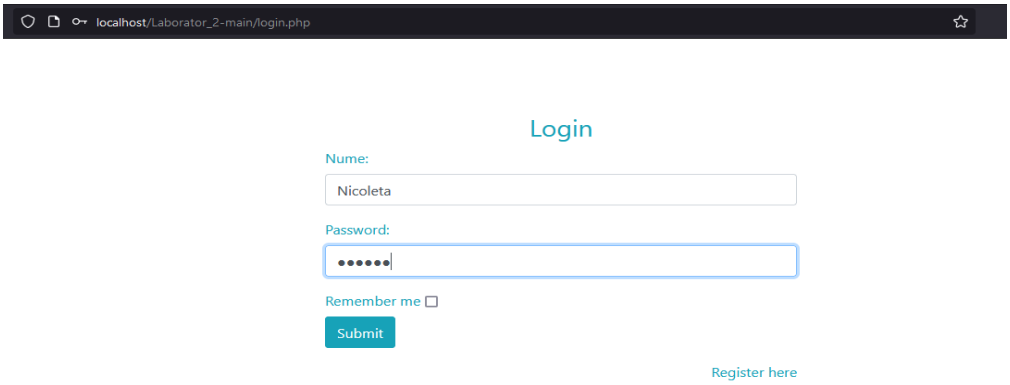
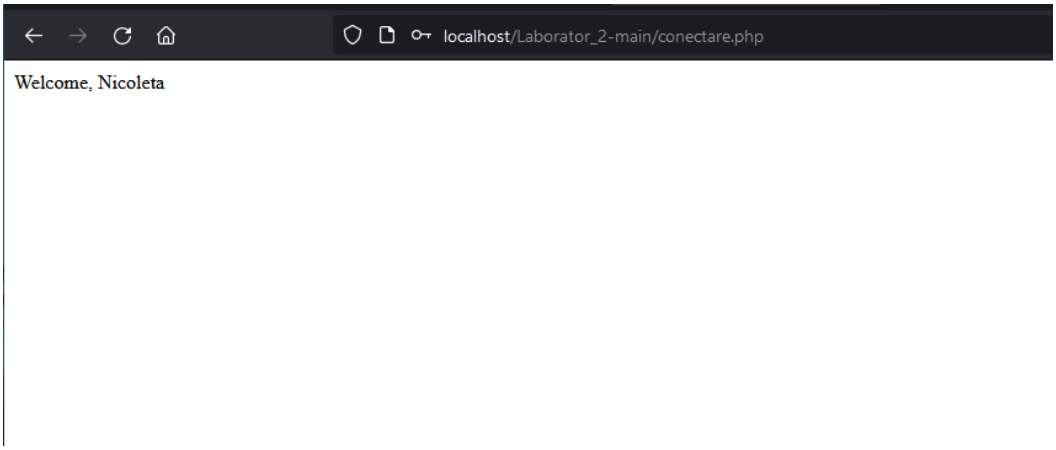
# Raport de activitate

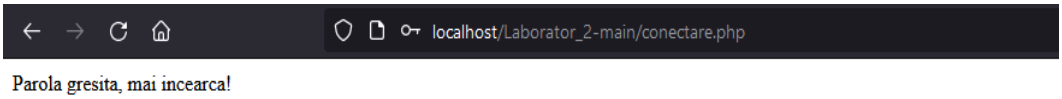
1. Incarcarea unui printscreen cu contul de GitHub


