

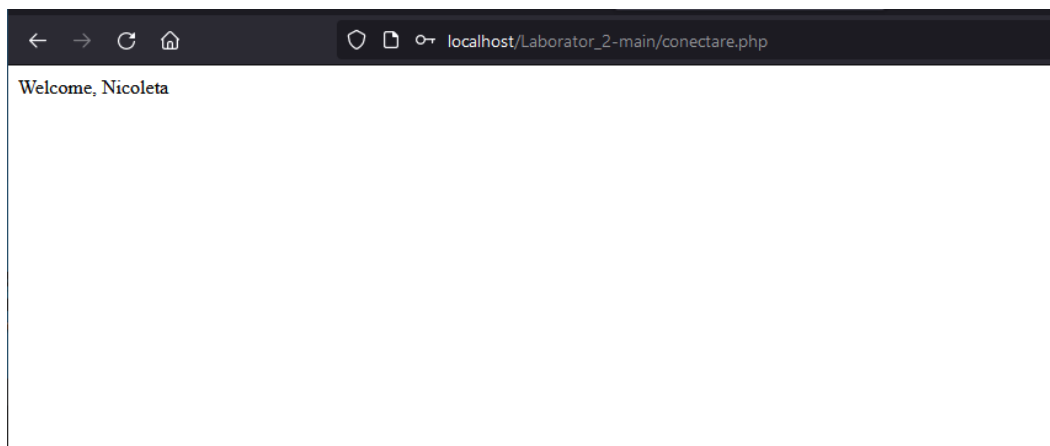
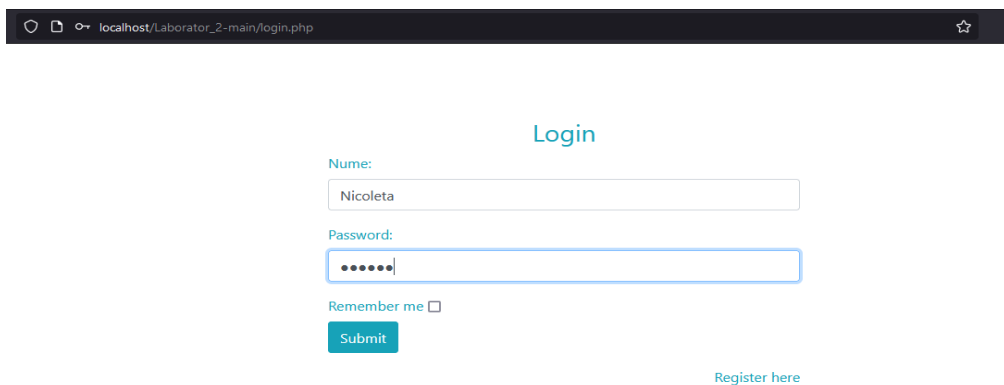
Raport de activitate

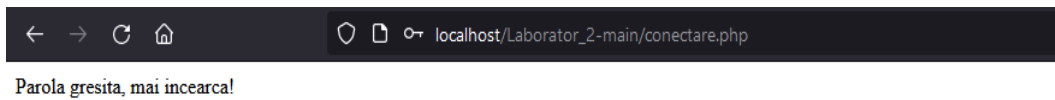
1. Incarcarea unui printscreen cu contul de GitHub



2. PHP - Incarcarea unui document cu codul care s-a rulat.

https://github.com/nicoleta-dumitru/Laborator_2-main





3. PHP+MYSQL

CRUD Metoda clasica (Create, Read, Update, Delete), folosind PHP si MYSQL.

https://github.com/nicoleta-dumitru/Laborator_3-main

-conectarea la baza de date

```
1  <?php
2
3  $dbUsername="root";
4  $dbPassword="";
5  $hostName="localhost";
6  $dbName="lab3";
7  $dbPort=3307;
8  $con=mysqli_connect($hostName,$dbUsername,$dbPassword,$dbName, $dbPort);
9
10 if(!$con){
11     $string = "Nu se poate conecta";
12     echo ("<script>console.log('".$string."');</script>");
13     die();
14 }
15 else{
16     $string = "Reusit !";
17     echo ("<script>console.log('".$string."');</script>");
18 }
19
20
21
22 ?>
23
24
```

Showing rows 0 - 0 (1 total, Query took 0.0002 seconds.)

