PHP – Hausübung 5

inc_db_config.php

```php
<?php

define('DB_SERVER', 'localhost');
define('DB_USERNAME', 'kwm226hue05');
define('DB_PASSWORD', '.kwm.');
define('DB_DATABASE', 'kwm226_ue05_schickmair');

?>
```

index.php

```php
<?php
session_name("sess_kwm226_hue05");
session_start();

spl_autoload_register(function ($sClassname) {
    require_once($sClassname.".php");
});

Database::loadConfig("inc_db_config.php");
?>

<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>ToDoList</title>
    <link rel="stylesheet" type="text/css" href="css/style.css"/>
    <!-- CSS only -->
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
+0n0xVW2eSR5OomGNYDnhzAbDsOXxcvSN1TPprVMTNDbiYZCxYbOOl7+AMvyTG2x"
crossorigin="anonymous">
</head>
<body>
<?php
```

```
//Login and Registration Process incl. Error-Handling
$errors = [];
if(!isset($_SESSION["user"])){
    //case user is not logged in
    if(isset($_REQUEST["action"]) && $_REQUEST["action"] == "login"){
        //case user wants to log in
        if(isset($_REQUEST["username"])
            && isset($_REQUEST["password"])
            && $_REQUEST["username"] != ""
            && $_REQUEST["password"] != ""
            && checkLogin($_REQUEST["username"], $_REQUEST["password"])){

            //case user data is correct
            $_SESSION["user"] = array("username" => $_REQUEST["username"],
                "userid" => getUserId($_REQUEST["username"]));
            showData($_SESSION["user"]["userid"]);
            showTaskForm();
            showLogoutForm();
        }
        else{
            //case user data is incorrect
            echo("<div class='notify red'>Login data incorrect!</div>");
            showLoginForm();
            showRegistrationForm();
        }
    }
    else if(isset($_REQUEST["action"]) && $_REQUEST["action"] == "register"){
        //case user wants to register
        if(checkRegistration() && usernameAvailable($_REQUEST["usernamereg"])){
            //data correct
            register($_REQUEST["usernamereg"], $_REQUEST["password1"],
$_REQUEST["email"]);
            echo "<div class='notify green'>Successfully registered! You can now login
with the username ".$_REQUEST["usernamereg"]."</div><br>";
        }
        else{
            //data incorrect
            echo("<div class='notify red'>Registration data not correct!</div>");
        }
        showLoginForm();
        showRegistrationForm();
    }
    else{
        //case user wants to view page
        showLoginForm();
        showRegistrationForm();
    }
}
//case user is logged in
else {
//case user creates new task
    if (isset($_REQUEST["action"]) && $_REQUEST["action"] == "addTasks") {
        if (titleAvailable($_REQUEST["addTitle"]) && checkAddEntry()) {
            createTask($_SESSION["user"]["userid"], $_REQUEST["addTitle"],
$_REQUEST["addNote"]);
            echo("<div class='notify green'>New task created.</div>");
        }
        else{
            echo("<div class='notify red'>Task could not be created -
incorrect/duplicate content.</div>");
        }
        showData($_SESSION["user"]["userid"]);
        showTaskForm();
        showLogoutForm();
    }
    //case user wants to logout
    else if (isset($_REQUEST["action"]) && $_REQUEST["action"] == "logout") {
        unset($_SESSION["user"]);
        showLoginForm();
        showRegistrationForm();
    }
    //case user wants to edit a task
    else if(isset($_REQUEST["action"]) && $_REQUEST["action"] == "edittask"){
        if(isset($_REQUEST["entryid"])) {
            $_SESSION["task"] = $_REQUEST["entryid"];
        }
        else{
            echo("<div class='notify red'>Task could not be edited.</div>");
        }
        showData($_SESSION["user"]["userid"]);
        showEditForm(getEditData($_REQUEST["entryid"], $_SESSION["user"]));
        showLogoutForm();
    }
    //case user wants to submit Edit-Form
    else if(isset($_REQUEST["action"]) && $_REQUEST["action"] == "editTaskSubmit") {
        if (checkEditEntry()) {
            if (editTask($_SESSION["task"], $_SESSION["user"])) {
                echo("<div class='notify green'>Task successfully edited.</div>");
            } else {
                echo("<div class='notify red'>Task could not be edited.</div>");
            }
            showData($_SESSION["user"]["userid"]);
            showTaskForm();
            showLogoutForm();
        }
        else {
            showData($_SESSION["user"]["userid"]);
            showEditForm(getEditData($_SESSION["task"], $_SESSION["user"]));
            showLogoutForm();
        }
    }
    //case user wants to delete task
    else if(isset($_REQUEST["action"]) && $_REQUEST["action"] == "deletetask"){
        if(isset($_REQUEST["entryid"])){
            if(deleteTask($_REQUEST["entryid"], $_SESSION["user"])){
                echo("<div class='notify green'>Task was successfully deleted.</div>");
            }
            else{
                echo("<div class='notify red'>Task could not be deleted.</div>");
            }
        }
        showData($_SESSION["user"]["userid"]);
        showTaskForm();
        showLogoutForm();
    }
    //case user wants to finish task
    else if(isset($_REQUEST["action"]) && $_REQUEST["action"] == "finishtask"){
        if(isset($_REQUEST["entryid"])){
            if(finishTask($_REQUEST["entryid"], $_SESSION["user"])){
                echo("<div class='notify green'>Task was successfully
finished.</div>");
            }
            else{
                echo("<div class='notify red'>Task could not be finished.</div>");
            }
        }
        showData($_SESSION["user"]["userid"]);
        showTaskForm();
        showLogoutForm();
    }
    //case user wants to open task
    else if(isset($_REQUEST["action"]) && $_REQUEST["action"] == "opentask"){
        if(isset($_REQUEST["entryid"])){
            if(openTask($_REQUEST["entryid"], $_SESSION["user"])){
                echo("<div class='notify green'>Task was successfully opened.</div>");
            }
            else{
                echo("<div class='notify red'>Task could not be opened.</div>");
            }
        }
        showData($_SESSION["user"]["userid"]);
        showTaskForm();
        showLogoutForm();
    }
    //case user wants to view page
    else {
        showData($_SESSION["user"]["userid"]);
        showTaskForm();
        showLogoutForm();
    }
}

//getEditData()
//Fetches the entry-data of the old entry to show it in the edit-form.
function getEditData(int $taskID, array $user){
    $userQuery = "SELECT creatorid FROM todolist_item WHERE entryid=".$taskID.";";
    $taskOwner = Database::selectQuery($userQuery)->fetch_assoc()['creatorid'];

    if($taskOwner == $user['userid'] || isAdmin($user['userid'])){
