Application code documentation

1. src/App/add.php

Add.php is a script for saving employee information from a form into a database.

The code processes a POST request containing employee information, saves the employee and their skills, and logs the activity accordingly. It also sets a session message for display in a template after redirecting.

// Includes various PHP files containing classes and functions needed for this script.

```
include_once '../Helpers/EmployeeManager.php';
include_once '../Config/config.php';
include_once '../Config/DBManager.php';
include_once 'Employee.php';
include_once 'EmployeeSkills.php';
include_once '../Logs/Traits/LogActivity.php';
include_once '../Validations/AddEmployeeValidations.php';
include_once '../Validations/AddEmployeeSkillsValidations.php';
```

// Declares namespaces and uses specific classes within those namespaces in the script.

```
use TANGENT\App\Employee;
use TANGENT\App\EmployeeSkills;
use TANGENT\Config\DBManager;
use TANGENT\Helpers\EmployeeManager;
use TANGENT\Logs\Traits\LogActivity;
```

// **saveEmployee** takes an employee and their skills as parameters and saves them using the **EmployeeManager** class.

```
function saveEmployee($employee, $skills)
{
    $results = EmployeeManager::addEmployee('employees', $employee);

    if($results && !empty($skills)) {
        if(DBManager::getInstance()->exists('employees', ['id'  => $employee-
>getID(),])) {
            $employeeSkills = new EmployeeSkills (
                $employee->getID(),
                json_encode($skills),
                ADMIN,
                NOW
            );
            EmployeeManager::addEmployeeSkills('employee_skills',
$employeeSkills);
        }
    }
    return $results;
}
```

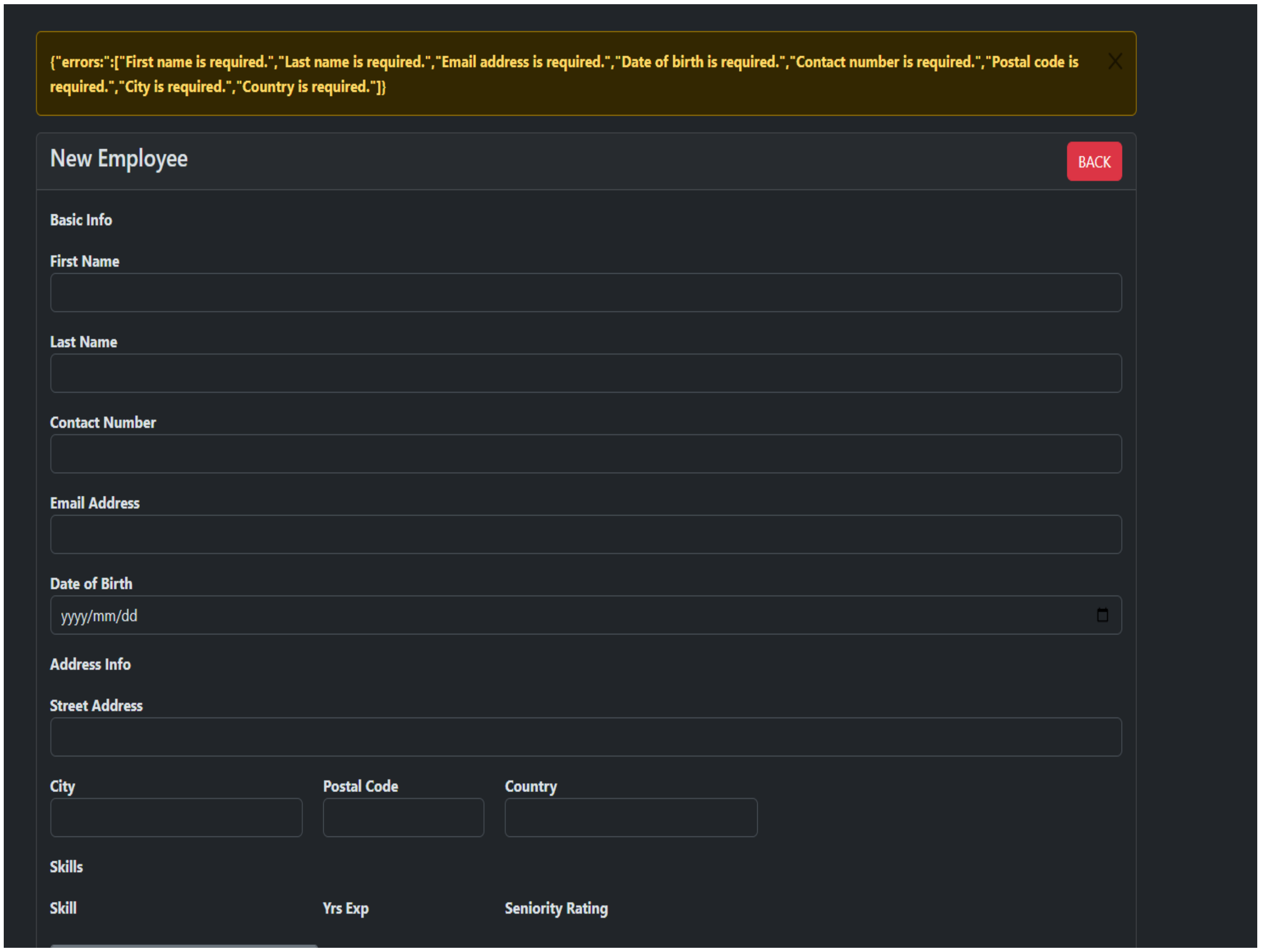
// Checks if the HTTP request method is POST and proceeds to process the employee and their skills submitted through a form.

```
if ($_SERVER['REQUEST_METHOD'] === 'POST') {


if ($_SERVER['REQUEST_METHOD'] === 'POST') {
```

