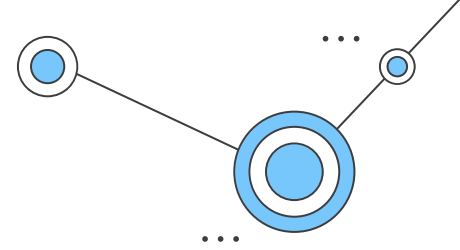


Employee Management

v1.0

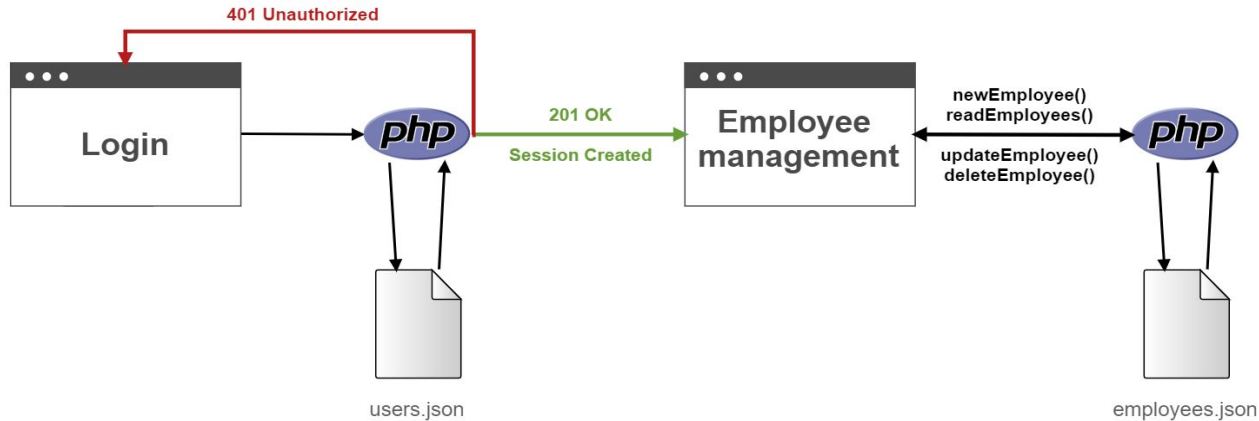
Paris Arcos
Paola Estefanía

Final Purpose



The final purpose is to be able to manage a **CRUD** of employees over a **json file**, handling encrypted **passwords**, and connecting with **external Web Services** from the **server** in order to get the employee **avatar**.

You can **only access** the operation through a **login** that will **redirect** you to the **Employees Dashboard page**. As you can see in the image, you must **store** all the information of **users and employees in JSON files**.



Main Objectives

01

\$_SESSION

Session variable knowledge

Handle session throughout all pages

02

AJAX & JSON

Learn and improve

03

Login Page

Creation and handling
encrypted data

04

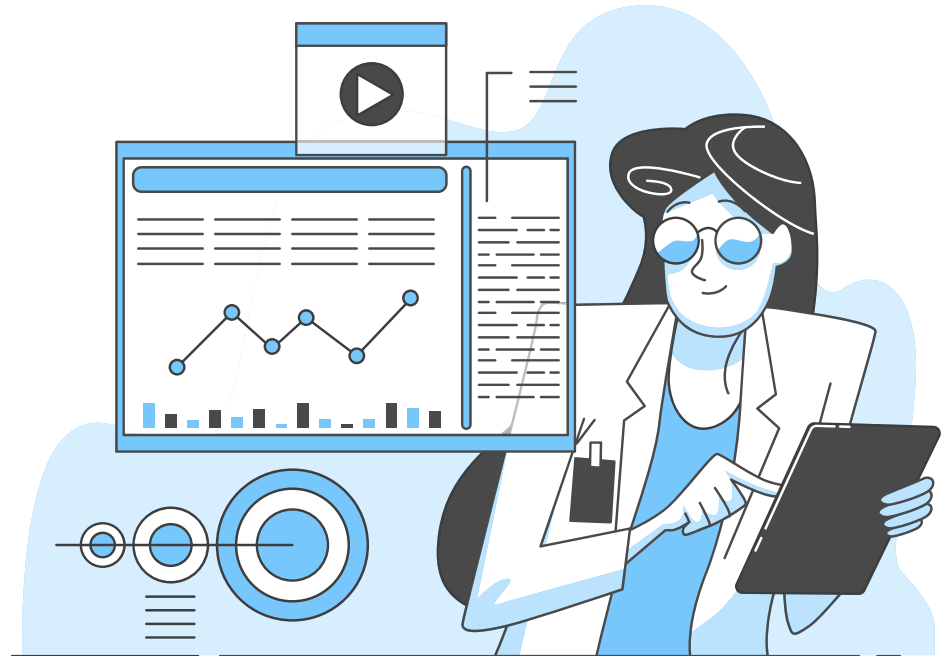
External Web Services

Connection and usage

05

Predefined Structure

How to work with it






01

\$_SESSION

What is it and how it works ?