        $deleteQuery = "SELECT title, text, entryid FROM todolist_item WHERE entryid =
".$taskID.";";
        $row = Database::selectQuery($deleteQuery)->fetch_assoc();
        if($row){
            return $row;
        }
        return null;
    }
    else return null;
}

//editTask()
//Checks if there are only allowed chars (escaped) to avoid SQL injection.
//Updates the DB row with new values.
function editTask(int $taskID, array $user){
    $escaped = Database::realEscape($_REQUEST["editTitle"], $_REQUEST["editNote"]);
    $userQuery = "SELECT creatorid FROM todolist_item WHERE entryid=".$taskID.";";
    $taskOwner = Database::selectQuery($userQuery)->fetch_assoc()['creatorid'];

    if($taskOwner == $user['userid'] || isAdmin($user['userid'])){
        $editQuery1 = "UPDATE todolist_item
                       SET title = '".$escaped["val1"]."',
                       text='".$escaped["val2"]."'  ,
                       editorid=".$_SESSION["user"]["userid"].",
                       editdate='".date("Y-m-d")."'
                       WHERE entryid=".$taskID.";";
        if(Database::updateQuery($editQuery1)){
            return true;
        }
        return false;
    }
    else return false;
}
```

```
//deleteTask()
//Deletes a task if the user is allowed to (creator or admin).
//Deletes the row in the todolist-item table.
function deleteTask(int $taskID, array $user){
    $userQuery = "SELECT creatorid FROM todolist_item WHERE entryid=".$taskID.";";
    $taskOwner = Database::selectQuery($userQuery)->fetch_assoc()['creatorid'];

    if($taskOwner == $user['userid'] || isAdmin($user['userid'])){
        $deleteQuery = "DELETE FROM todolist_item WHERE entryid=".$taskID.";";
        if(Database::deleteQuery($deleteQuery)){
            return true;
        }
        return false;
    }
    else return false;
}

//openTask()
//Opens a task if the user is allowed to (creator or admin).
//Updates the row in the todolist-item table -> status "aktiv".
function openTask(int $taskID, array $user){
    $userQuery = "SELECT creatorid FROM todolist_item WHERE entryid=".$taskID.";";
    $taskOwner = Database::selectQuery($userQuery)->fetch_assoc()['creatorid'];

    if($taskOwner == $user['userid'] || isAdmin($user['userid'])){
        $finishQuery = "UPDATE todolist_item SET status = 'aktiv' WHERE
entryid=".$taskID.";";
        if(Database::updateQuery($finishQuery)){
            return true;
        }
        return false;
    }
    else return false;
}

//finishTask()
//Opens a task if the user is allowed to (creator or admin).
//Updates the row in the todolist-item table -> status "abgeschlossen".
function finishTask(int $taskID, array $user){
    $userQuery = "SELECT creatorid FROM todolist_item WHERE entryid=".$taskID.";";
    $taskOwner = Database::selectQuery($userQuery)->fetch_assoc()['creatorid'];

    if($taskOwner == $user['userid'] || isAdmin($user['userid'])){
        $finishQuery = "UPDATE todolist_item SET status = 'abgeschlossen' WHERE
entryid=".$taskID.";";
        if(Database::updateQuery($finishQuery)){
            return true;
        }
        return false;
    }
    else return false;
}
```

```php
//createTask()
//Creates a new task in the database
function createTask(int $creatorid, string $title, string $text){
    $escaped = Database::realEscape($title, $text);
    $insertQuery = "INSERT INTO todolist_item (creationdate, creatorid, title, text)
                    VALUES ('".date("Y-m-d")."', '".$creatorid."',
'".$escaped["val1"]."', '".$escaped["val2"]."');";
    $taskID = Database::insertQuery($insertQuery);

    if($taskID != 0){
        $insertQuery2 = "INSERT INTO user_item (entryid, userid) VALUES (".$taskID.",
".$creatorid.");";
        Database::insertQuery($insertQuery2);
        return true;
    }
    return false;
}


//getUserId()
//Gets the UserId of a user with a certain, unique username.
function getUserId(string $username){
    $escaped = Database::realEscape($username);
    $selectQuery = "SELECT userid FROM user WHERE username='".$escaped["val1"]."';";
    $result = Database::selectQuery($selectQuery);
    if($row = $result->fetch_assoc()){
        return $row["userid"];
    }
    else return null;
}

//usernameAvailable()
//Checks if the entered username in the registration form is already taken or not.
function usernameAvailable(string $username){
    $escaped = Database::realEscape($username);
    $selectQuery = "SELECT username FROM user WHERE username='".$escaped["val1"]."';";
    $result = Database::selectQuery($selectQuery);

    if($result->num_rows < 0){
        return false;
    }
    return true;
}

//isAdmin()
//Checks if recent user has the role admin in the database
function isAdmin(int $userID):bool{
    $selectQuery = "SELECT role FROM user
                    WHERE userid = " . $userID . ";";

    $data = Database::selectQuery($selectQuery);
    $row = mysqli_fetch_assoc($data);
    if($row["role"] != 1)
        return false;
    else
        return true;
}
```

```
//register()
//This function allows a new user to register by entering all
//data needed. If the user wants to register with a already
//existing username, a error message shows up.
function register(string $username, string $password, string $email):bool{
    $escaped = Database::realEscape($username, $email);
    $insertQuery = "INSERT into user (username, password, email)
    VALUES ('".$escaped["val1"]."',
    '".md5($password)."',
    '".$escaped["val2"]."');";
    $id = Database::insertQuery($insertQuery);

    if($id != 0)
        return true;
    return false;
}

//titleAvailable()
//Checks if the title of the task is already used.
function titleAvailable(string $title){
    $selectQuery = "SELECT title FROM todolist_item
                    WHERE creatorid='" .$_SESSION["user"]["userid"]."'
                    AND title='".$title."';";

    $result = Database::selectQuery($selectQuery);

    if($result->num_rows > 0){
        return false;
    }
    return true;
}

//checkAddEntry()
//Checks the input fields of the form to add new tasks
function checkAddEntry():bool{
    global $errors;
    if(!isset($_REQUEST["addTitle"]) || strlen($_REQUEST["addTitle"]) < 6)
        $errors["addTitle"] = "<i>Title must be at least 6 characters long!</i>";

    if(count($errors) > 0)
        return false;
    return true;
}

//checkEditEntry()
//Checks the input fields of the form to edit tasks
function checkEditEntry():bool{
    global $errors;
    if(!isset($_REQUEST["editTitle"]) || strlen($_REQUEST["editTitle"]) < 6)
        $errors["editTitle"] = "<i>Title must be at least 6 characters long!</i>";

    if(count($errors) > 0)
        return false;
    return true;