SELECT * FROM `persoane`

☐ Profiling [\[Edit inline \]](#) [\[Edit \]](#) [\[Explain SQL \]](#) [\[Create PHP code \]](#) [\[Refresh \]](#)

☐ Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

	id	Nume	Prenume	Localitate
<input type="checkbox"/>	0	Dumitru	Nicoleta2	Baia Mare

☐ Check all | With selected: [Edit](#) [Copy](#) [Delete](#) [Export](#)

☐ Show all | Number of rows: 25 | Filter rows: Search this table

Query results operations

[Print](#) [Copy to clipboard](#) [Export](#) [Display chart](#) [Create view](#)

-adaugarea persoanelor

```
require_once("conectareDB.php");
if (isset($_POST['submit'])) {
    $query = "INSERT INTO persoane (Nume,Prenume,Localitate) VALUES ('$nume_action','$prenume_action','$localitate_action')";
    $result = mysqli_query($con, $query);
    if ($result) {
        header("location:index.php");
    } else {
        echo " can not insert into database";
    }
} else {
    header("location:index.php");
}
```

Adauga persoana

<input type="text" value="Nume"/>	<input type="text" value="Prenume"/>
<input type="text" value="Localitate"/>	<input type="button" value="Save"/>

Lista persoane

#	Nume	Prenume	Localitate		
0	Dumitru	Nicoleta	Baia Mare	Edit	Delete

-modificare persoana

```
if (isset($_POST['update'])) {
    $id = $_GET['id'];
    echo "<script> alert('" . $id . "')</script>";
    $name = $_POST['Nume'];
    $prenume = $_POST['Prenume'];
    $localitate = $_POST['Localitate'];
    echo $name, $prenume, $localitate;
    $query = "update persoane set Nume='" . $name . "',Prenume='" . $prenume . "',Localitate='" . $localitate . "'";
    $result = mysqli_query($con, $query);
    if ($result) {
        header("location:index.php");
    } else {
        echo "can not update in database";
    }
} else {
    header("location:index.php");
}
```

Modifica

Adauga persoana

Lista persoane

#	Nume	Prenume	Localitate		
0	Dumitru	Nicoleta2	Baia Mare	Edit	Delete

-stergere persoana

```
if (isset($_GET['DelID'])) {  
    $id = $_GET['DelID'];  
  
    $query = "delete from persoane where id='" . $id . "'";  
    $result = mysqli_query($con, $query);  
    if ($result) {  
        header("location:index.php");  
    } else {  
        echo " can not delete this task";  
    }  
}  
?>
```

