CTIS 256 Web Technologies II

Notes # 1

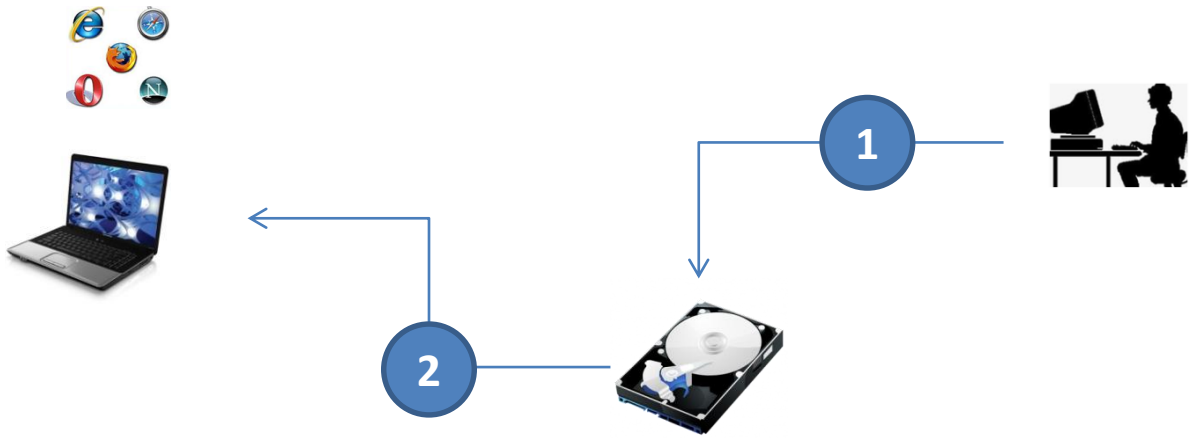
Web Arch., HTTP, Setup, NetBeans

Serkan GENÇ

Introduction

- Aim: be able to develop web-based applications using PHP (programming language) and mySQL (DBMS).
- Internet is a huge network structure connecting billions of computers.
- World Wide Web is an application running on this network structure using TCP/IP protocol. In WWW, there are two kinds of software; Web client, and Web Server.
- Web Server program is responsible for distributing its files to the outside world. Web Client (Browser) is a program that requests file from Web Servers.
- Why Web-based Application Development?
  - All application files are stored in server-side
  - **Easy to maintain**: bug fixes, adding new features, the same version for all users
  - **Platform independent**: it works in any platforms (unix,windows, iphone,etc). Develop application once for all platforms, so, it is time and cost effective.
  - **Advanced security :** no way to crack, and prevention of stealing know-how
  - **Connectivity**: one can easily connects to the system via browser and uses it immediately, no need to install, or setup.
  - **Flexible Licenses**: renting software for a specific period of time such as one month