}
```

```php
//checkLogin()
//Checks the input fields of the login form by comparing it with the saved data of the
userList.
//Only if username and password are correct, the user can login.
function checkLogin(string $username, string $password):bool{
    $escaped = Database::realEscape($username, $password);
    $selectQuery = "SELECT username FROM user WHERE username='".$escaped["val1"]."' AND
password='".md5($password)."';";
    $result = Database::selectQuery($selectQuery);
    if($result->num_rows > 0){
        return true;
    }
    else {
        global $errors;
        //username
        if (!isset($_REQUEST["username"]))
            $errors["username"] = "<br><i>Please enter valid username.</i>";
        //password
        if (!isset($_REQUEST["password"]))
            $errors["password"] = "<br><i>Please enter valid password.</i>";
        return false;
    }
}

//checkRegistration()
//Checks if various rules of the input fields of the registration form
//are followed. Otherwise the errors will be saved.
function checkRegistration():bool{
    global $errors;
    //username
    if(!isset($_REQUEST["usernamereg"]) || strlen($_REQUEST["usernamereg"]) < 6)
        $errors["usernamereg"] = "<br><i>Username must be at least 6 characters
long</i>";

    //password
    if(!isset($_REQUEST["password1"]) ||
        !isset($_REQUEST["password2"]) ||
        $_REQUEST["password1"] != $_REQUEST["password2"] ||
        strlen($_REQUEST["password1"]) < 8)
        $errors["password1"] = "<br><i>Password must be at least 8 characters long and
must match</i>";

    //email
    if(!isset($_REQUEST["email"]) || filter_var($_REQUEST["email"],
FILTER_VALIDATE_EMAIL) === false)
        $errors["email"] = "<br><i>Must be a valid email address</i>";

    if(count($errors) > 0)
        return false;
    return true;
}
```

```
//showData()
//This function prints the TodoList. Only tasks of the
//user who is logged in will be shown, all other tasks wont be
//visible for this user. (admin sees all tasks)
function showData($userID)
{
    if (!isAdmin($userID))
        $selectQuery = "SELECT * FROM todolist_item WHERE creatorid = " . $userID .
";";
    else
        $selectQuery = "SELECT * FROM todolist_item;";

    $data = Database::selectQuery($selectQuery);
    $todolist = new TodoList($_SESSION["user"]["userid"],
$_SESSION["user"]["username"]);

    while ($row = $data->fetch_assoc()) {
        $todolistitem = new TodoListItem($row["entryid"], $row["creationdate"],
$row["creatorid"], $row["title"], $row["editdate"], $row["text"], $row["status"],
$row["editorid"]);
        array_push($todolist->entries, $todolistitem);
    }
    echo $todolist;
}

//showTaskForm()
//Output of the TaskForm - if errors occur, the incorrect fields
//will be highlighted and the error will be described
function showTaskForm(){
    global $errors;
    $title = "";

    if(isset($_REQUEST["addTitle"]) && count($errors) > 0){
        $title = $_REQUEST["addTitle"];
    }
    ?>

    <div class="form-style-5">
        <form action="<?php echo($_SERVER['PHP_SELF']);?>" method="post">
            <fieldset>
                <legend><span class="number">+</span>Eintrag hinzufügen</legend>
                <?php
                if(isset($errors["addTitle"])){
                    echo('<input type="text" name="addTitle" id="addTitle"
placeholder="Titel*" class="error" value="'.$title.'">');
                    echo($errors["addTitle"]);
                }
                else echo('<input type="text" name="addTitle" id="addTitle"
placeholder="Titel*" value="'.$title.'">');

                if(isset($_REQUEST["addNote"]) && count($errors) > 0){
                    $desc = $_REQUEST["addNote"];
                    echo('<textarea name="addNote" id="addNote">'.$desc.'</textarea>');
                }
                else echo('<textarea name="addNote" id="addNote"
placeholder="Beschreibung"></textarea>');
                ?>
            </fieldset>
            <input type="hidden" name="action" value="addTasks">
            <input type="submit" id="addTasks" value="Eintrag hinzufügen">
        </form>
    </div>
    <?php
}
```

```
//showEditForm()
//Output of the EditForm - if errors occur, the incorrect fields
//will be highlighted and the error will be described
function showEditForm(array $row){
    global $errors;
    $title = "";

    if(isset($_REQUEST["editTitle"]) && count($errors) > 0){
        $title = $_REQUEST["editTitle"];
    }
    ?>

    <div class="form-style-5">
        <form action="<?php echo($_SERVER['PHP_SELF']);?>" method="post">
            <fieldset>
                <legend><span class="number">!</span>Eintrag bearbeiten</legend>
                <?php
                if(isset($errors["editTitle"])){
                    echo('<input type="text" name="editTitle" id="editTitle"
placeholder="Titel*" class="error" value="'.$title.'">');
                    echo($errors["editTitle"]);
                }
                else echo('<input type="text" name="editTitle" id="editTitle"
placeholder="Titel*" value="'.$row["title"].'">');

                if(isset($_REQUEST["editNote"]) && count($errors) > 0){
                    $desc = $_REQUEST["editNote"];
                    echo('<textarea name="editNote"
id="editNote">'.$desc.'</textarea>');
                }
                else echo('<textarea name="editNote" id="editNote"
placeholder="Beschreibung">'.$row["text"].'</textarea>');
                ?>
            </fieldset>
            <input type="hidden" name="action" value="editTaskSubmit">
            <input type="submit" id="editTasks" value="Eintrag ändern">
        </form>
    </div>
    <?php
}

//showRegistrationForm()
//Output of the registration form - if errors occur, the incorrect fields
//will be highlighted and the error will be described
function showRegistrationForm(){
    global $errors;
    $usernameVal = "";
    $emailVal = "";

    if(isset($_REQUEST["usernamereg"]) && count($errors) > 0){
        $usernameVal = $_REQUEST["usernamereg"];
    }
    if(isset($_REQUEST["email"]) && count($errors) > 0){
        $emailVal = $_REQUEST["email"];
    }
    ?>
    <div class="registration_form">
        <h3>Registrierung:</h3>
        <form action="<?php echo($_SERVER['PHP_SELF']);?>" method="post">
            <label for="usernamereg">Username:</label>
            <?php
            if(isset($errors["usernamereg"])){
                echo('<input type="text" name="usernamereg" id="usernamereg"
value="'.$usernameVal.'" class="error">');
                echo($errors["usernamereg"]);
            }
            else
                echo('<input type="text" name="usernamereg" id="usernamereg" 
value="'.$usernameVal.'">');
            ?>