An associative array containing session variables available to the current script.

This is a 'superglobal', or automatic global, variable. This simply means that it is available in all scopes throughout a script. There is no need to do global \$variable; to access it within functions or methods.

session_start () - Start new or resume existing session

...

...

...

...

session_abort — Discard session array changes and finish session
session_cache_expire — Get and/or set current cache expire
session_cache_limiter — Get and/or set the current cache limiter
session_commit — Alias of session_write_close
session_create_id — Create new session id
session_decode — Decodes session data from a session encoded string
session_destroy — Destroys all data registered to a session
session_encode — Encodes the current session data as a session encoded string
session_gc — Perform session data garbage collection
session_get_cookie_params — Get the session cookie parameters
session_id — Get and/or set the current session id
session_module_name — Get and/or set the current session module
session_name — Get and/or set the current session name
session_regenerate_id — Update the current session id with a newly generated one
session_register_shutdown — Session shutdown function
session_reset — Re-initialize session array with original values
session_save_path — Get and/or set the current session save path
session_set_cookie_params — Set the session cookie parameters
session_set_save_handler — Sets user-level session storage functions
session_start — Start new or resume existing session
session_status — Returns the current session status
session_unset — Free all session variables
session_write_close — Write session data and end session



02

AJAX

What is it and how it works ?



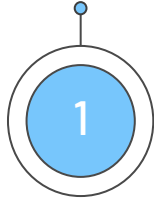
AJAX stands for **Asynchronous JavaScript And XML**. In a nutshell, it is the use of the XMLHttpRequest object to communicate with servers. It can send and receive information in various formats, including JSON, XML, HTML, and text files. AJAX's most appealing characteristic is its "asynchronous" nature, which means it can communicate with the server, exchange data, and update the page without having to refresh the page.

The two major features of AJAX allow you to do the following:

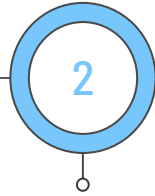
- Make requests to the server without reloading the page
- Receive and work with data from the server

How AJAX Works

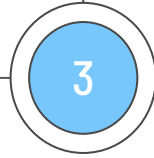
An event occurs in a web page (the page is loaded, a button is clicked)



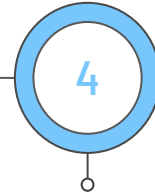
The XMLHttpRequest object sends a request to a web server



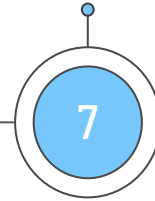
An XMLHttpRequest object is created by JavaScript



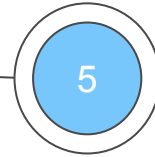
The server processes the request



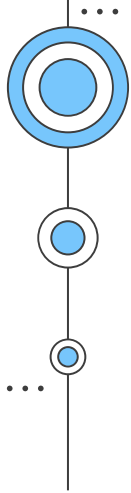
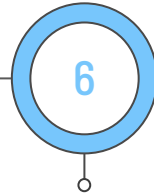
Proper action (like page update) is performed by JavaScript



The server sends a response back to the web page



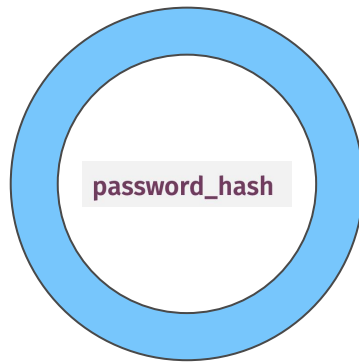
The response is read by JavaScript



03

login

Creation and handling encrypted data



-> **password_hash** — Creates a password hash

Returns the hashed password. `password_hash()` creates a new password hash using a strong one-way hashing algorithm. `password_hash()` is compatible with `crypt()`. Therefore, password hashes created by `crypt()` can be used with `password_hash()`.

