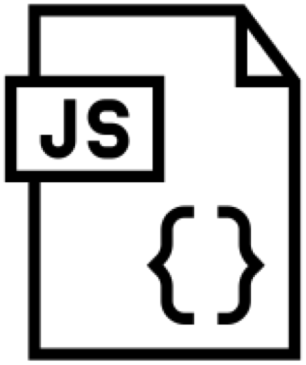
Web pages (HTML documents)

- **Static web pages**
  - content established at development time
- **Dynamic web pages**
  - content dynamically generated using data from database

World Wide Web (WWW)

- The total set of interlinked hypertext documents residing on Web servers worldwide

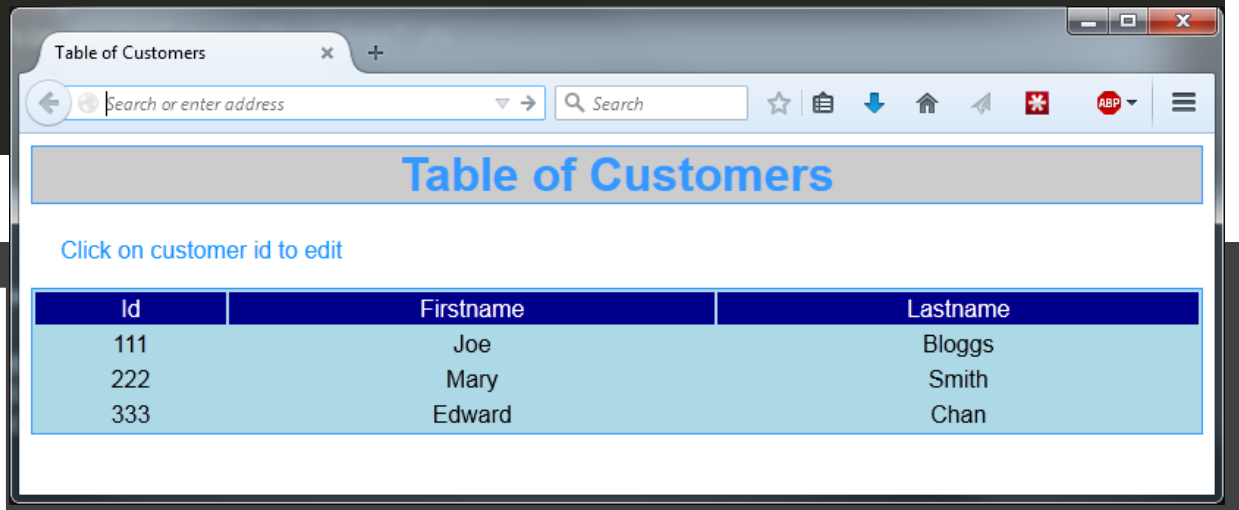




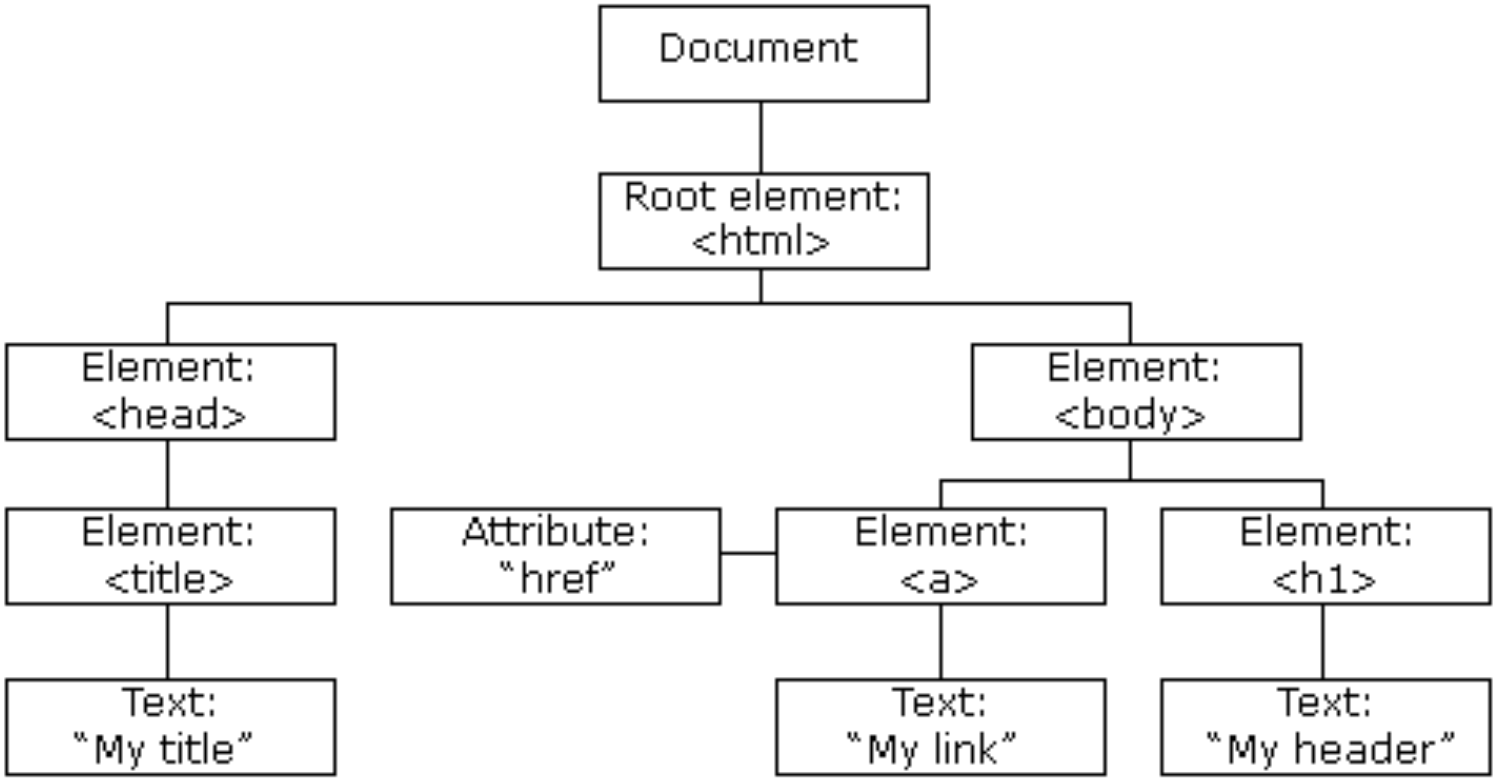
- Hypertext Markup Language (HTML)
  - Markup language used to define a web page
- Cascading Style Sheets (CSS)
  - Control appearance of an HTML document
- JavaScript (JS)
  - Scripting language that enable interactivity in HTML documents
- Extensible Markup Language (XML)
  - Markup language used to transport data between web services

Web-related languages

```html
<head>
    <title>Table of Customers</title>
    <link rel="stylesheet" href="simple.css" type="text/css" />
</head>

<body>
    <h1>Table of Customers</h1>
    <p>Click on customer id to edit</p>
    <table>
        <thead>
            <tr><td>Id<td>Firstname<td>Lastname</tr>
        </thead>
        <tr><td>111<td>Joe<td>Bloggs</tr>
        <tr><td>222<td>Mary<td>Smith</tr>
        <tr><td>333<td>Edward<td>Chan</tr>
    </table>
</body>
```



Web page = HTML document



Structure of an HTML document

THE UNIVERSITY OF MELBOURNE

<HEAD> … </HEAD> → document header

<BODY> … </BODY> → document body

<H1> … </H1> → Heading type 1

<H6> … </H6> → .. To Heading type 6

<P> … </P> → paragraph

<TABLE> → table

<TR> → table row

<TD> → table data

<UL> → list

<LI> → list item

Important HTML elements

HTML list

Does order matter?