// Creating Employee and Skills Objects:

```
employee = new Employee(EMPLOYEE_ID,
    json_encode($_POST["seniority_rating_id"]),
    json_encode($_POST["year_exp"]),
    $_POST["first_name"],
    $_POST["last_name"],
    $_POST["contact_number"],
    $_POST["email_address"],
    $_POST["date_of_birth"],
    $_POST["street_address"],
    $_POST["city"],
    $_POST["postal_code"],
    $_POST["country"],
    ADMIN,
    NOW
);

$skills = $_POST["skills"];


$results = saveEmployee($employee, $skills);

if ($results === true)
{
    LogActivity::logActivity('Employee Created Successfully by admin
'.ADMIN);
    $_SESSION['message'] = "Employee Created Successfully";
    header('Location: ../resources/templates/add.php');
    exit(0);
}
else
{
    LogActivity::logActivity('Failed to creat employee');
    $_SESSION['message'] = json_encode(['errors:' => $results]);
    header('Location: ../resources/templates/add.php');
    exit(0);
}
```

1.1 validate required fields



```
{"errors:":["First name is required.","Last name is required.","Email address is required.","Date of birth is required.","Contact number is required.","Postal code is required.","City is required.","Country is required."]}
```

New Employee

BACK

Basic Info

First Name

Last Name

Contact Number

Email Address

Date of Birth

yyyy/mm/dd

Address Info

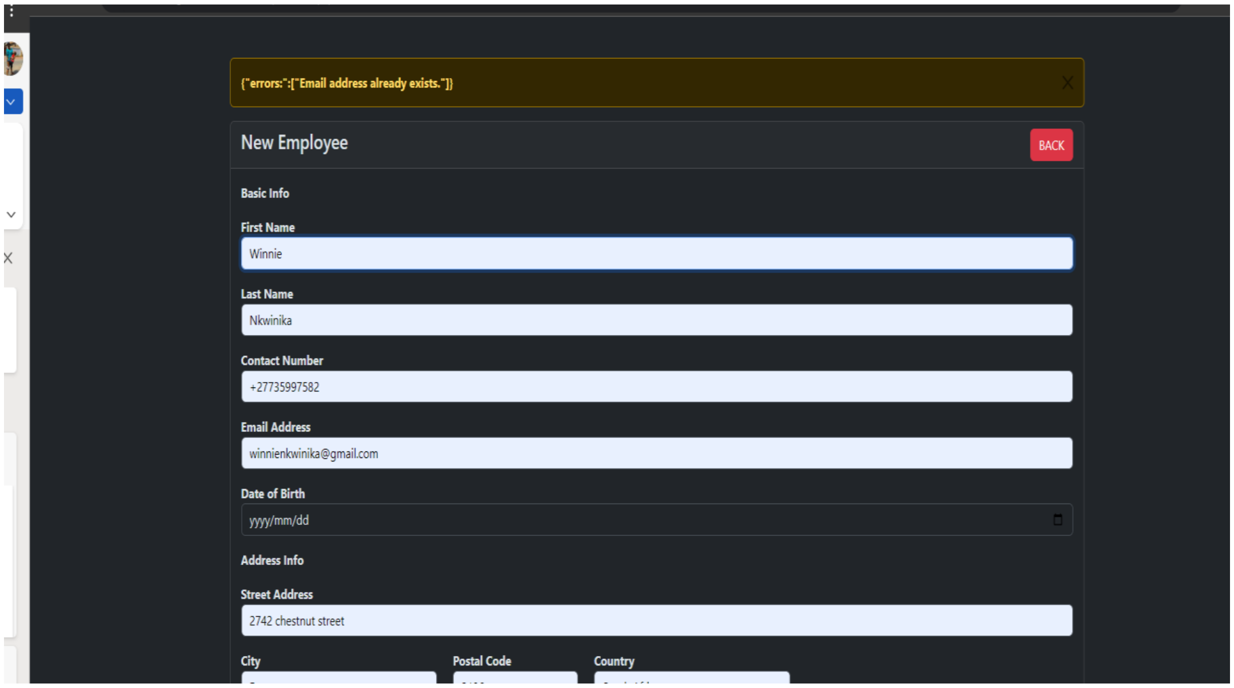
Street Address

City | Postal Code | Country

Skills

Skill | Yrs Exp | Seniority Rating

1.2 make sure that employee have unique email



{"errors":["Email address already exists."]}

New Employee

Basic Info

First Name: Winnie

Last Name: Nkwinika

Contact Number: +27735997582

Email Address: winnienkwinika@gmail.com

Date of Birth: yyyy/mm/dd

Address Info

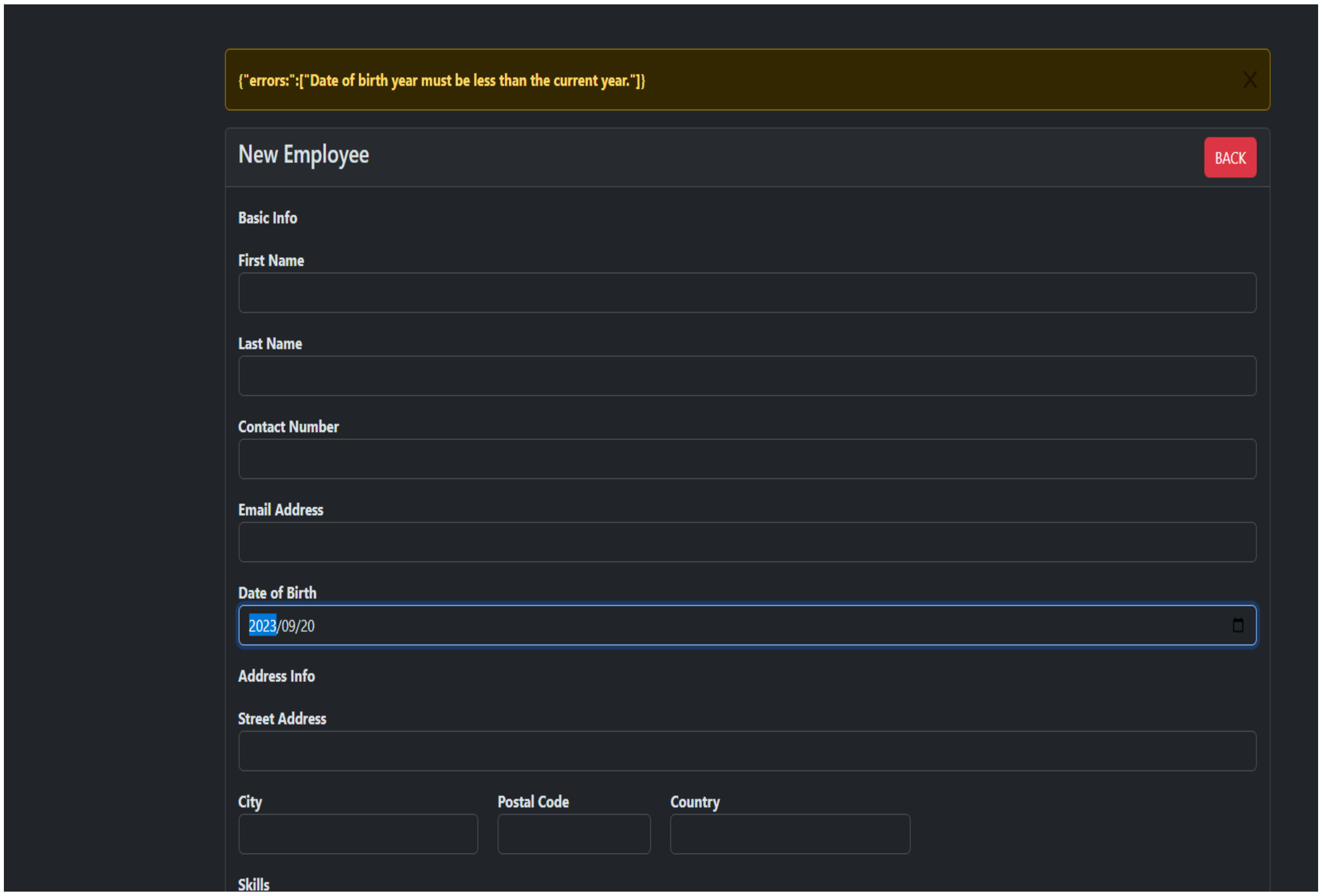
Street Address: 2742 chestnut street

City

Postal Code

Country

1.3 it checks if date of birth is current year of future year