            <br>
            <label for="password1">Passwort:</label>
            <?php
            if(isset($errors["password1"])){
                echo('<input type="password" name="password1" id="password1" 
class="error">');
                echo($errors["password1"]);
            }
            else
                echo('<input type="password" name="password1" id="password1">');
            ?>
            <br>
            <label for="password2">Wiederholen:</label>
            <input type="password" name="password2" id="password2"><br>
            <label for="email">Email:</label>
            <?php
            if(isset($errors["email"])){
                echo('<input type="email" name="email" id="email" class="error" 
value="'.$emailVal.'">');
                echo($errors["email"]);
            }
            else echo('<input type="email" name="email" id="email" 
value="'.$emailVal.'">');
            ?>
            <br>
            <input type="hidden" name="action" value="register">
            <input type="submit" value="Registrieren" class="reg_button">
        </form>
    </div>
    <?php
}

//showLoginForm()
//Output of the login form
function showLoginForm(){
    global $errors;
    $usernameVal = "";

    if(isset($_REQUEST["username"]) && count($errors) > 0){
        $usernameVal = $_REQUEST["username"];
    }?>

    <div class="login_form">
    <h3>Login:</h3>
    <form action="<?php echo($_SERVER['PHP_SELF']);?>" method="post">
        <label for="username">Username:</label>
        <?php
        if(isset($errors["username"])){
            echo('<input type="text" name="username" 
id="username"/>'.$usernameVal.'<br>');
            echo($errors["username"]);
        }
        else echo('<input type="text" name="username" id="username"/><br>');
        ?>
        <label for="password">Passwort:</label>
        <?php
        if(isset($errors["password"])){
            echo('<input type="password" name="password" id="password"/><br>');
            echo($errors["username"]);
        }
        else echo('<input type="password" name="password" id="password"/><br>');
        ?>
        <input type="hidden" name="action" value="login">
        <input type="submit" class="login_button" value="Login">
    </form>
    </div><?php
}

//showLogoutForm()
//Output of the logout button
function showLogoutForm(){ ?>
    <form action="<?php echo($_SERVER['PHP_SELF']);?>" method="post">
        <input type="hidden" name="action" value="logout">
        <input type="submit" value="Logout" class="logout_button">
    </form>
    <?php
}?>

</body>
</html>
```

```php
<?php

class Database
{
    public static $oMysqli;

    public static function loadConfig(string $sConfigFile)
    {
        require_once ($sConfigFile);
    }

    public static function deleteQuery(string $sQuery):bool
    {
        if (Database::connect())
        {
            $mResult = Database::$oMysqli->query($sQuery);
            Database::disconnect();
            return $mResult; // result only tells us if the SQL statement could be
processed and not if something was actually deleted
        }
        else
        {
            echo "Could not connect in deleteQuery!";
            return false;
        }
    }

    public static function insertQuery(string $sQuery):int
    {
        if (Database::connect())
        {
            $mResult = Database::$oMysqli->query($sQuery);
            $iID = Database::$oMysqli->insert_id;
            Database::disconnect();
            return $iID; // We return the id of the newly inserted row
        }
        else
        {
            echo "Could not connect in insertQuery!";
            return 0;
        }
    }
    public static function selectQuery(string $sQuery): ?mysqli_result
    {
        if (Database::connect())
        {
            $mResult = Database::$oMysqli->query($sQuery);
            Database::disconnect();
            return $mResult; // a mysqli result object -> later fetch_assoc
        }
        else
        {
            echo "Could not connect in selectQuery!";
            return null;
        }
    }

    public static function updateQuery(string $sQuery):bool
    {
        if (Database::connect())
        {
            $mResult = Database::$oMysqli->query($sQuery);
            Database::disconnect();
            return $mResult; // result only tells us if the SQL statement could be
processed and not if something was actually deleted
        }
        else
        {
            echo "Could not connect in deleteQuery!";
            return false;
        }
    }

    public static function realEscape(string $val1, string $val2 = ""):array
    {
        if (Database::connect())
        {
            $aResult = array("val1" => Database::$oMysqli->real_escape_string($val1),
                "val2" => Database::$oMysqli->real_escape_string($val2));
            Database::disconnect();
            return $aResult;
        }
        else
        {
            echo "Could not connect in realEscape!";
            return [];
        }
    }
    private static function connect():bool
    {
        Database::$oMysqli = @new mysqli(DB_SERVER, DB_USERNAME, DB_PASSWORD,
DB_DATABASE);

        if (Database::$oMysqli->connect_error)
        {
            return false;
        }
        return true;
    }

    private static function disconnect()
    {
        if (Database::$oMysqli != null)
        {
            Database::$oMysqli->close();
        }
    }
}
```

```php
<?php


class TodoList
{
    public function __construct(
        private int $userId,
        private string $username,
        public array $entries = []
    ) {
    }

    //Getter
    public function __get(string $entries):mixed
    {
        if (property_exists('TodoList', $entries)) {
            return $this->{$entries};
        } else throw new Exception("Attribute " . $entries . " does not exist in class 
TodoList!");
    }

    //Setter
    public function __set(string $entries, mixed $value):void
    {
        if (property_exists('TodoList', $entries)) {
            $this->{$entries} = $value;
        } else throw new Exception("Attribute " . $entries . " does not exist in class 
TodoList!");
    }

    //Magic Method __toString() für die Ausgabe der
    //TodoListe für den gerade angemeldeten User
    public function __toString():string
    {
        $result="<div class='welcome'>\n
        <h1>DEINE TO-DO LISTE</h1>\n
        <h2>Hallo " . $this->username . "! (ID: ".$this->userId.")</h2>\n
        </div>\n";
        $result.="<div class='toDoList'>\n";

        foreach($this->entries as $entry) {
                $result .= $entry;
        }

        $result .= "\n</div>";
        return $result;
    }
}
```