Yes

```
<ol>
    <li></li>
    <li></li>
    <li></li>
</ol>
```

No

```
<ul>
    <li></li>
    <li></li>
    <li></li>
</ul>
```

https://www.w3schools.com/html/html_lists.asp

HTML list

*You were hired to build a **food recipe** website. Which HTML elements would you use for the **ingredients list** and **directions list**, respectively?*

*a) and *

*b) and *

*c) and *

*d) and *

HTML list

*You were hired to build a **food recipe** website. Which HTML elements would you use for the **ingredients list** and **directions list**, respectively?*

*a) and *

*b) and *

***c) and ***

*d) and *

Ingredients

- 1½ c. sifted cake flour
- 1½ tsp. baking powder
- ¼ tsp. salt
- ½ c. unsalted butter
- 1 c. sugar
- 2 large eggs
- ½ tsp. vanilla extract
- ½ c. whole milk

```
▼<ul class="recipe-list recipe-ingredients-list">
   <li class="recipe-ingredients-item" itemprop="ingredients">
   1½ c. sifted cake flour</li>
   <li class="recipe-ingredients-item" itemprop="ingredients">
   1½ tsp. baking powder</li>
   <li class="recipe-ingredients-item" itemprop="ingredients">
   ¼ tsp. salt</li>
   <li class="recipe-ingredients-item" itemprop="ingredients">
   ½ c. unsalted butter</li>
   <li class="recipe-ingredients-item" itemprop="ingredients">
   1 c. sugar</li>
   <li class="recipe-ingredients-item" itemprop="ingredients">
   2 large eggs</li>
   <li class="recipe-ingredients-item" itemprop="ingredients">
   ½ tsp. vanilla extract</li>
   <li class="recipe-ingredients-item" itemprop="ingredients">
   ½ c. whole milk</li>
 </ul>
```

HTML list

Run »

```
<!DOCTYPE html>
<HTML> <HEAD> <title>Some Simple Lists</title> </HEAD>
•<BODY bgcolor="#FFFF99">
<H1>My Fruit and Medal List </H1>
<UL>
  <LI>Banana</LI>
  <LI>Orange</LI>
  <LI>Grape</LI>
</UL>
<OL>
  <LI>Gold Medal</LI>
  <LI>Silver Medal</LI>
  <LI>Bronze Medal</LI>
</OL>
<DL>
  <DT>Apple
    <DD>A crisp juicy fruit, red, yellow or green in
colour.
  <DT>Banana
    <DD>A tropical fruit, yellow skinned.
</DL>
</BODY> <HTML>
```

My Fruit and Medal List

- Banana
- Orange
- Grape

1. Gold Medal
2. Silver Medal
3. Bronze Medal

Apple
: A crisp juicy fruit, red, yellow or green in colour.

Banana
: A tropical fruit, yellow skinned.

HTML table

Item	Price	Calories
Espresso	$3.00	1 Cal
Cappuccino	$4.50	68 Cal
Flat white	$4.00	68 Cal