{"errors:":["Date of birth year must be less than the current year."]}

New Employee

BACK

Basic Info

First Name

Last Name

Contact Number

Email Address

Date of Birth

2023/09/20

Address Info

Street Address

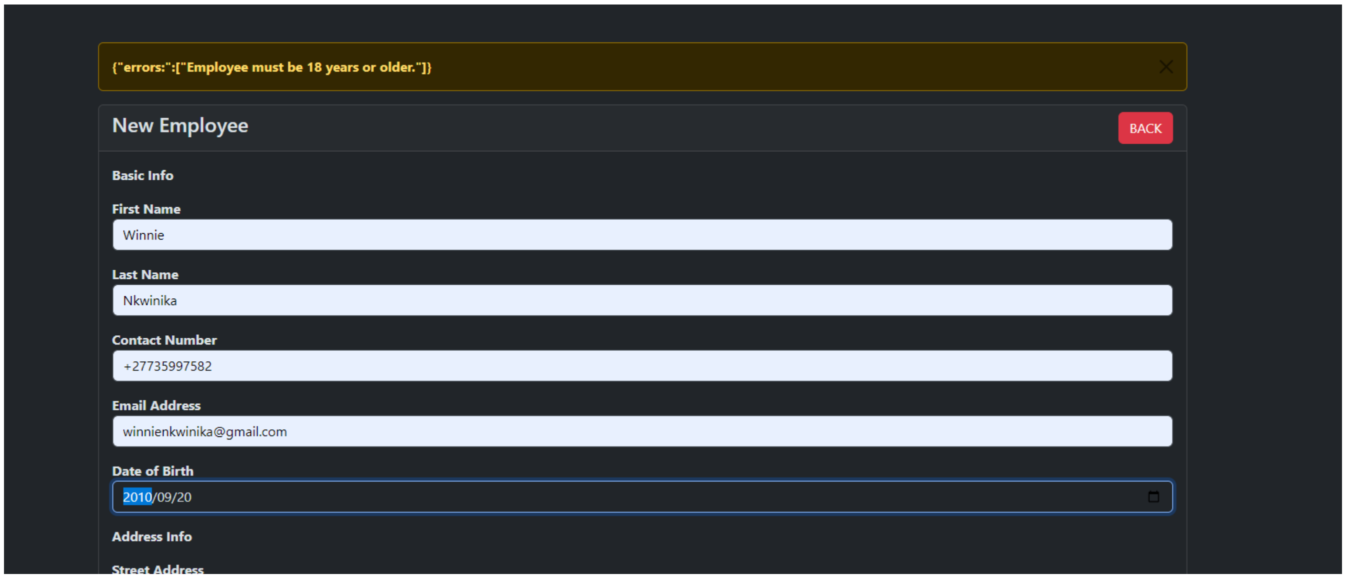
City

Postal Code

Country

Skills

1.4 checks if the employee is at the right age to work



{"errors:":["Employee must be 18 years or older."]}

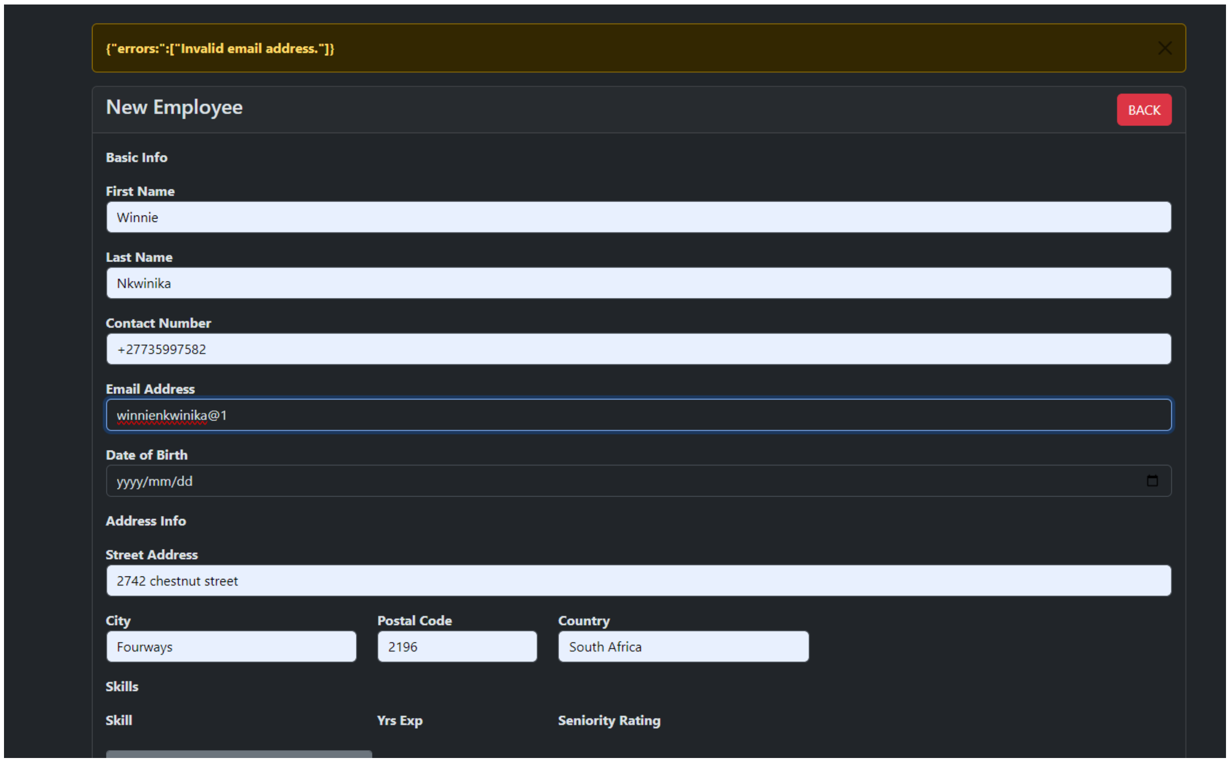
New Employee

Basic Info

First Name	Winnie
Last Name	Nkwinika
Contact Number	+27735997582
Email Address	winnienkwinika@gmail.com
Date of Birth	2010/09/20

Address Info

Street Address



{"errors:":["Invalid email address."]}

New Employee

Basic Info

First Name	Winnie
Last Name	Nkwinika
Contact Number	+27735997582
Email Address	winnienkwinika@1
Date of Birth	yyyy/mm/dd

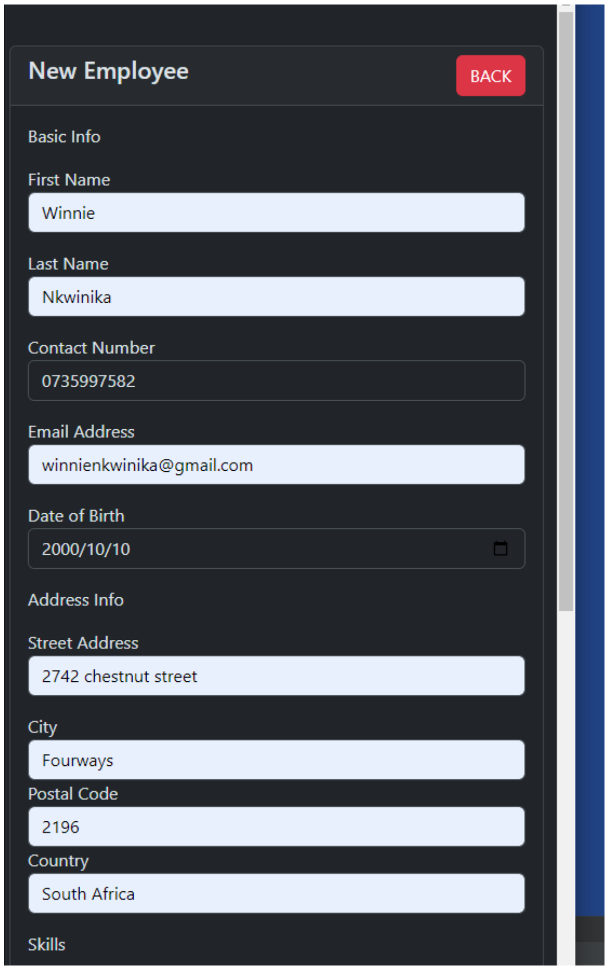
Address Info

Street Address	2742 chestnut street
City	Fourways
Postal Code	2196
Country	South Africa