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

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





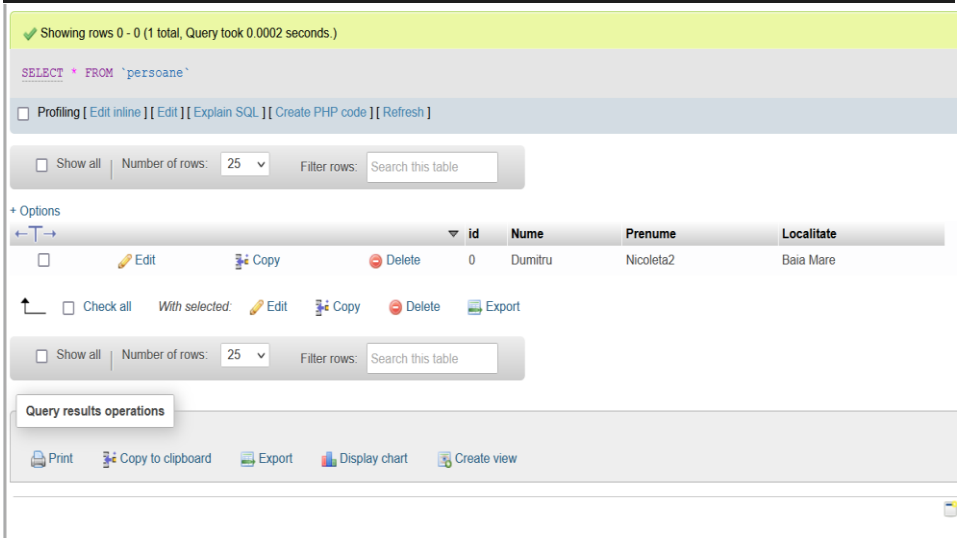

3. PHP+MYSQL

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

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

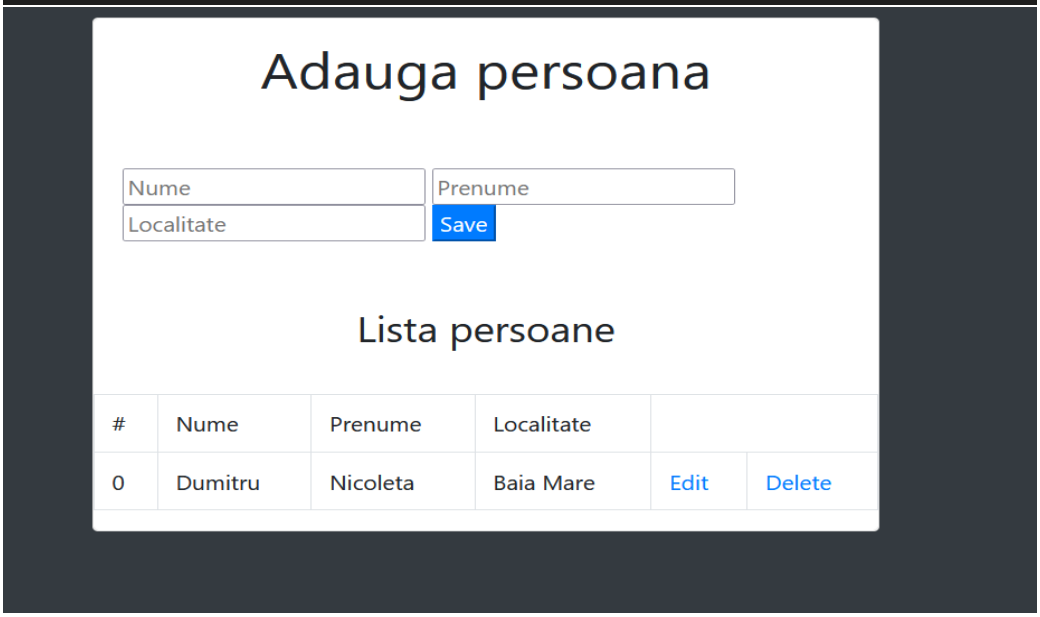
-conectarea la baza de date

```php
<?php

$dbUsername="root";
$dbPassword="";
$hostName="localhost";
$dbName="lab3";
$dbPort=3307;
$con=mysqli_connect($hostName,$dbUsername,$dbPassword,$dbName, $dbPort);

if(!$con){
    $string = "Nu se poate conecta";
    echo ("<script>console.log('".$string."');</script>");
    die();
}
else{
    $string = "Reusit !";
    echo ("<script>console.log('".$string."');</script>");
}



?>
```