4. MVC+LARAVEL

<https://github.com/nicoleta-dumitru/Laborator4-main>

```
8 class TodoController extends Controller
9 {
10     public function index()
11     {
12         $td = Todo::latest()->paginate(5);
13         return view('todo.index', compact('td'))->with('i', (request()->input('page', 1) - 1) * 5);
14     }
15     public function create()
16     {
17         return view('todo.create');
18     }
19     public function store(Request $request)
20     {
21         $this->validate($request, ['title' => 'required', 'des' => 'required',]);
22         echo implode(" ", $request->all());
23         Todo::create($request->all());
24         return redirect()->route('todo.index')->with('success', 'ToDo created succesfully');
25     }
26     public function show(Todo $todo)
27     {
28         return view('todo.show', compact('todo'));
29     }
30     public function edit(Todo $todo)
31     {
32         return view('todo.edit', compact('todo'));
33     }
34     public function update(Request $request, Todo $todo)
35     {
36         $request->validate(['title' => 'required', 'des' => 'required',]);
37         $todo->update($request->all());
38         return redirect()->route('todo.index')->with('success', 'ToDo updated successfully');
39     }
40     public function destroy(Todo $todo)
41     {
42         $todo->delete();
43         return redirect()->route('todo.index')->with('success', 'Blogs deleted successfully');
44     }
45 }
46
```

```
1 @extends('todo.layout');
2 @section('content')
3 <div class="row">
4     <div class="col-lg-12 margin-tb">
5         <div class="pull-left">
6             <h2>Check the list</h2>
7         </div>
8         <div class="pull-right">
9             <a class="btn btn-success" href="{{ route('todo.create') }}"> Create new ToDo</a>
10        </div>
11    </div>
12 </div>
13 @if($message=Session::get('success'))
14 <div class="alert alert-success">
15     <p>{{ $message }}</p>
16 </div>
17 @endif
18 <table class="table table-bordered">
19     <tr>
20         <th>No</th>
21         <th>Title</th>
22         <th>Description</th>
23         <th width="250px">Action</th>
24     </tr>
25     @foreach($td as $blog)
26     <tr>
27         <td>{{ ++$i }}</td>
28         <td>{{ $blog->title }}</td>
29         <td>{{ $blog->des }}</td>
30         <td>
31             <form action="{{ route('todo.destroy',$blog->id) }}" method="POST">
32                 <a class="btn btn-info" href="{{ route('todo.show',$blog->id) }}">Show</a>
33                 <a class="btn btn-primary" href="{{ route('todo.edit',$blog->id) }}">Edit</a>
34                 @csrf
35                 @method('DELETE')
36                 <button type="submit" class="btn btn-danger">Delete</button>
37             </form>
38         </td>
39     </tr>
40     @endforeach
41 </table>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

[Thu Jan 27 22:18:46 2022] 127.0.0.1:60081 Closing

```
.env | TodoController.php | create.blade.php | index.blade.php | Controller.php
resources > views > todo > create.blade.php > div.row > div.col-lg-12.margin-tb > div.pull-right > a.btn.btn-primary
1 @extends('todo.layout')
2 @section('content')
3 <div class="row">
4     <div class="col-lg-12 margin-tb">
5         <div class="pull-left">
6             <h2>Create New ToDo</h2>
7         </div>
8         <div class="pull-right">
9             <a class="btn btn-primary" href="{{ route('todo.index') }}"> Back</a>
10        </div>
11    </div>
12</div>
13@if ($errors->any())
14<div class="alert alert-danger">
15    <strong>Warning!</strong> Please check your input code<br><br>
16    <ul>
17        @foreach ($errors->all() as $error)
18            <li>{{ $error }}</li>
19        @endforeach
20    </ul>
21</div>
22@endif
23<form action="{{ route('todo.store') }}" method="POST">
24    @csrf
25    <div class="row">
26        <div class="col-xs-12 col-sm-12 col-md-12">
27            <div class="form-group">
28                <strong>Title:</strong>
29                <input type="text" name="title" class="form-control" placeholder="Title">
30            </div>
31        </div>
32        <div class="col-xs-12 col-sm-12 col-md-12">
33            <div class="form-group">
34                <strong>Description:</strong>
35                <textarea class="form-control" style="height:280px" type="text" name="des"
36                    placeholder="Description"></textarea>
37            </div>
38        </div>
39        <div class="col-xs-12 col-sm-12 col-md-12 text-center">
40            <button type="submit" class="btn btn-primary">Submit</button>
41        </div>
42    </div>
43</form>
```

Crud application

Create New ToDo

[Back](#)

Title:

Description:

[Submit](#)

Crud application

Check the list

[Create new ToDo](#)

ToDo created successfully

No	Title	Description	Action
1	Comparatur1	Lactate, carne, fructe	Show Edit Delete

Footer

Edit ToDo

[Back](#)

Title:

Description:

[Submit](#)

Crud application

Check the list

Create new ToDo

ToDo updated successfully

No	Title	Description	Action
1	Cumparaturi2	Lactate, carne, fructe multe	Show Edit Delete

Crud application

Check the list

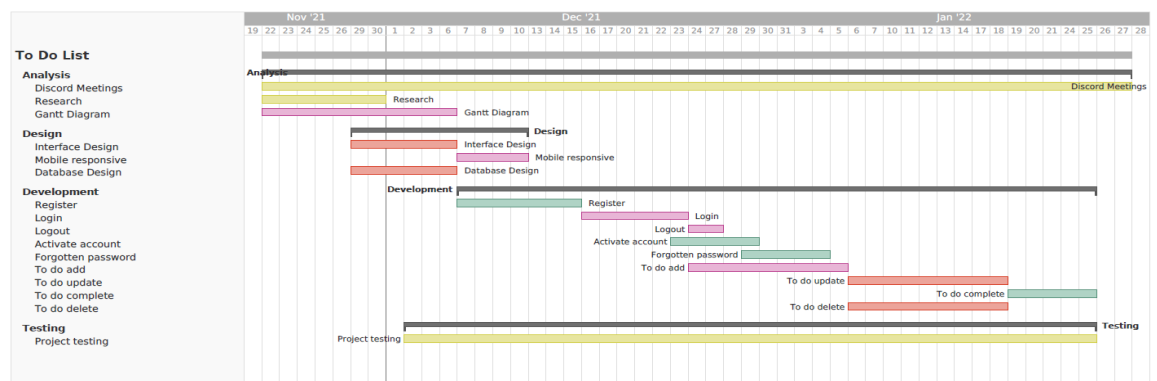
Create new ToDo

Blogs deleted successfully

No	Title	Description	Action
----	-------	-------------	--------

5. Diagrama Gantt

teamgantt
Created with Free Edition



6. To Do List

Proiect

Pentru proiectul semestrial s-a propus spre implementare o solutie software pentru gestionarea sarcinilor de lucru, dupa modelul CRUD.

Obiectivele principale ale aplicatiei sunt:

- Dezvoltarea unei interfete cat mai simplu de utilizat si prietenoasa pentru utilizator
- Stocarea sarcinilor de lucru intr-un mediu online pentru fiecare utilizator
- Vizualizarea sarcinilor de lucru
- Actualizarea sarcinilor de lucru
- Stergerea sarcinilor de lucru
- Filtrarea conturilor prin introducerea obligativitatii activarii acestora printr-o confirmare email
- Posibilitatea resetarii parolei prin confirmarea actiunii prin email

Sarcini realizate:

-Diagrama Gantt

-Mobile responsive