Skills

Skill	Yrs Exp	Seniority Rating

App employee user interface



New Employee

BACK

Basic Info

First Name
Winnie

Last Name
Nkwinika

Contact Number
0735997582

Email Address
winnienkwinika@gmail.com

Date of Birth
2000/10/10

Address Info

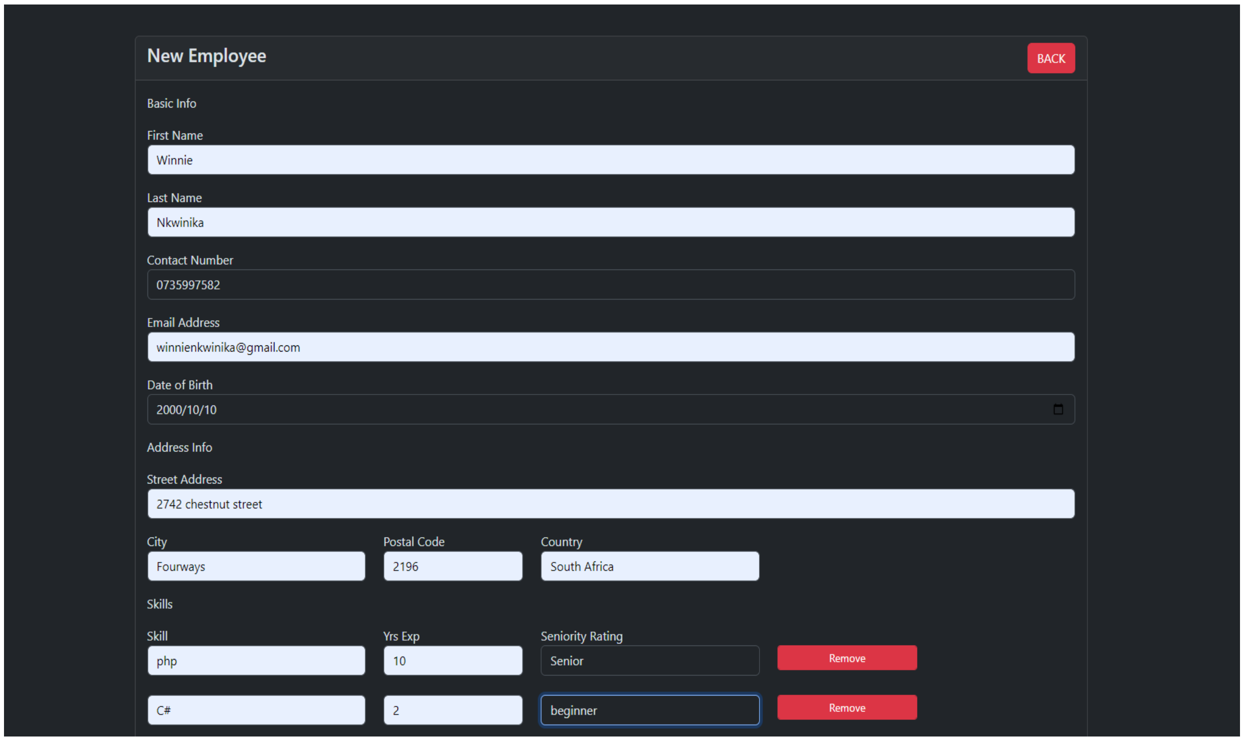
Street Address
2742 chestnut street

City
Fourways

Postal Code
2196

Country
South Africa

Skills


New Employee

BACK

Basic Info

First Name
Winnie

Last Name
Nkwinika

Contact Number
0735997582

Email Address
winnienkwinika@gmail.com

Date of Birth
2000/10/10

Address Info

Street Address
2742 chestnut street

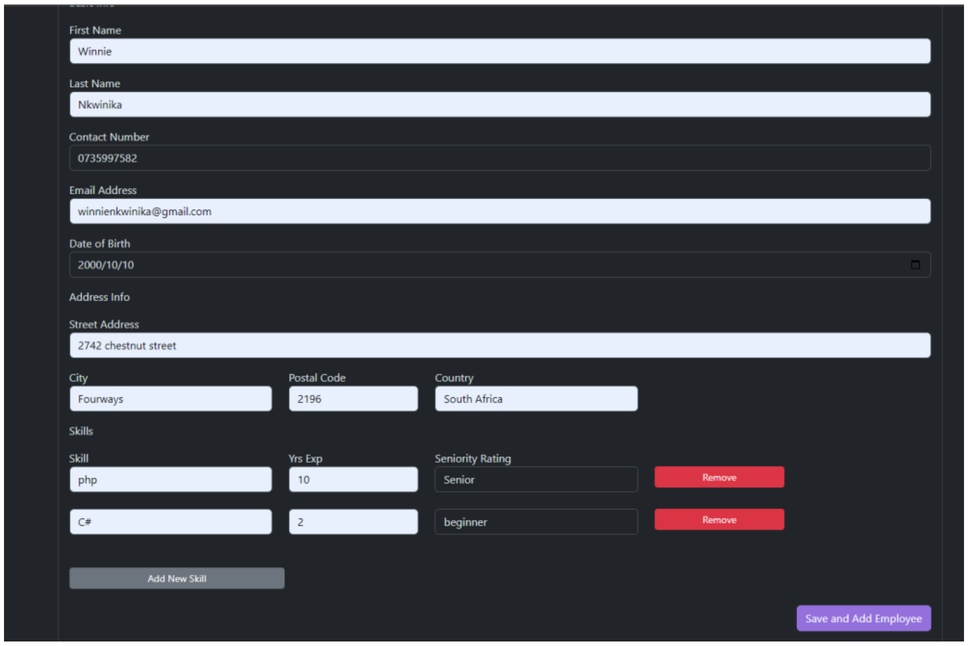
City
Fourways

Postal Code
2196

Country
South Africa

Skills

Skill	Yrs Exp	Seniority Rating	
php	10	Senior	Remove
C#	2	beginner	Remove


First Name
Winnie

Last Name
Nkwinika

Contact Number
0735997582

Email Address
winnienkwinika@gmail.com

Date of Birth
2000/10/10

Address Info

Street Address
2742 chestnut street

City
Fourways

Postal Code
2196

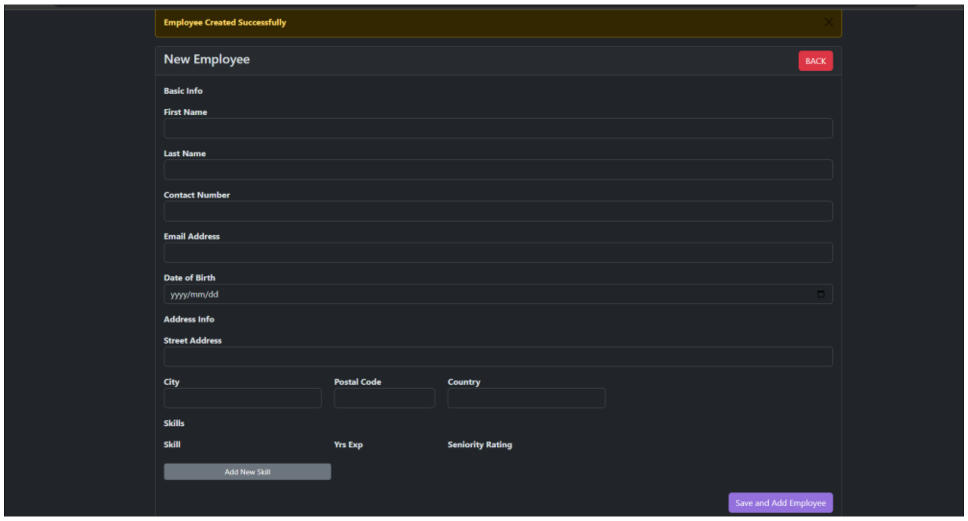
Country
South Africa

Skills

Skill	Yrs Exp	Seniority Rating	
php	10	Senior	Remove
C#	2	beginner	Remove

Add New Skill

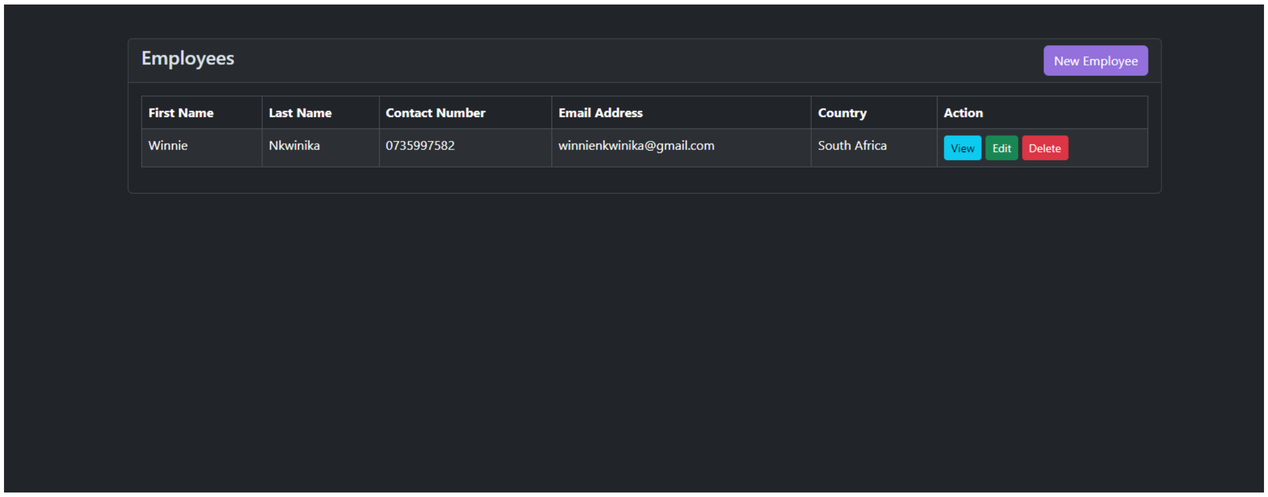
Save and Add Employee



2 list of employees