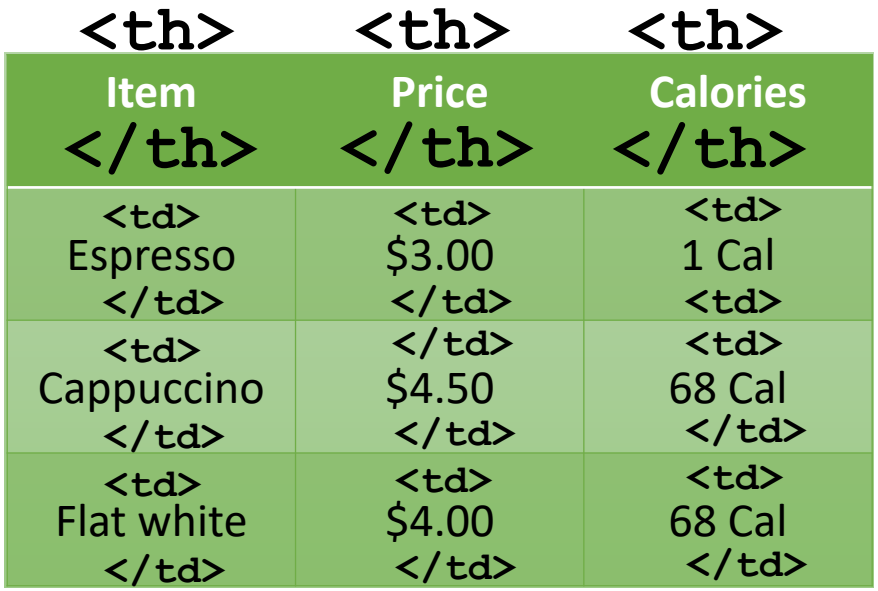
Header

Rows

Columns

HTML table

```
<table>
```




```
</table>
```

<table>

- <tr>
  - <th>**Item**</th>
  - <th>**Price**</th>
  - <th>**Calories**</th>
- </tr>
- <tr>
  - <td>**Espresso**</td>
  - <td>**$3.00**</td>
  - <td>**1**</td>
- </tr>
- <tr>
  - <td>**Cappuccino**</td>
  - <td>**$4.50**</td>
  - <td>**68**</td>
- </tr>
- <tr>
  - <td>**Flat White**</td>
  - <td>**$4.00**</td>
  - <td>**68**</td>
- </tr>