The used algorithm, cost and salt are returned as part of the hash. Therefore, all information that's needed to verify the hash is included in it. This allows the `password_verify()` function to verify the hash without needing separate storage for the salt or algorithm information.

...



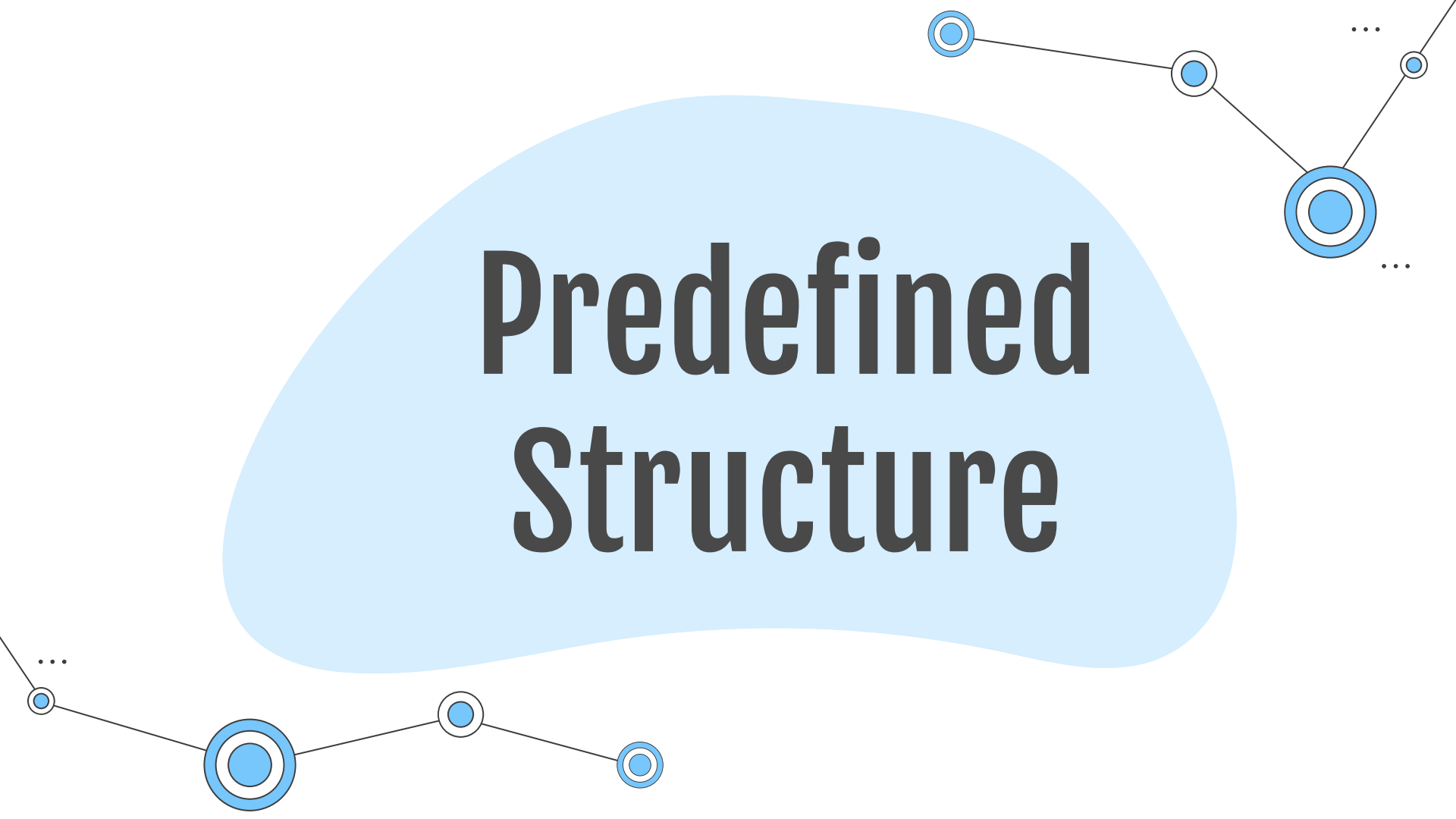
-> **password_verify** – Verifies that a password matches a hash