the script fetches employee data using **EmployeeManager::getAllEmployees()** and displays it in a tabular format. It provides actions to view, edit, and delete employees, and handles cases where no records are found.

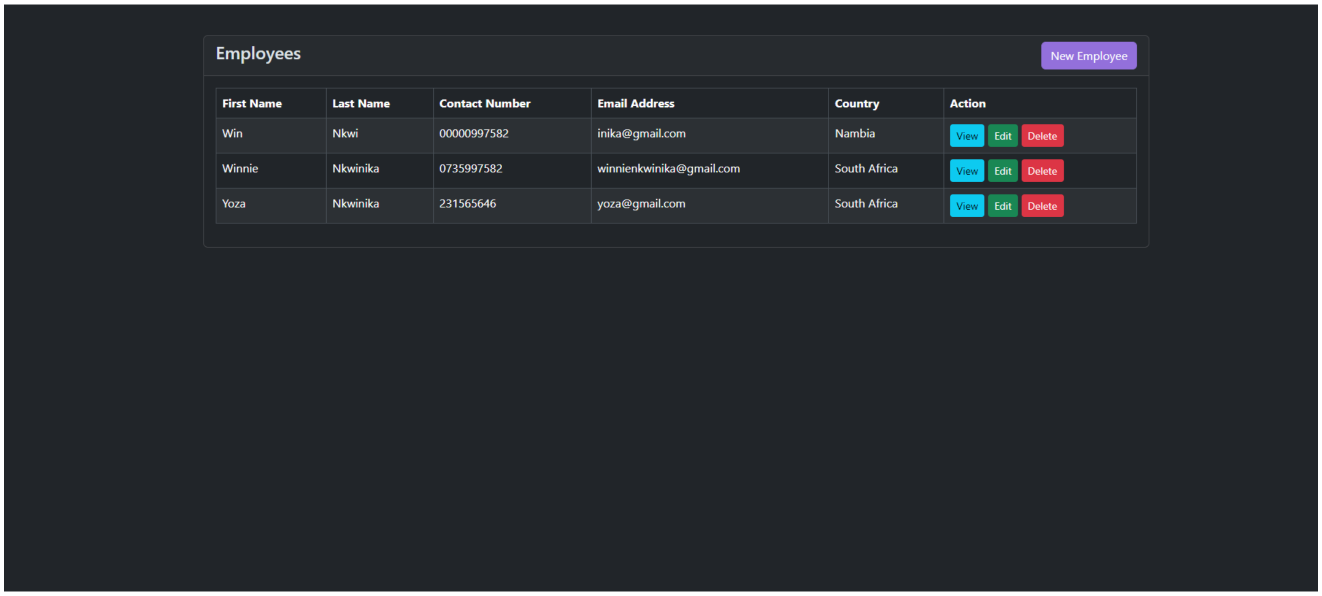
```php
<?php
include_once '../../Helpers/EmployeeManager.php';
include_once '../../Config/DBManager.php';

use TANGENT\Helpers\EmployeeManager;

$employees = EmployeeManager::getAllEmployees();

if($employees > 0)
{
    foreach($employees as $employee)
    {
        ?>
        <tr>
            <td><?= $employee['first_name']; ?></td>
            <td><?= $employee['last_name']; ?></td>
            <td><?= $employee['contact_number']; ?></td>
            <td><?= $employee['email_address']; ?></td>
            <td><?= $employee['country']; ?></td>
            <td>
                <a href="view.php?id=<?= $employee['id']; ?>" class="btn btn-
info btn-sm">View</a>
                <a href="edit.php?id=<?= $employee['id']; ?>" class="btn btn-
success btn-sm">Edit</a>
                <form action="../../App/delete.php" method="POST" class="d-
inline">
                    <button type="submit" name="delete_employee"
value="<?=$employee['id'];?>" class="btn btn-danger btn-sm">Delete</button>
                </form>
            </td>
        </tr>
        <?php
    }
}
else
{
    echo "<h5> No Record Found </h5>";
}
```