System in CTIS 255 or Web 1 Course



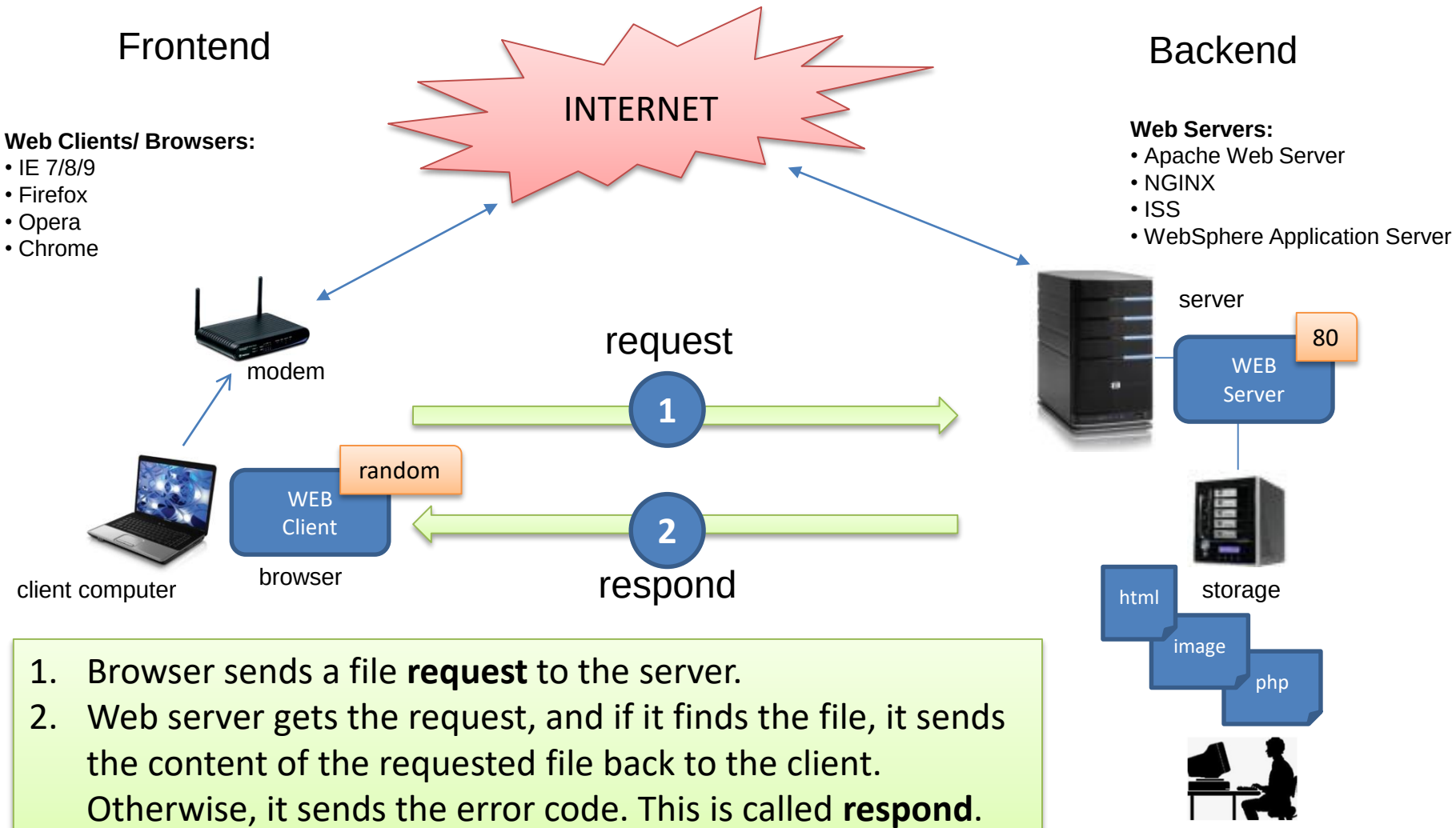
1. The user writes html codes into a file and save it into an attached storage device
2. The user clicks on html file and browser renders it.

Problem:

*The html content is written by the web programmer. Therefore, the content is **static**. To change the content, the programmer must update the content of html file manually. (Think about stock exchange rates, they are updated many times in a day.)*

Overall picture of World Wide Web Architecture

(Client-Server Architecture)



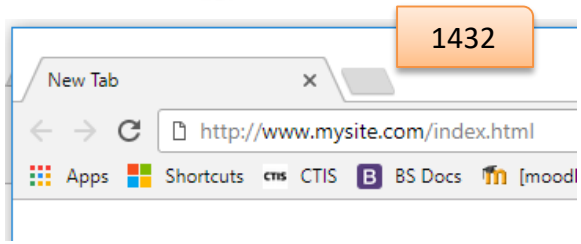
1. Browser sends a file **request** to the server.
2. Web server gets the request, and if it finds the file, it sends the content of the requested file back to the client. Otherwise, it sends the error code. This is called **respond**.
3. A request and respond pair is called a **transaction**.

WWW Architecture – **Static Content**

INTERNET

139.179.130.124

host: www.mysite.com
IP: 145.45.32.45



1 Launch a web client(browser)
Type the web address of the server (URL) into location bar (www.mysite.com) and press enter. Browser retrieves IP address from DNS server.

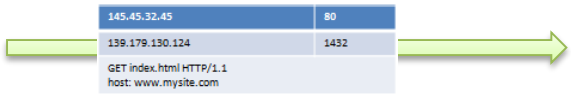
2 Browser prepares and sends a network packet called HTTP request packet *(root folder **/** sign if filename is not given, means "default file" selected by server)*

145.45.32.45 (dest IP)	80 (dest Port)
139.179.130.124 (src IP)	1432 (src Port)
GET index.html HTTP/1.1 host: www.mysite.com	

HTTP Request format

TCP/IP Packet (simplified)

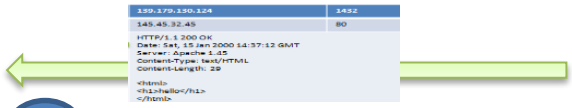
3 Browser sends an **HTTP Request** to the server.