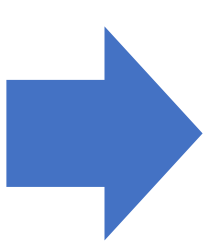
HTML table

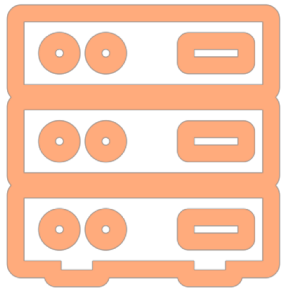
</table>

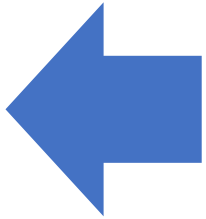
HTML form

Username: fkhan

Password: *******

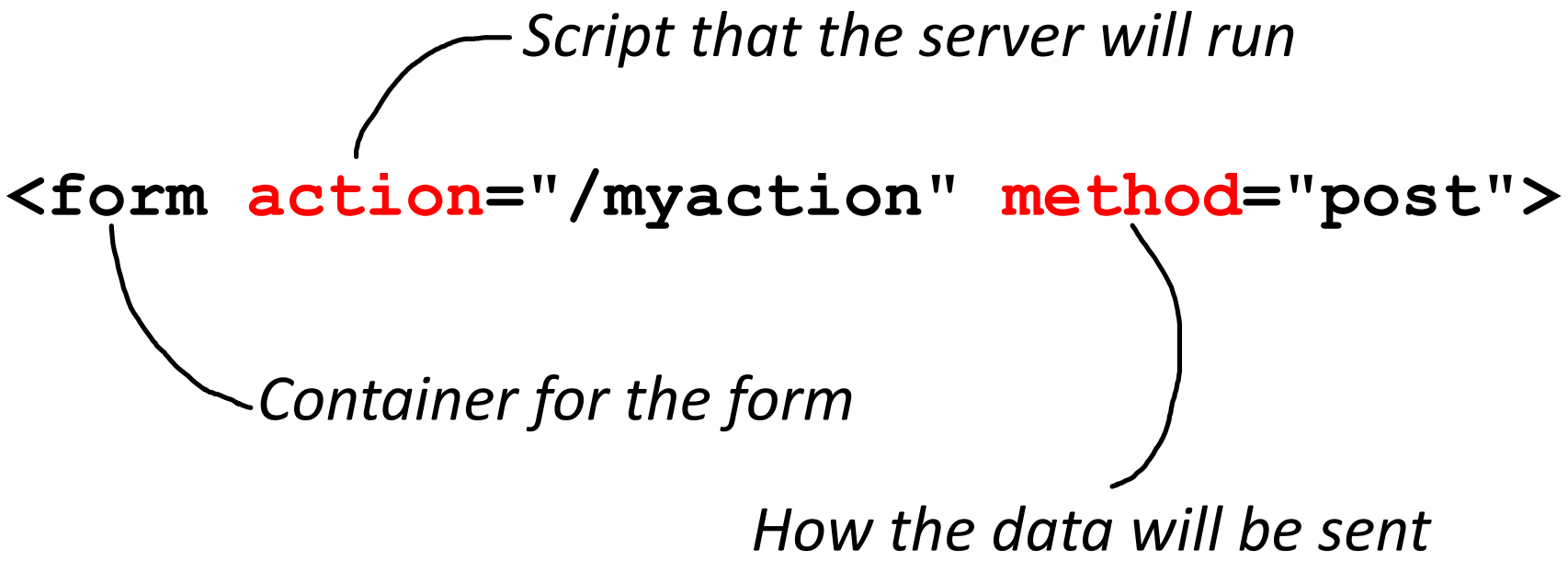






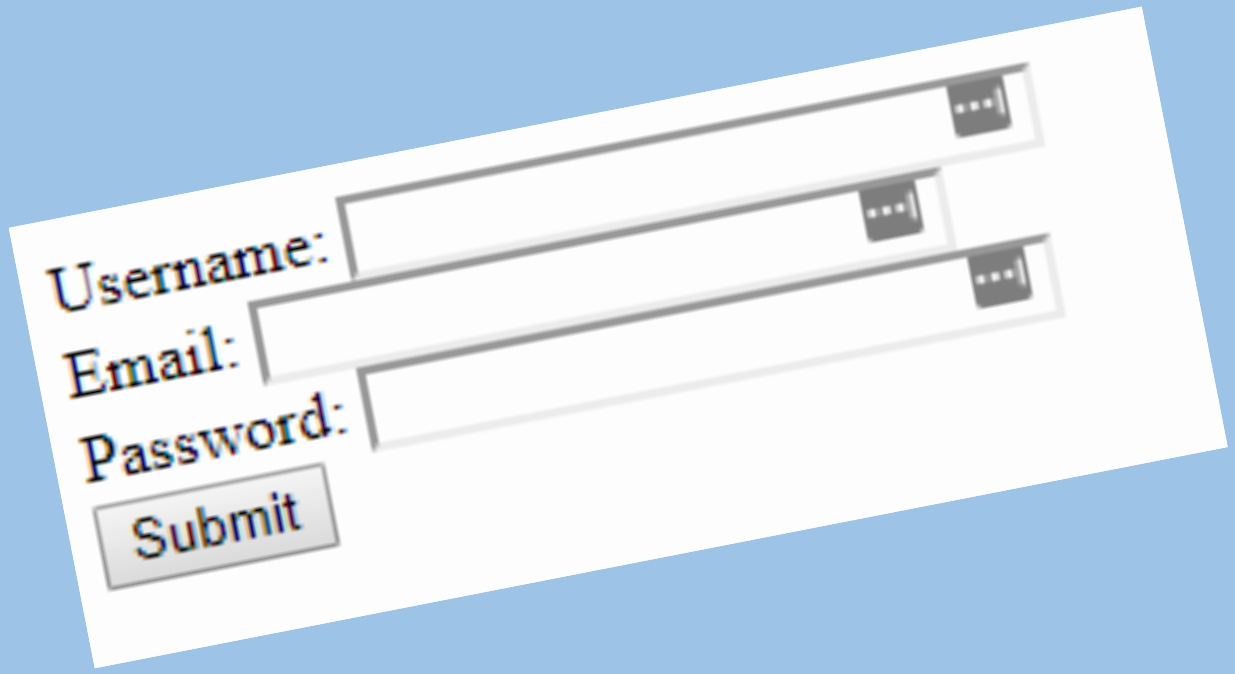
Welcome, Farah!