```
<div class="container-fluid p-3 h-100">
  <div class="row h-100">
    <div class="col-lg-6 col-sm-12 p-5 align-self-center">
      @hasSection('title')
      <h1 class="text-light" style="font-size: 64px;">@yield('title')</h1>
      @endif
      @yield('rightContent')
    </div>
    <div class="col-lg-6 col-sm-12 p-5 align-self-center">
      <div class="card p-3">
        <div class="card-body ">
          @yield('subtitle')
          @yield('content')
        </div>
      </div>
    </div>
  </div>
</div>
```

-Login

```
public function login(Request $request)
{
    $data = $request->validate([
        'email' => 'email|required',
        'password' => 'required'
    ]);
    if (!auth()->attempt($data)) {
        return response()->json(['error' => "Unauthorized"], 401);
    }
    $token = auth()->user()->createToken('API Token')->accessToken;

    return response(['user' => auth()->user(), 'token' => $token]);
}
```

-Logout

```
Route::post('/logout', function () {
    session()->remove('userSession');
    session()->remove('userId');
    return redirect('/');
});
```

-To do add

```
public function store(Request $request)
{
    $data = $request->validate([
        'email' => 'email|required',
        'password' => 'required'
    ]);
    $errors = new MessageBag();

    if (!auth()->attempt($data)) {
        $errors->add('invalid_credentials', 'Invalid email or password!');
        return view('auth.login.index')->withErrors($errors);
    }
    $user = auth()->user();
    if($user->email_verified_at == null){
        $errors->add('verification_needed', 'Please verify your account!');
        return view('auth.login.index')->withErrors($errors);
    }
    $token = $user->createToken('API Token')->accessToken;
    session()->put('userSession', $token);
    session()->put('userId', $user->id);

    return redirect('/home');
}
```