TodoListItem.php

```php
<?php


class TodoListItem
{
    public function __construct(
        private string $entryId,
        private string $creationDate,
        private string $creatorId,
        private string $title,
        private mixed $editDate,
        private string $text,
        private string $status,
        private mixed $editorId
    )
    {
    }

    //Getter
    public function __get(string $entry):mixed
    {
        if (property_exists('TodoListItem', $entry)) {
            return $this->{$entry};
        } else throw new Exception("Attribute " . $entry . " does not exist in class 
TodoListItem!");
    }

    //Setter
    public function __set(string $entry, mixed $mValue):void
    {
        if (property_exists('TodoListItem', $entry)) {
            $this->{$entry} = $mValue;
        } else throw new Exception("Attribute " . $entry . " does not exist in class 
TodoListItem!");
    }

    //Magic Method __toString() für die Ausgabe der
    //einzelnen Einträge in der TodoListe
    public function __toString():string
    {
        $result = "\n<div id='".$this->entryId."' class='entry'>
            <div class='status'>Status: ".$this->status."</div>
            <h3 class='title'>Titel: ".$this->title."</h3>";

        if($this->text != null)
            $result .= "<p class='note'>".$this->text."</p>";

        $result .= "<p>Ersteller-ID: ".$this->creatorId."</p>
            <p>Erstellungsdatum: ".$this->creationDate."</p>";

        //Erst wenn eine Bearbeitung stattfand soll auch das
        // Bearbeitungsdatum und die ID des Bearbeiters angezeigt werden
        if($this->editorId != null) {
            $result .= "<p>Letzter Bearbeiter-ID: " . $this->editorId . "</p>";
            $result .= "<p>Letztes Bearbeitungsdatum: " . $this->editDate . "</p>";
        }
        $result .= "<div class='buttons'>\n";
        if($this->status != "abgeschlossen") {
            $result .= "<input type='hidden' name='action' value='edit'>";
            $result .= "<a href='index.php?action=edittask&entryid=".$this->entryId."'
class='edit' data-toogle='modal' data-target='#editModal'>Bearbeiten</a>";
            $result .= "<input type='hidden' name='action' value='finish'>";
            $result .= "<a href='index.php?action=finishtask&entryid=" . $this->entryId
. "' class='finish'>Abschließen</a>";
        }
        else{
            $result .= "<input type='hidden' name='action' value='open'>";
            $result .= "<a href='index.php?action=opentask&entryid=" . $this->entryId .
"' class='finish'>Aktivieren</a>";
        }
        $result .= "<input type='hidden' name='action' value='delete'>";
        $result .= "<a href='index.php?action=deletetask&entryid=".$this->entryId."'
class='delete'>Löschen</a>";
        $result .= "\n</div>\n</div>";
        return $result;
    }

}
```

```css
body{
    position: relative;
    font-family: "Bahnschrift", sans-serif;
    box-sizing: border-box;
    background: url("../img/bg.jpg");
}

.welcome{
    width: 100%;
    background-color: #fff;
    text-align: center;
    color: #5e8949;
    padding: 10px;
    border-bottom: 8px solid #5e8949;
}

h1, h2{
    margin: 0;
    font-size: 28px !important;
}

.title{
    text-align: left;
    font-size: 18px !important;
}

.login_form, .registration_form{
    position: absolute;
    padding: 20px 50px 5px 50px;
    background-color: white;
    left: 35vw;
    width: 400px;
}

.login_form{
    top: 13vh;
}

.registration_form{
    top: 43vh;
    z-index: -1;
}

.registration_form h3{
    z-index: 999;
}
```

```css
i{
    color: red;
}

.notify{
    top: 40px;
    left: 20px;
    position: absolute;
    background-color: white;
    font-size: 20px;
}

.red{
    color: red;
}

.green{
    color: green;
}

form input, form a{
    box-sizing: border-box;
    border-radius: 5px;
    border: 1px solid #999;
    padding: 4px;
    outline: none;
}

.reg_button, .login_button, .logout_button{
    font-size: 1.1em;
    padding: 10px;
    font-family: Roboto, sans-serif;
    font-weight: 300;
    color: #fff;
    background-color: #5e8949;
    border-radius: 20px;
    cursor: pointer;
    margin-bottom: 2em;
    margin-top: 15px;
}

.logout_button{
    position: absolute;
    top: 10px;
    right: 30px;
}
```

```css
.toDoList{
    width: 100%;
    padding: 2%;
    display: flex;
    gap: 50px;
    flex-wrap: wrap;
    justify-content: center;
}

.entry{
    display: flex;
    justify-content: space-evenly;
    flex-direction: column;
    background-color: #fff;
    color: #5e8949;
    padding: 30px;
    border-radius: 20px;
    font-size: 13px;
    width: 400px;
}

.entry h3{
    margin-bottom: 20px;
}

.status{
    background-color: #aaa;
    color: #fff;
    padding: 20px;
    text-transform: uppercase;
    margin-bottom: 20px;
    font-size: 15px;
    text-align: center;
}

.note{
    color: #aaa;
    font-size: 14px;
    margin-bottom: 20px;
}

.buttons{
    display: flex;
    justify-content: center;
    gap: 20px;
    text-align: center;
}
```

```css
.delete, .edit, .finish{
    background-color: #aaa;
    color: #fff;
    padding: 10px;
    border-radius: 10px;
    cursor: pointer;
    font-size: 15px;
    border: none;
    width: 30%;
    text-decoration: none;
}

label{
    display: inline-block;
    width: 100px;
    margin: 2px;
}

input{
    display: inline-block;
    margin: 2px;
}

.error{
    border-color: red;
}

.form-style-5{
    max-width: 500px;
    background: #f4f7f8;
    margin: 10px auto;
    padding: 20px;
    border-radius: 8px;
}
.form-style-5 fieldset{
    border: none;
}
.form-style-5 legend {
    font-size: 1.4em;
    margin-bottom: 10px;
}


.form-style-5 label {
    display: block;
    margin-bottom: 8px;
}
```