HTML form


Script that the server will run
<form action="/myaction" method="post">
Container for the form
How the data will be sent

HTML form

```
<form action="/myaction" method="post">
    Username:
  <input type="text" name="username"><br>
  Email:
  <input type="email" name="email"><br>
  Password:
  <input type="password" name="psw"><br>
  <input type="submit">
</form>
```


Username:
Email:
Password:
Submit

- Forms allow users to input data to a web page
- The web server process the user's input using the file named in the 'action' attribute.

```
<form action = "buy.pl" method="post">

<p> What would you like to buy? <br>
<input type="text" name="product">

<p> How many? <br>
<input type="text" name="quantity"> <br>

<input type="submit" value="Do e-Commerce">
</form>
```

Example HTML form

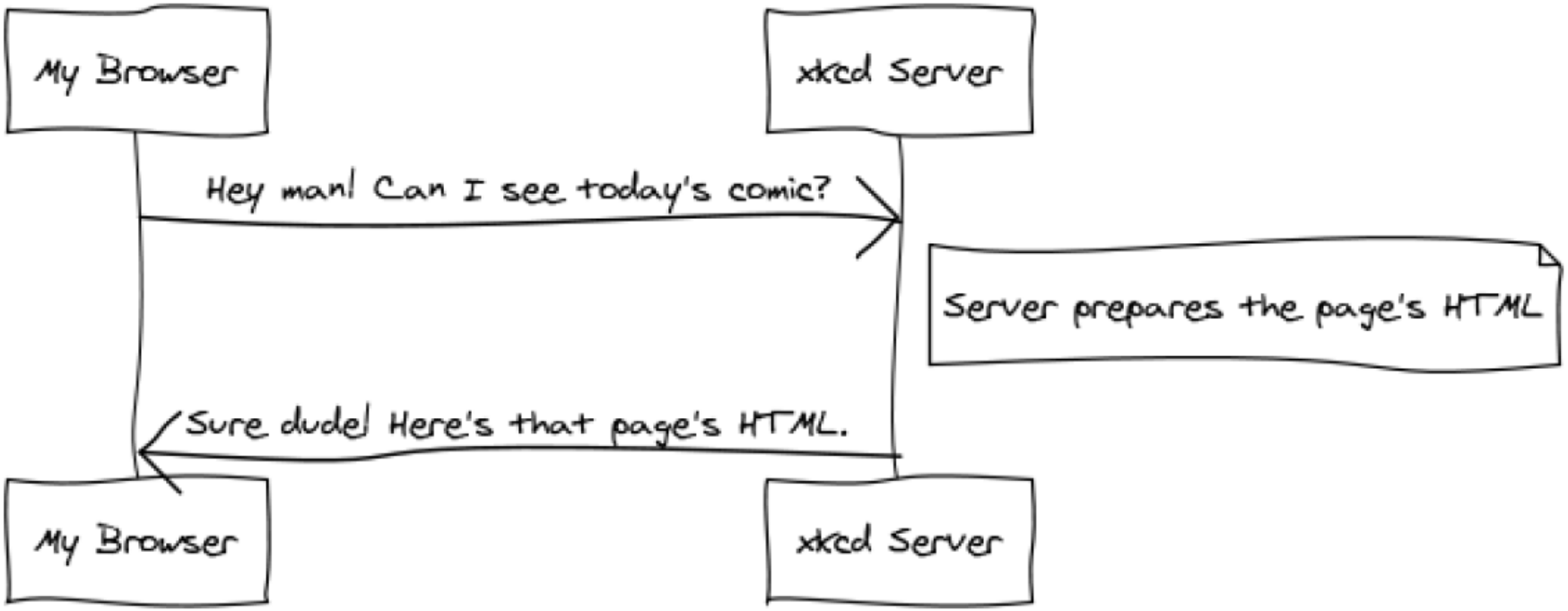
browser displays form,
sends input data to a script called 'buy.pl'

HTML form

HTTP: how HTML documents move

- User wants to see a web page
- Types URL into browser
- Browser fetches page from server and displays it

could simply load a pre-prepared HTML file (static), or query the database to produce a tailored HTML file (dynamic)



picture source: Symfony Book

Static vs Dynamic web pages

STATIC web page

- the URL identifies a file on the server's file system
- server fetches the file and sends it to the browser
- the file contains HTML
- browser interprets the HTML for display on screen