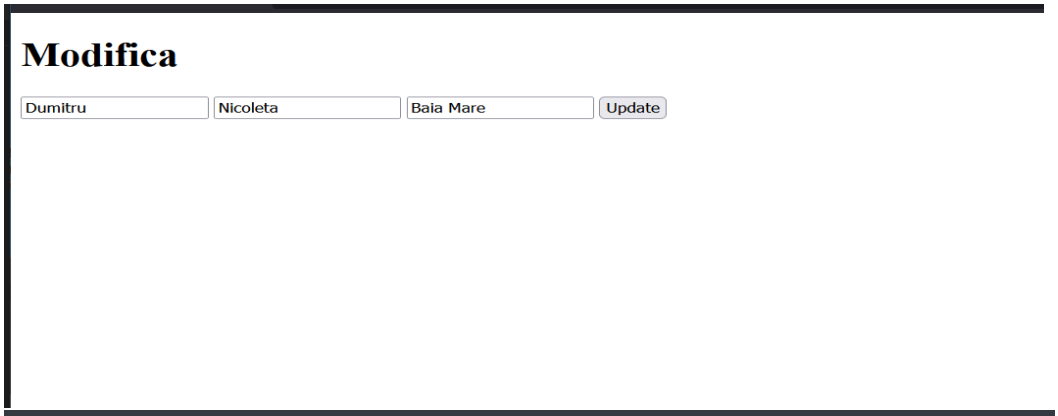
-adaugarea persoanelor

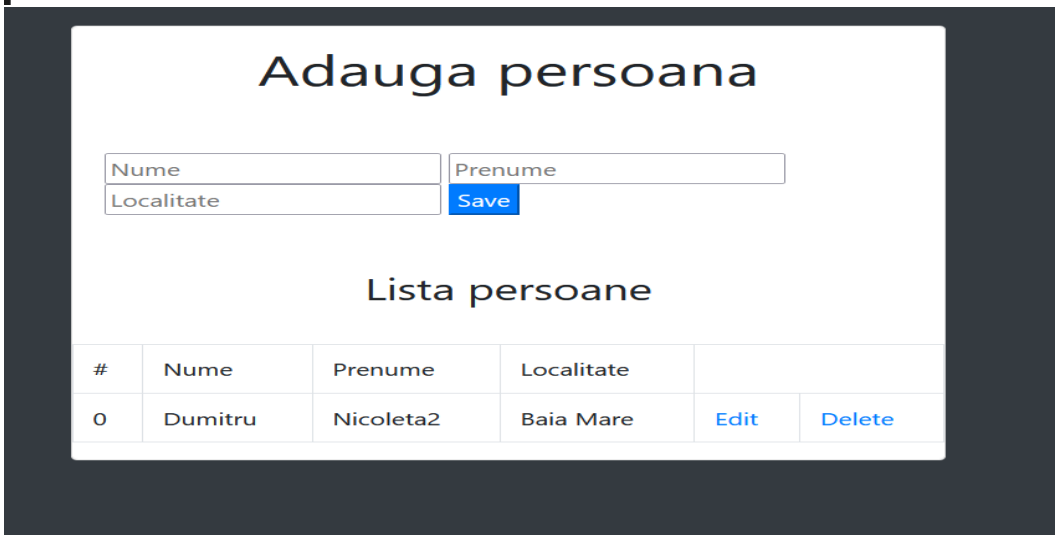
```php
require_once("conectareDB.php");
if (isset($_POST['submit'])) {
    $query = "INSERT INTO persoane (Nume,Prenume,Localitate) VALUES ('$nume_action','$prenume_action','$l
    $result = mysqli_query($con, $query);
    if ($result) {
        header("location:index.php");
    } else {
        echo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




-modificare persoana