4 Web Server gets HTTP request packet
Checks out the content of the packet
Finds the requested file in its storage.

5 Prepares HTTP response packet and sends back to client.

139.179.130.124	1432
145.45.32.45	80
HTTP/1.1 200 OK Date: Sat, 15 Jan 2000 14:37:12 GMT Server: Apache 1.45 Content-Type: text/HTML Content-Length: 29 <html> hello </html>	



6 Web Server sends an **HTTP response** packet to the client (browser).

7 Browser gets response packet, takes out HTTP part, and renders HTML codes.

80 : standard port address for Web Servers

WWW Architecture – **Dynamic Content**

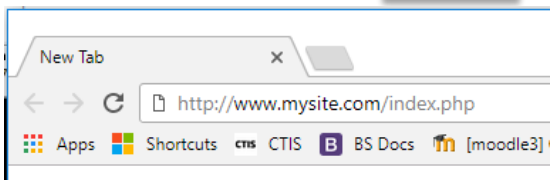
139.179.130.124

www.mysite.com
145.45.32.45

3 Browser send s HTTP Request to the server.

145.45.32.45	80
139.179.130.124	1432
GET index.php HTTP/1.1 host: www.mysite.com	

1432



WEB Server 80

4 Web Server gets HTTP request packet
Checks out the content of the packet
Finds the requested file in its storage.

5 If the file is a PHP file, then it does not open the file and send the content. Instead,

1. **Web Server sends the file to the PHP compiler,**
2. **PHP Compiler executes the codes inside the file**
3. **Web Server gets the output of php program**
4. **Sends the output to the client**

PHP Interpreter

Output of PHP program

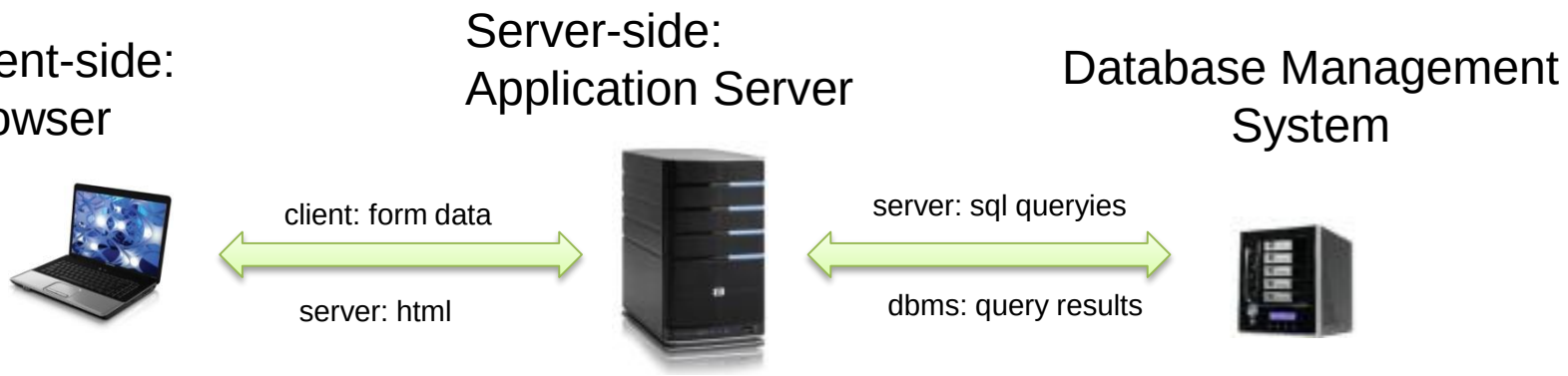
```
<html>
<h1>hello</h1>
</html>
```

6 Prepares HTTP response packet using output of PHP program and sends back to client.

139.179.130.124	1432
145.45.32.45	80
HTTP/1.1 200 OK Date: Sat, 15 Jan 2000 14:37:12 GMT Server: Apache 1.45 Content-Type: text/HTML Content-Length: 29 <html> hello </html>	

Sends to client

Three Tier Architecture



There are three basic components in Web Applications:

1. **Client-side:**
   - interface of the application (Input/Output part of Web Application)
   - displays html data produced by server-side programs (php)
   - gets input from users mostly through html forms
   - send form data to server-side programs
2. **Server-side:**
   - All server-side programs(php,jsp,etc) and resources (image, html, pdf etc) reside in server-side.
   - It gets data from client-side, processes it, and generates html codes
   - All business logic are implemented in server-side.
3. **DBMS:**
   - It may be in the same server machine with application server or may be in another machine.
   - Gets queries from server-side programs, and return results to them.

HTTP Protocol Basics

- The language between Web Client and Web Server.
- Basically, it defines how a client sends requests to a server and vice versa.
- Need to know HTTP protocol for redirection, cookie and session management, and cache management.
- You can use WireShark tool/Built-in Developer Tools/Browser Plugins to analyze HTTP packets between client and server.

  Reference: HTTP Developer's Handbook, Chris Shiflett, Sams Publishing, 2003.