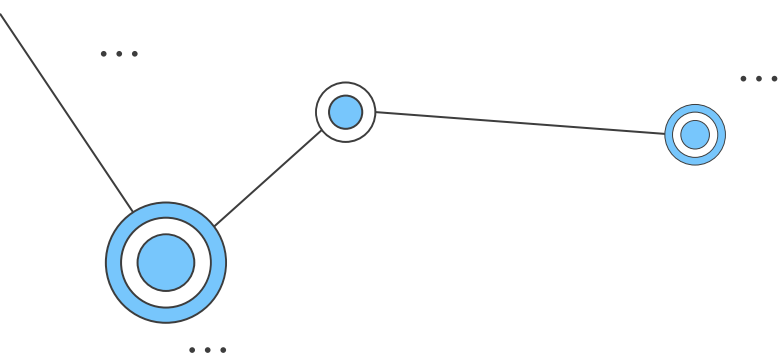
Verifies that the given hash matches the given password.

Note that `password_hash()` returns the algorithm, cost and salt as part of the returned hash. Therefore, all information that's needed to verify the hash is included in it. This allows the verify function to verify the hash without needing separate storage for the salt or algorithm information.

This function is safe against timing attacks.

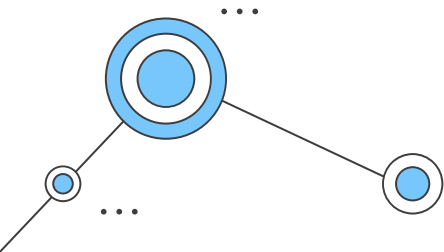
Predefined Structure





- **git**: use of words like -build -feed -fix -bug
- Js **templates** for html injection
- Documentation: **js doc y php doc**

—Helpful tools among the Project





Logic that makes everything go

01

Client View (Front)

Its where the client
Action takes place

02

AJAX request

Connection between client and server.
They communicate each other through
messages back and forward

03

Controller

Server figuring out
what to do with the
request

04

Manager (Server)

Server action, logic,
implemented. Sends
the response that will
reach the view once
again.

Understanding the Problem



Front

Implemented with
javascript,
bootstrap helped a
lot.
...



Server

Php is who makes
the magic here
...



BDD

JSON files
Will hold all the
secrets
...

Software Documentation

Implementing php y js docs

Useful for further use and to better understanding of the code



Including html & js templates

01

Include html

The code makes use
Of include for a clean
Software for the views

02

Script js template

And when the content
Its dynamic, templates
Are injected in with js

Lets see some of the front



Log in & log out

You have logged out correctly

Assembler
School of Software Engineering

admin@assemblerschool.com

.....

Sign in

© Assembler School 2021

Assembler
School of Software Engineering

Email address

Password

Sign in

© Assembler School 2021

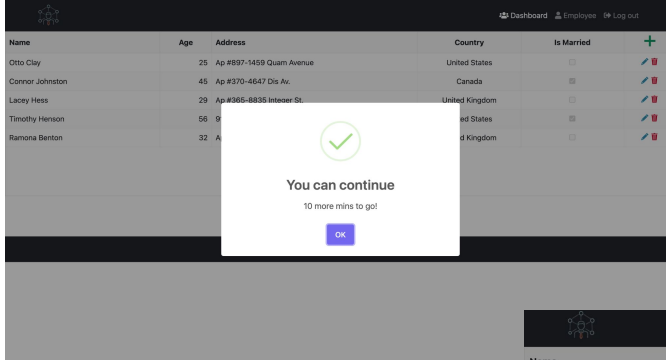
Dashboard













Dashboard Employee Log out

Name	Age	Address	Country	Is Married	+
Otto Clay	25	Ap #897-1459 Quam Avenue	United States	<input type="checkbox"/>	
Connor Johnston	45	Ap #370-4647 Dis Av.	Canada	<input checked="" type="checkbox"/>	
Lacey Hess	29	Ap #365-8835 Integer St.	United Kingdom	<input type="checkbox"/>	
Timothy Henson	56	911-5143 Luctus Ave	United States	<input checked="" type="checkbox"/>	
Ramona Benton	32	Ap #614-689 Vehicula Street	United Kingdom	<input type="checkbox"/>	