```php
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='" . $localit
    $result = mysqli_query($con, $query);
    if ($result) {
        header("location:index.php");
    } else {
        echo "can not update in database";
    }
} else {
    header("location:index.php");
}
```




-stergere persoana

```php
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

```php
class TodoController extends Controller
{
    public function index()
    {
        $td = Todo::latest()->paginate(5);
        return view('todo.index', compact('td'))->with('i', (request()->input('page', 1) - 1) * 5);
    }
    public function create()
    {
        return view('todo.create');
    }
    public function store(Request $request)
    {
        $this->validate($request, ['title' => 'required', 'des' => 'required',]);
        echo implode(" ", $request->all());
        Todo::create($request->all());
        return redirect()->route('todo.index')->with('success', 'ToDo created succesfully');
    }
    public function show(Todo $todo)
    {
        return view('todo.show', compact('todo'));
    }
    public function edit(Todo $todo)
    {
        return view('todo.edit', compact('todo'));
    }
    public function update(Request $request, Todo $todo)
    {
        $request->validate(['title' => 'required', 'des' => 'required',]);
        $todo->update($request->all());
        return redirect()->route('todo.index')->with('success', 'ToDo updated successfully');
    }
    public function destroy(Todo $todo)
    {
        $todo->delete();
        return redirect()->route('todo.index')->with('success', 'Blogs deleted successfully');
    }
}
```

```
.env  TodoController.php  create.blade.php  show.blade.php  index.blade.php
resources > views > todo > index.blade.php > table.table.table-bordered > tr > td > form
```

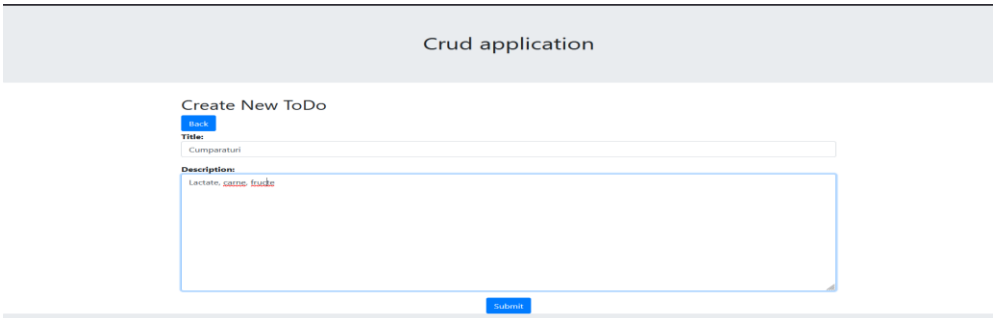
```blade
@extends('todo.layout');
@section('content')
<div class="row">
    <div class="col-lg-12 margin-tb">
        <div class="pull-left">
            <h2>Check the list</h2>
        </div>
        <div class="pull-right">
            <a class="btn btn-success" href="{{ route('todo.create') }}"> Create new ToDo</a>
        </div>
    </div>
</div>
@if($message=Session::get('success'))
<div class="alert alert-success">
    <p>{{ $message }}</p>
</div>
@endif
<table class="table table-bordered">
    <tr>
        <th>No</th>
        <th>Title</th>
        <th>Description</th>
        <th width="250px">Action</th>
    </tr>
    @foreach($td as $blog)
    <tr>
        <td>{{ ++$i }}</td>
        <td>{{ $blog->title}}</td>
        <td>{{ $blog->des}}</td>
        <td>
            <form action="{{ route('todo.destroy',$blog->id) }}" method="POST">
                <a class="btn btn-info" href="{{ route('todo.show',$blog->id) }}">Show</a>
                <a class="btn btn-primary" href="{{ route('todo.edit',$blog->id) }}">Edit</a>
                @csrf
                @method('DELETE')
                <button type="submit" class="btn btn-danger">Delete</button>
            </form>
        </td>
    </tr>
    @endforeach
</table>
```