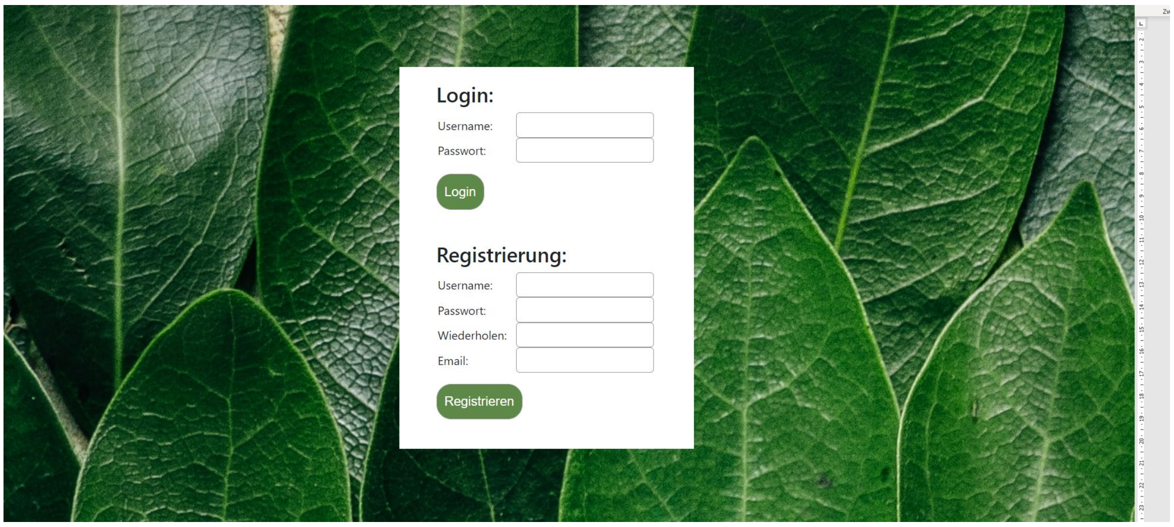
```css
.form-style-5 input[type="text"],
.form-style-5 textarea{
    background: rgba(255,255,255,.1);
    border: none;
    border-radius: 4px;
    font-size: 15px;
    outline: 0;
    padding: 10px;
    width: 100%;
    box-sizing: border-box;
    -webkit-box-sizing: border-box;
    -moz-box-sizing: border-box;
    background-color: #e8eeef;
    color:#8a97a0;
    -webkit-box-shadow: 0 1px 0 rgba(0,0,0,0.03) inset;
    box-shadow: 0 1px 0 rgba(0,0,0,0.03) inset;
    margin-top: 20px;
}
.form-style-5 input[type="text"]:focus,
.form-style-5 textarea:focus{
    background: #d2d9dd;
}
.form-style-5 select{
    height:35px;
}
.form-style-5 .number {
    background: #5e8949;
    color: #fff;
    height: 30px;
    width: 30px;
    display: inline-block;
    font-size: 0.8em;
    margin-right: 4px;
    line-height: 30px;
    text-align: center;
    text-shadow: 0 1px 0 rgba(255,255,255,0.2);
    border-radius: 15px 15px 15px 0px;
}
```

```
.form-style-5 input[type="submit"],
.form-style-5 input[type="button"]
{
    position: relative;
    display: block;
    padding: 19px 39px 18px 39px;
    color: #FFF;
    margin: 0 auto;
    background: #5e8949;
    font-size: 18px;
    text-align: center;
    font-style: normal;
    width: 100%;
    border: 1px solid #5e8949;
    border-width: 1px 1px 3px;
    margin-bottom: 10px;
}


.form-style-5 input[type="submit"]:hover,
.form-style-5 input[type="button"]:hover,
.delete:hover, .edit:hover, .finish:hover, .logout_button:hover,
.login_button:hover, .reg_button:hover{
    color: white;
    text-decoration: none;
    background: #6eaa69;
    transition: 0.5s ease;
}
```