Conversation Scenarios

Web Client

Web Server

1. send me page.html
2. Here it is, I am sending it
3. send me page.html
4. I could not find "page.html"

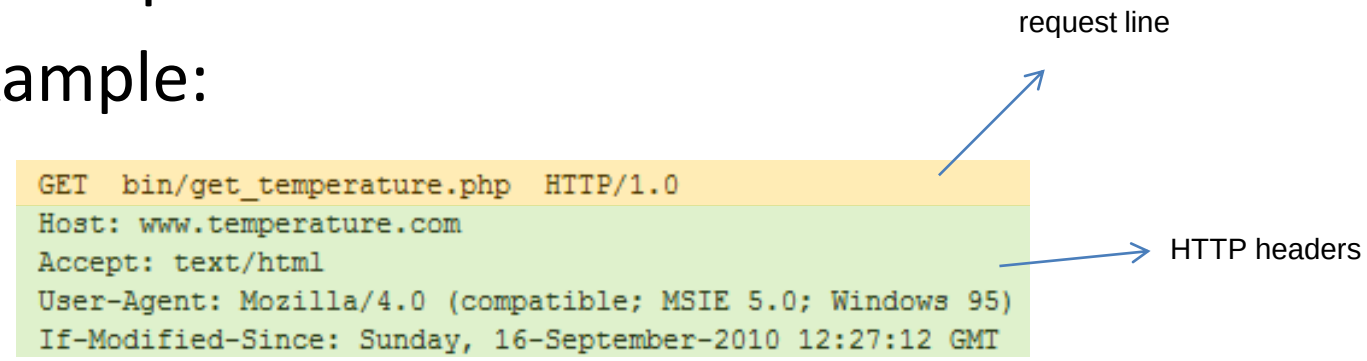
1. File/Resource Request
2. Run remote program
3. Caching
4. Redirection
5. Authentication/Security

2. Run and send the result of list.php
3. This is the result of "list.php", it is in html format.
4. Run and send the result of list.php
5. Here is the result but do not cache it, because it may change
6. Send me "test.html" if it is modified since 10.12.2016, 10:00
7. It is not modified, use cached version in your system
8. Run and send the result of list.php
9. This resource is not here, request it from this location.
10. Run and send the result of list.php
11. This file requires authentication, send me your username and password

HTTP Request - 1

- Client sends a network packet to the server about the file/resource it requests.
- There are three basic sections in the packet:
  1. Request line (the first line)
  2. HTTP headers
  3. Request content

Example:



HTTP Request - 2

- Request line Format : METHOD URL Version
- Method: *GET*, *POST*, PUT, HEAD, DELETE
- GET: Request the resource located at the specified URL
  - When you write a URL address to address bar in the browser, and press Enter key, the browser automatically generates GET request packet, and send it to the server.
  - When you click on a link, the browser generates GET request packet.
  - It is also possible to send data to server-side program with GET method without using any html form at all. In the URL, **?**var1=value1 **&** var2=value2 format is used to send data using GET method.
  - After the ? mark, remaining string is called **query string**. This string should be in **URL encoded format**. Only 0-9A-Za-z (ISO-8859-1 character set) and some reserved characters +$?_-* are allowed to be used in URL encoded string, other characters are represented by their character codes. For examle: Ali Gül →Ali**%20**G**%FC**l, here %20 and %FC represent a space character and ü letter respectively.
- POST: Sends data to the program located at the specified URL
  - After filling an html form, and click on a submit button, browser generates a POST request packet. The data you filled in the form are sent to the given server-side script inside the request body in URL encoded format.
- URL: The name of server-side script (for example: bin/test.php)
- Version: HTTP/1.0 and HTTP/1.1 are available.

HTTP Request -3

What do those requests mean?

```
GET  news.php  HTTP/1.1
Host: www.ctis.bilkent.edu.tr
Accept: text/html
User-Agent: Mozilla/4.0 (compatible; MSIE 5.0; Windows 95)
If-Modified-Since: Sunday, 16-September-2010 12:27:12 GMT
```

```
GET mysite.html  HTTP/1.1
Host: www.blog.com
Accept: text/html
```

```
GET person.php?name=ali&stat=graduated  HTTP/1.1
Host: www.myschool.com
Accept: text/html
```

```
POST /search HTTP/1.1
Host: www.google.com
User-Agent: Mozilla/5.0 Galeon/1.2.5 (X11; Linux i686; U;) Gecko/20020606
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,
        text/plain;q=0.8,video/x-mng,image/png,image/jpeg,image/gif;q=0.2,
        text/css,*/*;q=0.1
Accept-Language: en
Accept-Encoding: gzip, deflate, compress;q=0.9
Accept-Charset: ISO-8859-1, utf-8;q=0.66, *;q=0.66
Keep-Alive: 300
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 31

hl=en&q=HTTP&btnG=Google+Search
```

request line

MIME type

HTTP headers

Content

HTTP Response

- Application/Web server sends the results in HTTP response packet format.
- Format:
  - Status line: version, status code, description
    - 200 OK, 404 File not Found, 301 Moved permanently
    - 302 Found (Redirect), 401 Unauthorized, 403 Forbidden
    - 500 Internal Server Error
  - HTTP headers
    - Date, Content-Type, Content-Length, Location, Server
    - Set-Cookie, WWW-Authenticate, Cache-Control, ETag
  - Content