Employees

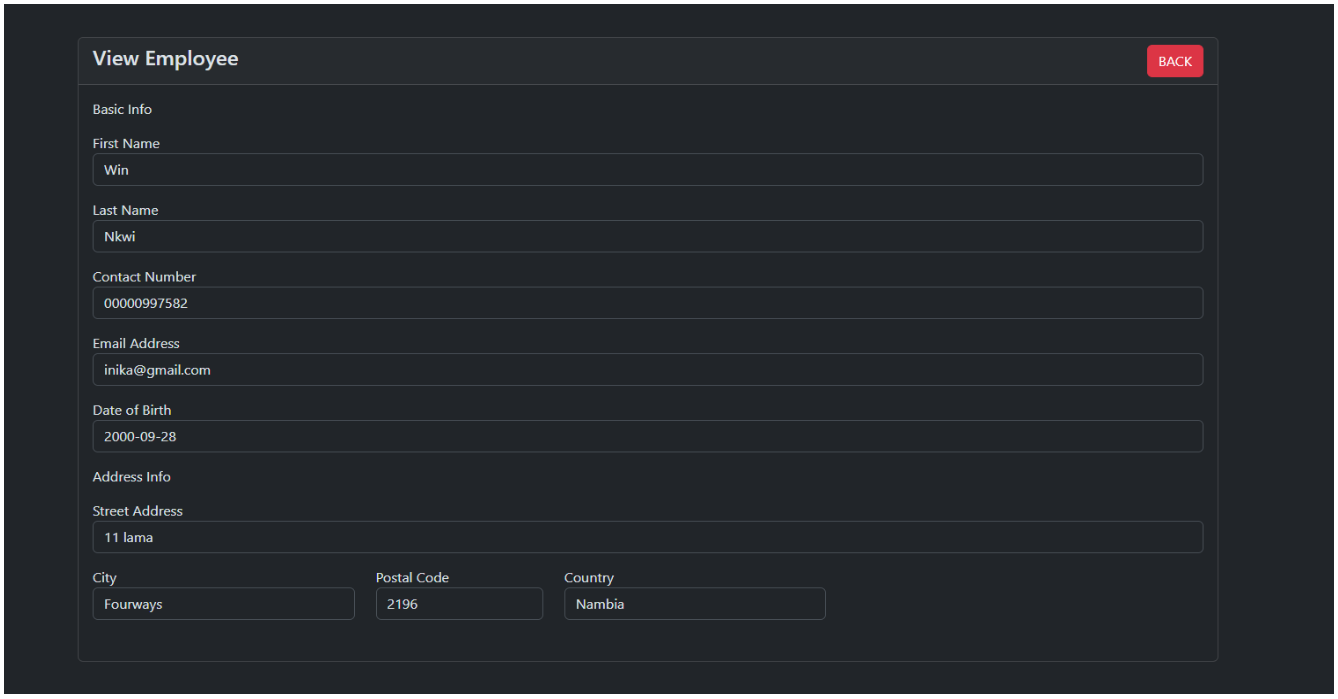
New Employee

First Name	Last Name	Contact Number	Email Address	Country	Action
Winnie	Nkwinika	0735997582	winnienkwinika@gmail.com	South Africa	View Edit Delete



Employees

First Name	Last Name	Contact Number	Email Address	Country	Action
Win	Nkwi	00000997582	inika@gmail.com	Nambia	View Edit Delete
Winnie	Nkwinika	0735997582	winnienkwinika@gmail.com	South Africa	View Edit Delete
Yoza	Nkwinika	231565646	yoza@gmail.com	South Africa	View Edit Delete



View Employee

Basic Info

First Name: Win

Last Name: Nkwi

Contact Number: 00000997582

Email Address: inika@gmail.com

Date of Birth: 2000-09-28

Address Info

Street Address: 11 lama

City: Fourways

Postal Code: 2196

Country: Nambia

EmployeeManager class is an Object-Oriented Programming (OOP) approach, utilizing namespaces and proper method structure. It also incorporates validation checks to ensure data integrity before interacting with the database.

- **addEmployee($table_name, $employee)**:

  Adds an employee to the specified table in the database.

- **addEmployeeSkills($table_name, $employeeSkills)**:

  Adds skills for an employee to the specified table in the database after validating the input.