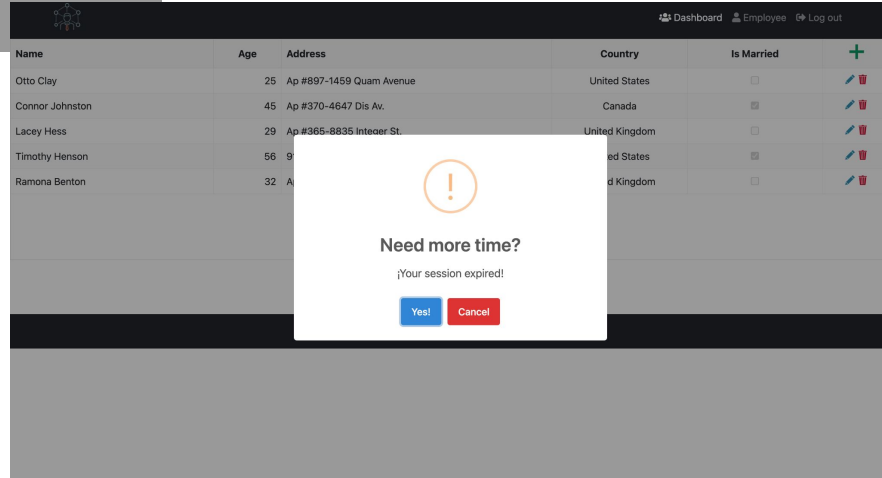
Session TimeOut













Dashboard Employee Log out

Name	Age	Address	Country	Is Married	
Otto Clay	25	Ap #897-1459 Quam Avenue	United States	<input type="checkbox"/>	 
Connor Johnston	45	Ap #970-4647 Dis Av.	Canada	<input type="checkbox"/>	 
Lacey Hess	29	Ap #965-8835 Integer St.	United Kingdom	<input type="checkbox"/>	 
Timothy Henson	56	Ap #365-8835 Integer St.	United States	<input type="checkbox"/>	 
Ramona Benton	32	Ap #365-8835 Integer St.	United Kingdom	<input type="checkbox"/>	 

You can continue
10 more mins to go!
OK

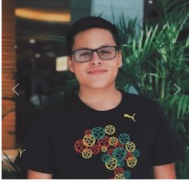


Dashboard Employee Log out

Name	Age	Address	Country	Is Married	
Otto Clay	25	Ap #897-1459 Quam Avenue	United States	<input type="checkbox"/>	 
Connor Johnston	45	Ap #370-4647 Dis Av.	Canada	<input type="checkbox"/>	 
Lacey Hess	29	Ap #365-8835 Integer St.	United Kingdom	<input type="checkbox"/>	 
Timothy Henson	56	Ap #365-8835 Integer St.	United States	<input type="checkbox"/>	 
Ramona Benton	32	Ap #365-8835 Integer St.	United Kingdom	<input type="checkbox"/>	 

Need more time?
your session expired!
Yes! Cancel

Employee



Select Employee
2 - John
Delete New

Name

John

Email

jhondoe@foo.com

City

New York

State

WA

Postal code

09889

Last Name

Doe

Gender

Male

Street Address

89

Age

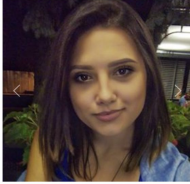
34

Phone Number

1283645645

Update

Return



Select Employee

Name

Surname

Email

Choose...

City

Street Address

State

Age

Postal code

PhoneNumber

Name

Surname

Gender

Choose...

Street Address

Street Address

Age

Age

Phone Number

PhoneNumber

Submit

Return



Select Employee

Name

Name

Email

Email

City

City

State

State

Postal code

Postal code

Last Name

Surname

Gender

Choose...

Street Address

Street Address

Age

Age

Phone Number

PhoneNumber

Submit

Return

Select Employee

✓ Choose...
1 - Rack
2 - John
3 - Leila
4 - Richard
5 - Susan
6 - Brad
7 - Neil
8 - Robert

© 2021 Copyright: Paola y Paris



Whoa!

PHP JS AJAX JSON has a lot of
to offer!



Thanks!

Do you have any questions?

CREDITS: This presentation template was created by [Slidesgo](#), including icons by [Flaticon](#), infographics & images by [Freepik](#) and illustrations by [Stories](#)

Please keep this slide for attribution



Resources

Links

- <https://www.json-generator.com/>
- <https://www.chartjs.org/docs/latest/samples/utils.html>
- <http://js-grid.com/demos/>
- <https://sweetalert2.github.io/>
- <https://readme.so/es/editor>
 - <https://gist.github.com/Villanuevand/6386899f70346d4580c723232524d35a>
- <https://uifaces.co/>