HTTP Response Example

```
HTTP/1.0 200 OK
Content-Type: image/gif
Content-Length: 8572
Server: Test Server
        Version 1.0
```

Client requested an image file from the server. It successfully returns the image. Image is in GIF format, and its size is 8572 bytes

```
HTTP/1.1 302 Found
Date: Tue, 21 May 2002 12:34:56 GMT
Location: http://httphandbook.org/
Transfer-Encoding: chunked
Content-Type: text/html

0
```

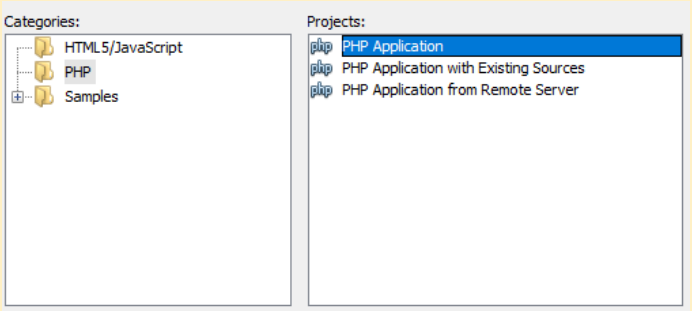
Client requested a file, but server redirects the client since the resource requested is moved to http://httphandbook.org
After getting this response, browser automatically sends another GET http packet to the given url address

Installation

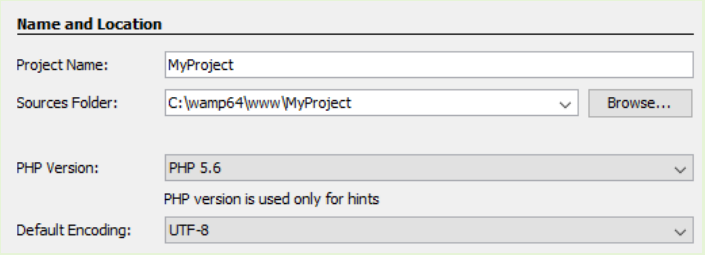
- An executable packet (WAMP for windows, LAMP for Linux, and MAMP for MacOS) is available to install and configure required software stack.
- WAMP: **W**indows **A**pache **M**ySQL **P**hp
- It installs necessary components for Web Application development (Web Server, PHP, Database).
- Store your (php, html, css, etc.) files under ***C:/wamp64/www*** folder. (Assuming C: is the installation folder)
- Access any (php,html) files using browser: ***http://localhost/path/your_file.php***
- Do not click on any php file to execute it, you have to execute php files using a browser.

Project in NetBeans

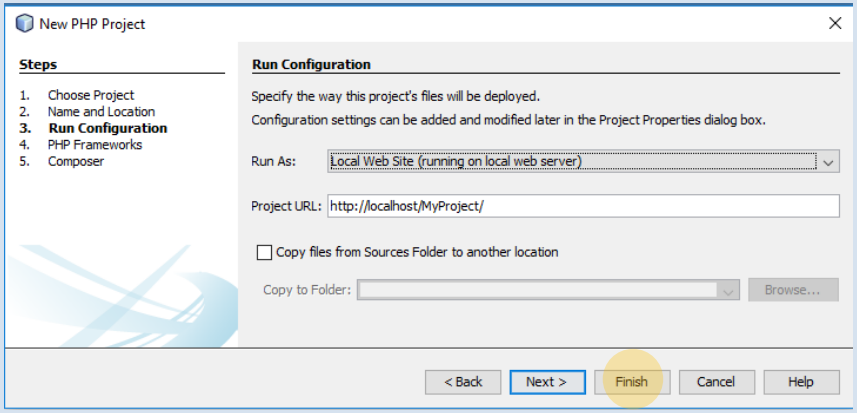
1. A new project in a ***local*** computer



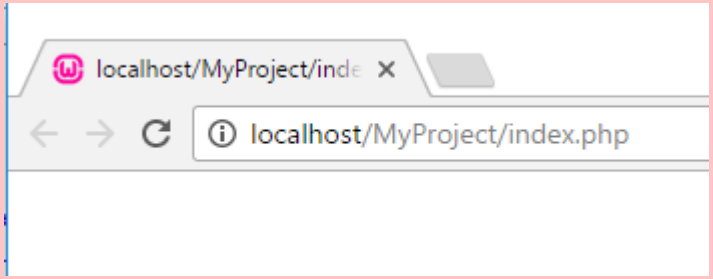
1. Select "PHP" project type and "PHP Application"