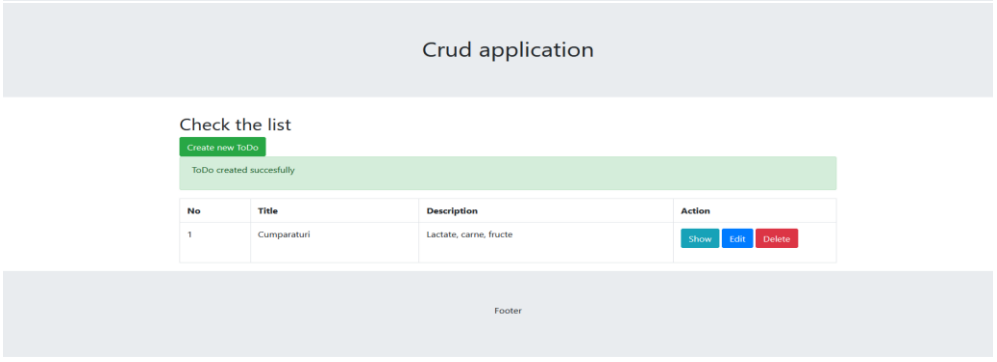
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
[Thu Jan 27 22:18:46 2022] 127.0.0.1:60081 Closing
php
```

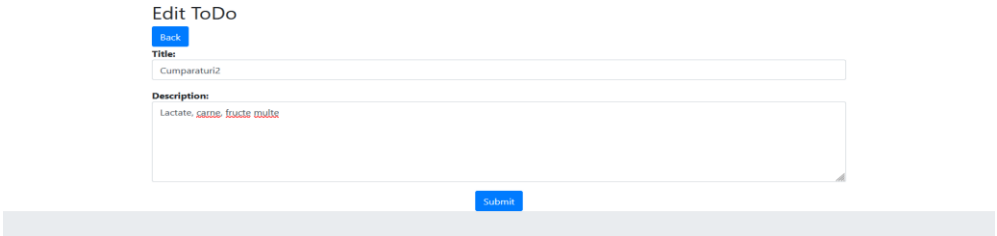
.env | TodoController.php | create.blade.php | index.blade.php | Controller.php

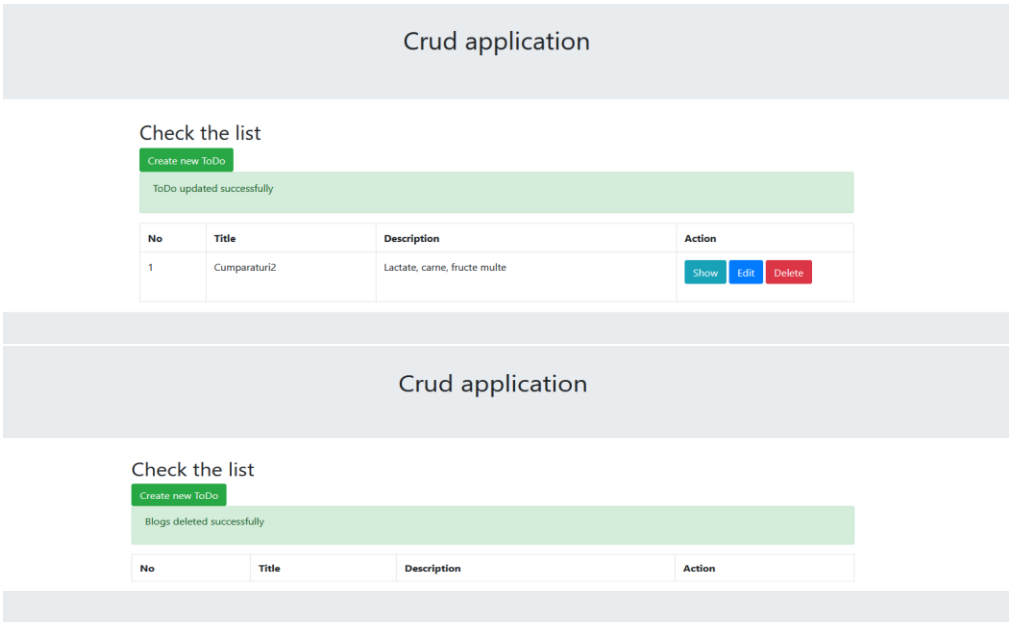
resources > views > todo > create.blade.php > div.row > div.col-lg-12.margin-tb > div.pull-right > a.btn.btn-primary