DYNAMIC web page

- URL identifies a program to be run
- web app runs the program
- program typically retrieves data from database
- elements such as TABLE, LIST are populated with data
  - web app uses LOOPS to fill the contents of TABLEs and LISTs.
  - e.g. SELECT * FROM Product; (returns a set of product entities)
  - FOR p IN ProductList, print a row in HTML table

THE UNIVERSITY OF MELBOURNE

```php
<?php

print '<h1> This page selects from a table </h1>';

print '<p> connecting to database ... </p>';
// connect to server, select database
$link = mysql_connect('localhost', 'root', '')
    or die('Could not connect: ' . mysql_error());
print '<p> connected successfully </p>';
mysql_select_db('webappdemo') or die('could not select database');

// perform SQL query
$query = 'SELECT * FROM mytable';
$result = mysql_query($query) or die('Query failed: ' . mysql_error());

print '<h2> table starts now </h2>';

// print results in an HTML table
print "<table>\n";
while ($line = mysql_fetch_array($result, MYSQL_ASSOC)) {
    print "\t<tr>\n";
    foreach ($line as $col_value) {
        print "\t\t<td>$col_value</td>\n";
    }
    print "\t</tr>\n";
}
print "</table>\n";

```

Simple web app using PHP and SQL

- Program logs into db
- Selects all rows from database table
- Displays them inside an HTML table

THE UNIVERSITY OF MELBOURNE

This page selects from a table

connecting to database ...

connected successfully

table starts now

1 first row
2 second row
3 third row - working nicely

form starts now

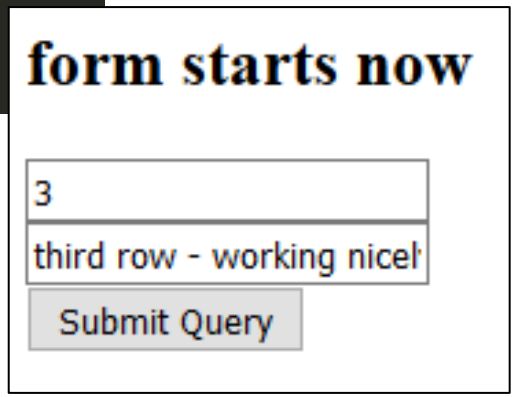
3
hird row - working nicely

```php
<?php

print '<h1> This page selects from a table </h1>';

print '<p> connecting to database ... </p>';
// connect to server, select database
$link = mysql_connect('localhost', 'root', '')
    or die('Could not connect: ' . mysql_error());
print '<p> connected successfully </p>';
mysql_select_db('webappdemo') or die('could not select database'

// perform SQL query
$query = 'SELECT * FROM mytable';
$result = mysql_query($query) or die('Query failed: ' . mysql_er

print '<h2> table starts now </h2>';

// print results in an HTML table
print "<table>\n";
while ($line = mysql_fetch_array($result, MYSQL_ASSOC)) {
    print "\t<tr>\n";
    foreach ($line as $col_value) {
        print "\t\t<td>$col_value</td>\n";
    }
    print "\t</tr>\n";
}
print "</table>\n";