2. Create folder under "root" directory, in Windows it is "C:\wamp64\www" by default.

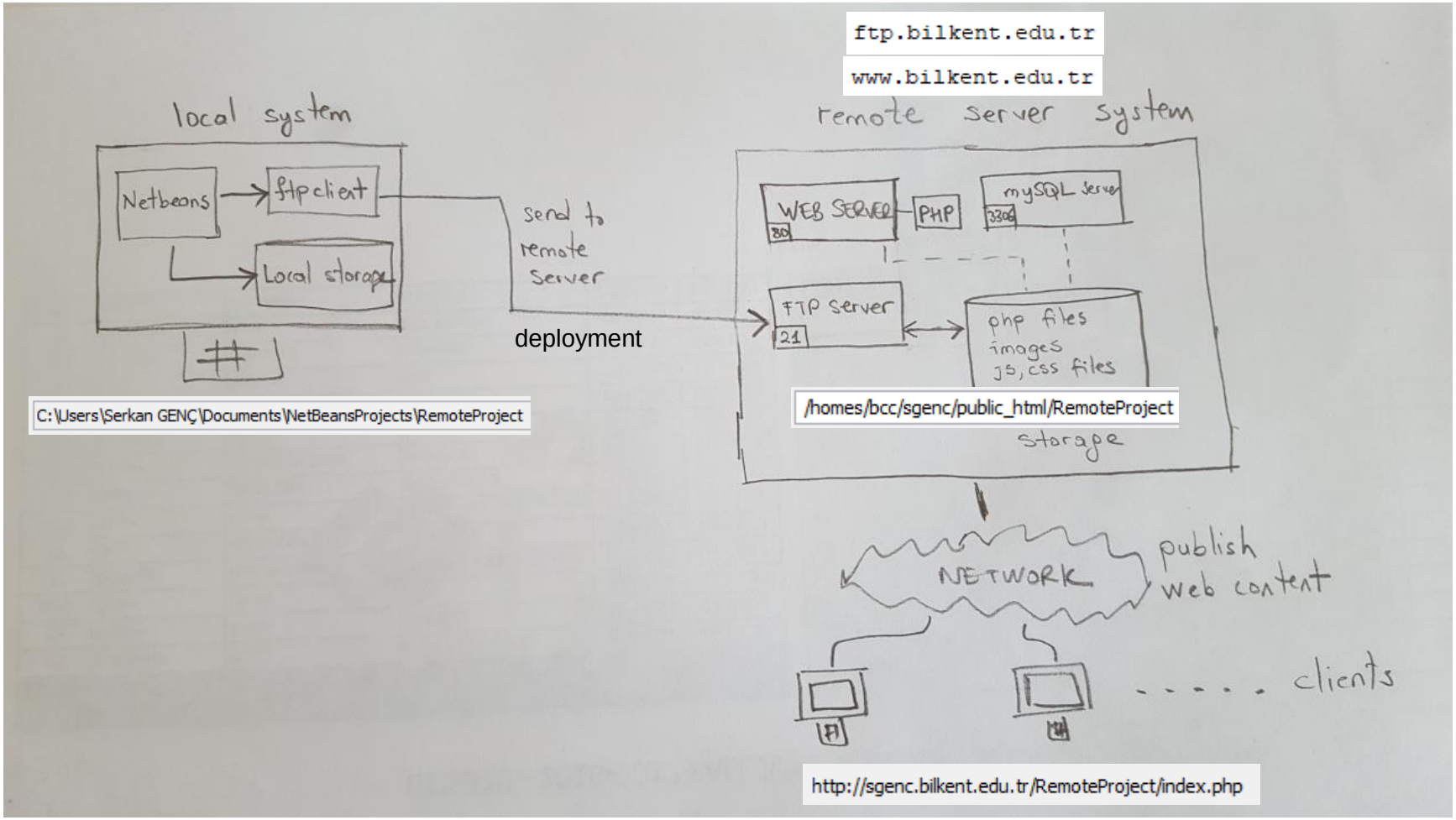


3. Create the project by "Finish"

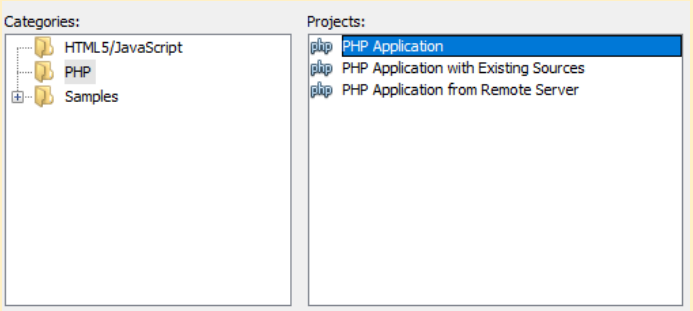


4. Run your program using a browser, in Netbeans, select the program, and "Run" in context menu.

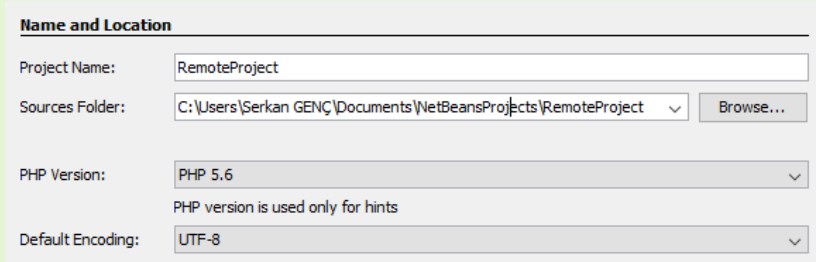
2. A NEW project in ***remote*** server



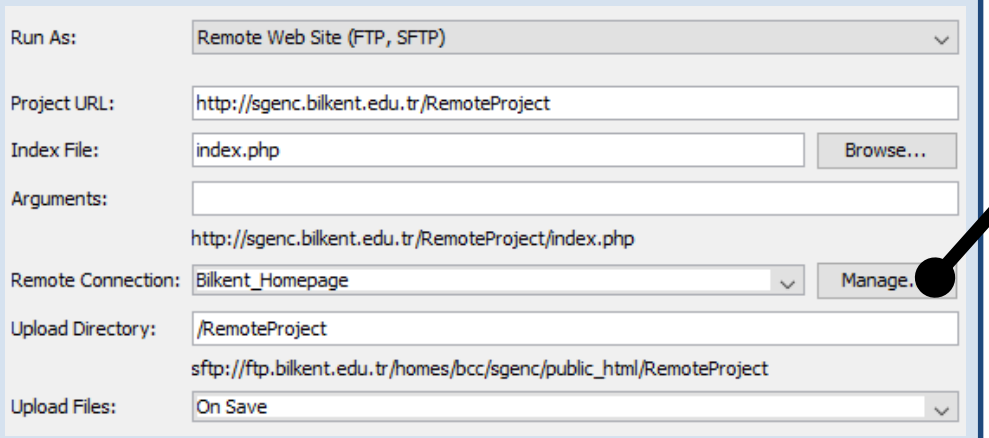
Remote server is responsible for publishing web contents. Local system produces contents and sends them to a remote server usually by ftp