```blade
@extends('todo.layout')
@section('content')
<div class="row">
    <div class="col-lg-12 margin-tb">
        <div class="pull-left">
            <h2>Create New ToDo</h2>
        </div>
        <div class="pull-right">
            <a class="btn btn-primary" href="{{ route('todo.index') }}"> Back</a>
        </div>
    </div>
</div>
@if ($errors->any())
<div class="alert alert-danger">
    <strong>Warning!</strong> Please check your input code<br><br>
    <ul>
        @foreach ($errors->all() as $error)
        <li>{{ $error }}</li>
        @endforeach
    </ul>
</div>
@endif
<form action="{{ route('todo.store') }}" method="POST">
    @csrf
    <div class="row">
        <div class="col-xs-12 col-sm-12 col-md-12">
            <div class="form-group">
                <strong>Title:</strong>
                <input type="text" name="title" class="form-control" placeholder="Title">
            </div>
        </div>
        <div class="col-xs-12 col-sm-12 col-md-12">
            <div class="form-group">
                <strong>Description:</strong>
                <textarea class="form-control" style="height:280px" type="text" name="des"
                    placeholder="Description"></textarea>
            </div>
        </div>
        <div class="col-xs-12 col-sm-12 col-md-12 text-center">
            <button type="submit" class="btn btn-primary">Submit</button>
        </div>
```

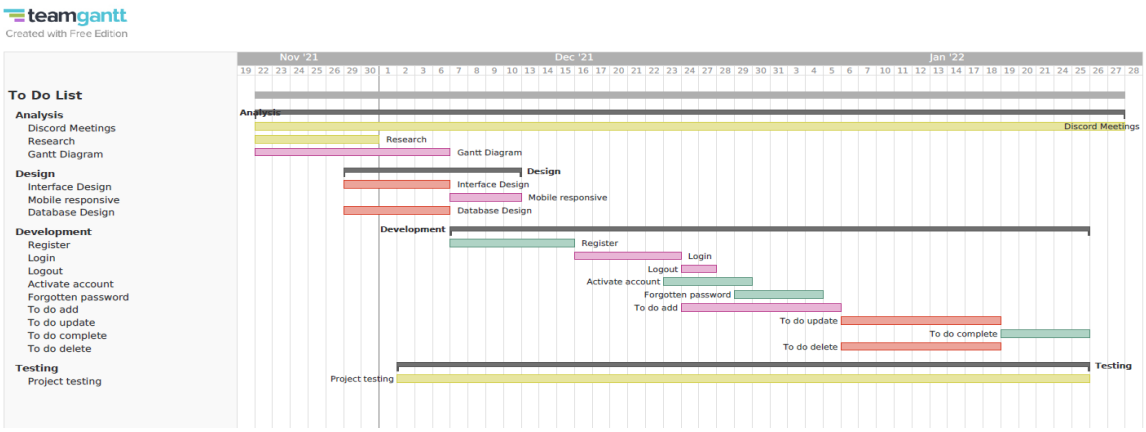





5. Diagrama Gantt




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 pin introducerea obligativitatii activarii acestora printr-o confirmare email
- Posibilitatea resetarii parolei prin confirmarea actiunii prin email

Sarcini realizate:
-Diagrama Gantt
-Mobile responsive

```blade
<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

```php
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

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

-To do add

```php
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');

}
```