- **editEmployee($table_name, $employee)**:

  Edits the employee details in the specified table in the database.

- **editEmployeeSkills($table_name, $employeeSkills)**:

  Edits the skills of an employee in the specified table in the database after validating the input.

- **updatedSkills($table_name, $employeeSkills, $employee)**:

  Updates the skills of an employee in the specified table in the database, adding new skills.

- **getSeniorityRatings()**:

  Retrieves seniority ratings from the database.

- **getAllEmployees()**:

  Retrieves all employees from the database.

- **getEmployeeSkills()**:

  Retrieves employee skills from the database.

- **getEmployee($employee)**:

  Retrieves a specific employee based on their ID.

- **deleteEmployee($table, $employee)**:

  Deletes an employee from the specified table in the database.

- **deleteEmployeeSkills($table, $employee)**:

  Deletes employee skills from the specified table in the database.

- **searchEmployee($searchTerm)**:

  Searches for employees based on a search term.

- **employeeFields($table_name, $employee, $bool)**:

  Validates employee details and returns an array of employee data.

DBManager class is intended to manage a MySQL database connection and perform various database operations such as querying, creating, updating, deleting, and checking for existence of records.

- It encapsulates database management functionalities.
- **Properties**:

  Private static property **$_instance** holds an instance of the class (singleton pattern).

- **Constructor and Connection**:

  The private constructor initiates a database connection by calling **createConnection()** method.

- **createConnection()**
  method connects to the MySQL database using **mysqli_connect**.

- **Singleton Pattern**:

  getInstance() method provides access to a singleton instance of the **DBManager** class.

- **Database Operations**:

  Methods like **query**, **create**, **update**, **delete**, and **exists** perform their respective database operations using MySQL queries.

- **Destructor**:

The **__destruct()** method closes the database connection when the instance is destroyed.

Composer.json

It outlines the project's metadata, dependencies, and autoload settings required to build both the front-end and backend components of an employee application.

- **Project Information:**
- **Name:** tangent/employee
- **Description:** Build out the front-end and backend components of an employee application.
- **Type:** Project
- **Authors:**
- Name: Winnie Nkwinika
- Email: winnienkwinika@gmail.com
- **Requirements (Dependencies):**
- **ext-mysqli:** Version ^8.0
- **ext-ctype:** Any version
- **ext-json:** Any version
- **Development Requirements:**
- **phpunit/phpunit:** Version ^9.6
- **mockery/mockery:** Version ^1.6
- **Minimum Stability:** Stable
- **Autoload Settings:**
- **Psr-4 (autoload):** TANGENT namespace is mapped to the "src/" directory.
- **Autoload Development Settings:**
- **Psr-4 (autoload-dev):** TANGENT\Tests namespace is mapped to the "tests/" directory.

AddEmployeeTest code defines a PHPUnit test case to ensure that creating an **Employee** object functions correctly and handles empty fields appropriately. The **testEmployeeCreation** method checks if the employee object is created correctly with the given data, while the **testEmptyFields** method ensures that the object handles empty fields appropriately.

```
<?php
```

```php
namespace TANGENT\Tests\Unit;


use PHPUnit\Framework\TestCase;
use TANGENT\App\Employee;

class AddEmployeeTest extends TestCase
{    public function testEmptyFields()
    {
        $post = [
            'id' => '',
            "seniority_rating_id" => "",
            "year_exp" =>"",
            "first_name" => "",
            "last_name" => "",
            "contact_number" => "",
            "email_address" => "",
            "date_of_birth" => "",
            "street_address" => "",
            "city" => "",
            "postal_code" => "",
            "country" => "",
            "created_by" => "",
            "created_at" => ""
        ];

        $employee = $this->createObject($post);

        $this->assertEmpty($employee->getFirstName());
        $this->assertEmpty($employee->getCity());
        $this->assertEmpty($employee->getContactNumber());
        $this->assertEmpty($employee->getCountry());
        $this->assertEmpty($employee->getCreatedAt());
        $this->assertEmpty($employee->getCreatedBy());
        $this->assertEmpty($employee->getDateOfBirth());
        $this->assertEmpty($employee->getEmail());
        $this->assertEmpty($employee->getPostalCode());
        $this->assertEmpty($employee->getYearExp());
    }

    public function testEmployeeCreation()
    {
        $post = [
            "id" => "NW1236",
            "seniority_rating_id" => ['2'],
            "year_exp" => ['2'],
            "first_name" => "nana",
            "last_name" => "nkwi",
            "contact_number" => "0121111111",
            "email_address" => "win@gmail.com",
            "date_of_birth" => "1990-02-02",
            "street_address" => "154 llama",
            "city" => "JHB",
            "postal_code" => "0000",
            "country" => "South Africa"
        ];

        $employee = $this->createObject($post);

        $this->assertInstanceOf(Employee::class, $employee);

        $this->assertEquals($post["id"], $employee->getId());
        $this->assertEquals($post["seniority_rating_id"], $employee-
>getSeniorityRating());
        $this->assertEquals(ucwords($post["first_name"]), $employee-
>getFirstName());
        $this->assertEquals(ucwords($post["last_name"]), $employee-
>getLastName());
        $this->assertEquals(($post["postal_code"]), $employee-
>getPostalCode());
        $this->assertEquals(($post["email_address"]), $employee->getEmail());
        $this->assertEquals(ucwords($post["city"]), $employee->getCity());
        $this->assertEquals(ucwords($post["country"]), $employee-
>getCountry());
    }

    private function createObject($post)
    {
        return new Employee(
            $post["id"],
            $post["seniority_rating_id"],
            $post["year_exp"],
            $post["first_name"],
            $post["last_name"],
            $post["contact_number"],
            $post["email_address"],
            $post["date_of_birth"],
            $post["street_address"],
            $post["city"],
            $post["postal_code"],
            $post["country"],
                '',
                ''
            );
        }
    }
```