client. This is called deploying/uploading" a project. Local system stores the copy of the contents which is called working copy.



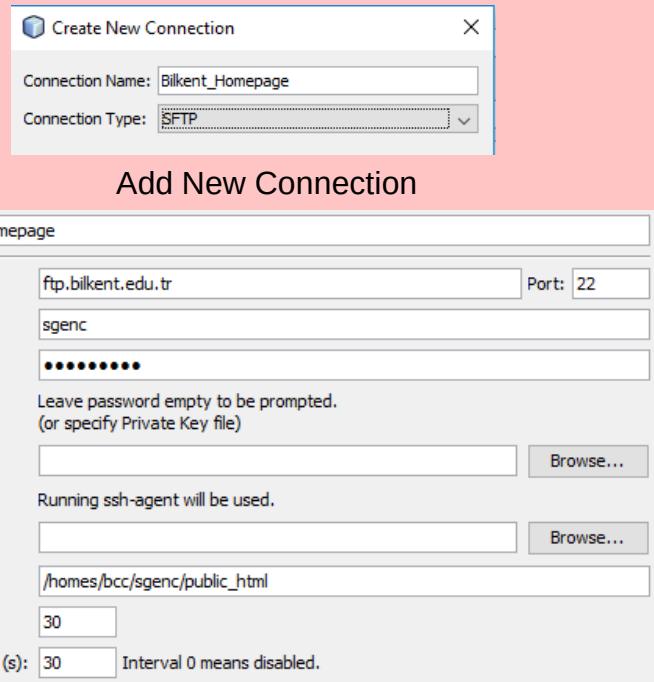
1. Select "PHP" project type and "PHP Application"



2. Create your "working copy folder" *anywhere* in your drive. and click "Finish"

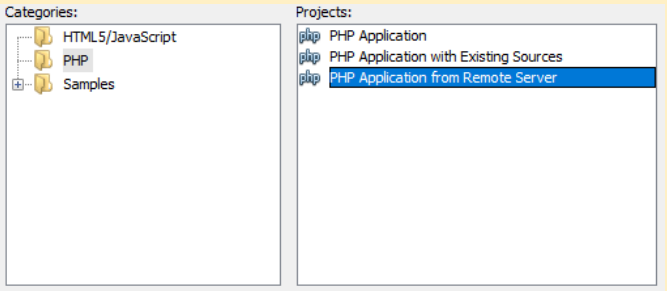


3a. Run As : "Remote Web Site", and public URL address, **Remote Connection** : create a collection method, and the upload folder at remote site, and select the way how you transfer your files (manually, On Run, and On Save)