```

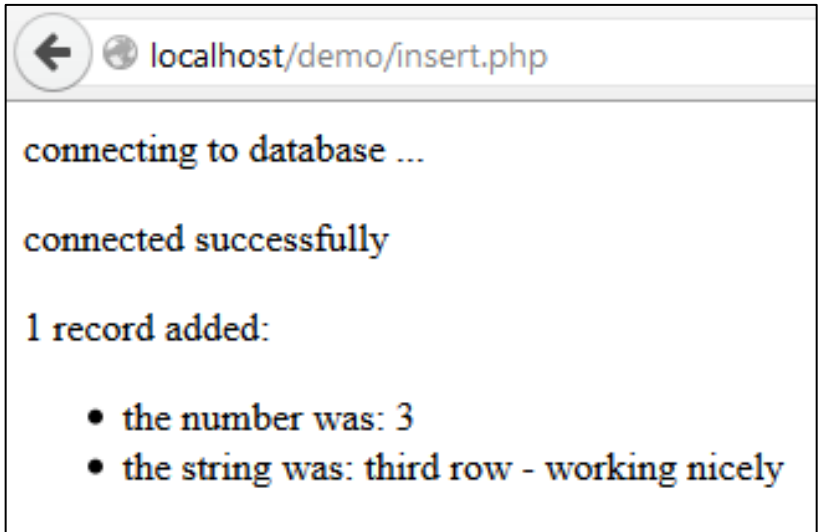
Simple web app: *select*

Simple web app: *insert*

```php
print '<h2> form starts now </h2>';

// display a form for entering data
print '<form action="insert.php" method="post">';
print '<input type="text" name="number" value="type a number" /> <br />';
print '<input type="text" name="string" value="type a string" /> <br />';
print '<input type="submit" value="send to database" />';
print '</form>';
?>
```




```php
<?php

print '<p> connecting to database ... </p>';
// connect to server, select database
$link = mysql_connect('localhost', 'root', '')
    or die('Could not connect: ' . mysql_error());
print '<p> connected successfully </p>';
mysql_select_db('webappdemo') or die('could not select database');

// form the INSERT statement from the user's input
$sql="insert into mytable values ('$_POST[number]','$_POST[string]')";

// run the INSERT statement
if (!mysql_query($sql,$link))
  die('Error: ' . mysql_error());

// print friendly message
print "<p> 1 record added:  </p>";
print "<ul>";
print "<li>the number was: " . $_POST['number'];
print "<li>the string was: " . $_POST['string'];
print "</ul>";

// close connection to database
mysql_close($link);

?>
```

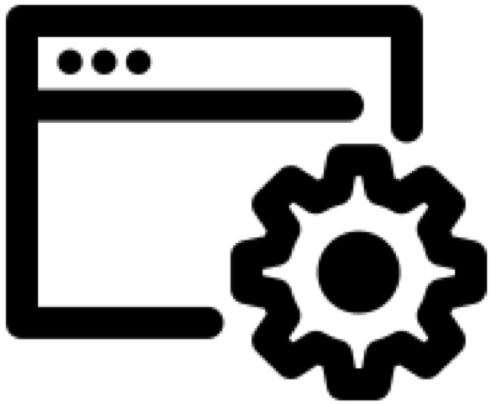



DEMO

Problems with old-style web apps

- **Placing "raw" SQL inside PHP/HTML files**
  - Mixes presentation, business, database logic
  - Hard to maintain when things change
  - Want separation of concerns e.g. MVC
- **Lots of reinvention of wheels**
  - each dev writes their own solution to common features
  - e.g. login security, presentation templates, database access
- **Increasing variety of clients e.g. phones and tablets**
  - Manually program for different platforms

Web Services



- The WWW allows humans to access remote databases
- Web Services allow *computers* to access remote databases
- 2 major approaches: SOAP and REST
  - Simple Object Access Protocol (SOAP)
  - REpresentational State Transfer (REST)
- structured data usually returned in XML or JSON format
- REST nouns are resources, addressed via URIs
- REST verbs correspond to DML statements
- GET (select), POST (insert), PUT (update), DELETE (delete)
- Try this example web service
  https://www.googleapis.com/books/v1/volumes?q=quilting

The following JSON example defines an employees object, with an array of 3 employee records:

JSON Example

```
{"employees":[
    {"firstName":"John", "lastName":"Doe"},
    {"firstName":"Anna", "lastName":"Smith"},
    {"firstName":"Peter", "lastName":"Jones"}
]}
```

The following XML example also defines an employees object with 3 employee records:

XML Example

```
<employees>
    <employee>
        <firstName>John</firstName> <lastName>Doe</lastName>
    </employee>
    <employee>
        <firstName>Anna</firstName> <lastName>Smith</lastName>
    </employee>
    <employee>
        <firstName>Peter</firstName> <lastName>Jones</lastName>
    </employee>
</employees>
```

- XML -- eXensible Markup Language
- JSON -- JavaScript Object Notation

example sourced from W3 schools

XML and JSON data formats