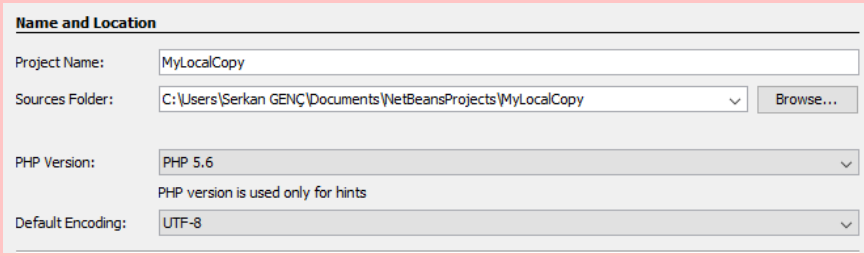


3b. Remote Connection Method. Usually using sftp. "username" and "password" are provided by "hosting" company/university.

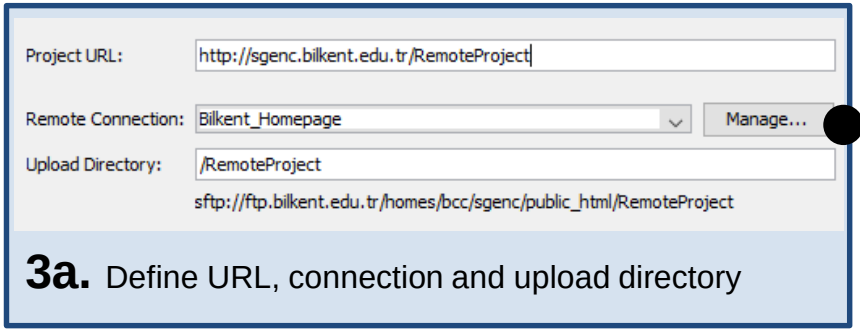
3. An existing project in ***remote*** server



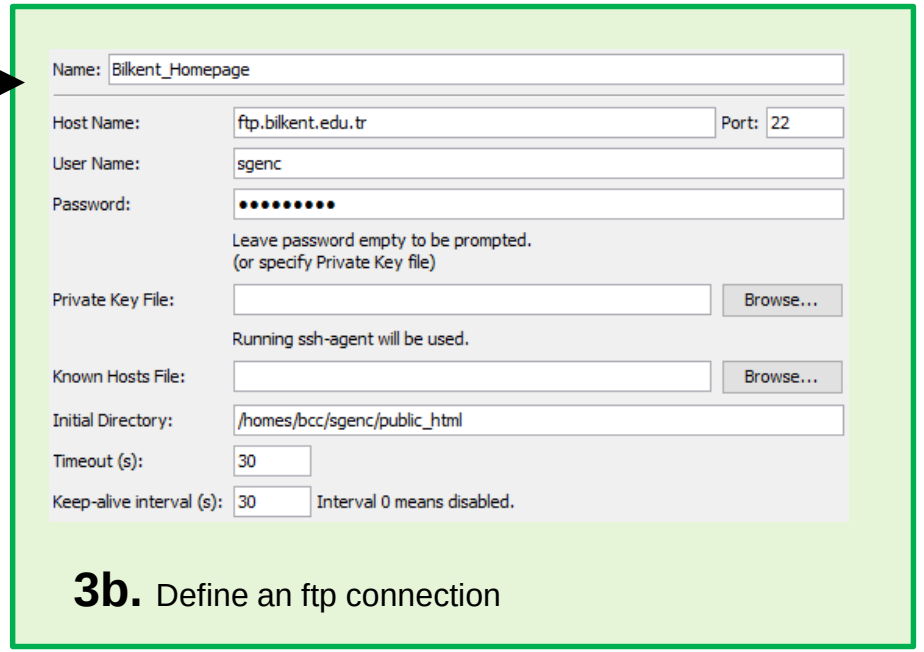
1. Select "PHP Application for Remote Server"



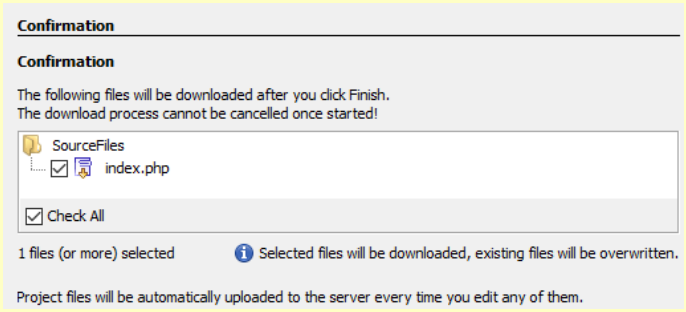
2. Select any folder for "working copy"



3a. Define URL, connection and upload directory



3b. Define an ftp connection



4. Download remote files to local project folder. Now, there are two copies, one in the local and another one is in remote.