Testfälle

Login Page



Login:

Username:

Passwort:

Login

Registrierung:

Username:

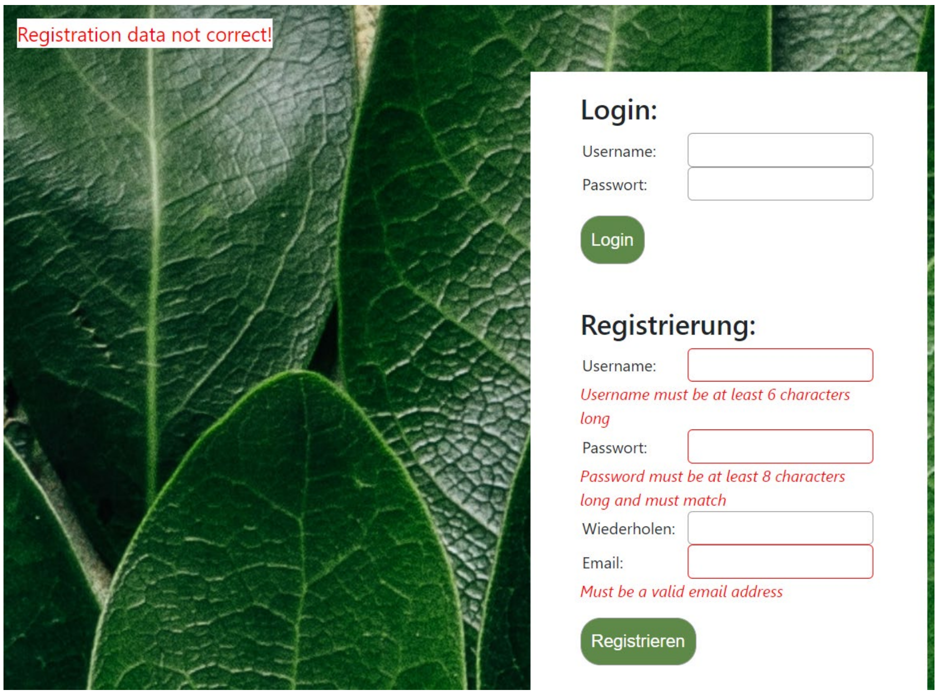
Passwort:

Wiederholen:

Email:

Registrieren

Fehlerfälle Registrierung



Registration data not correct!

Login:

Username:

Passwort:

Login

Registrierung:

Username:

Username must be at least 6 characters long

Passwort:

Password must be at least 8 characters long and must match

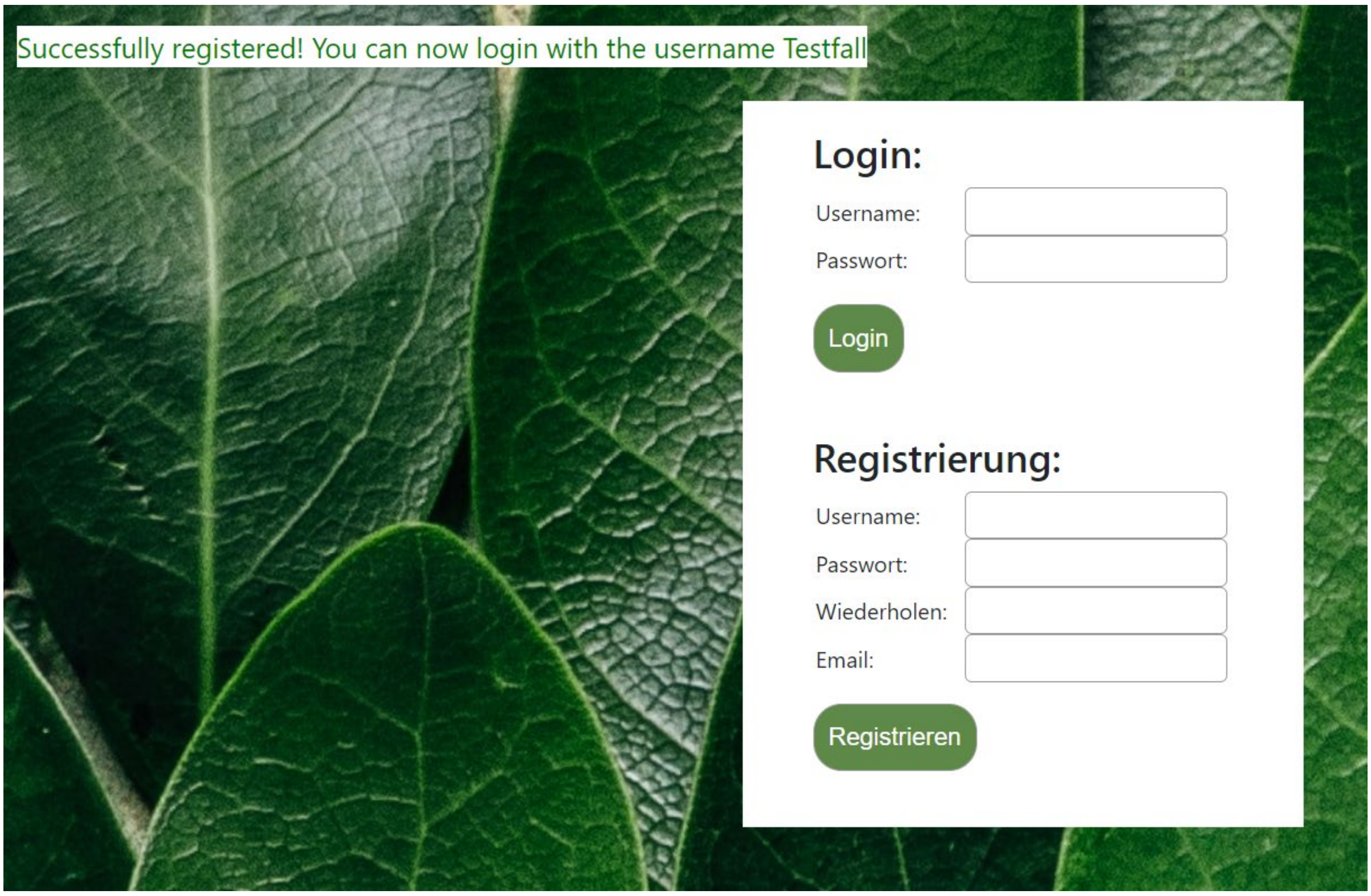
Wiederholen:

Email:

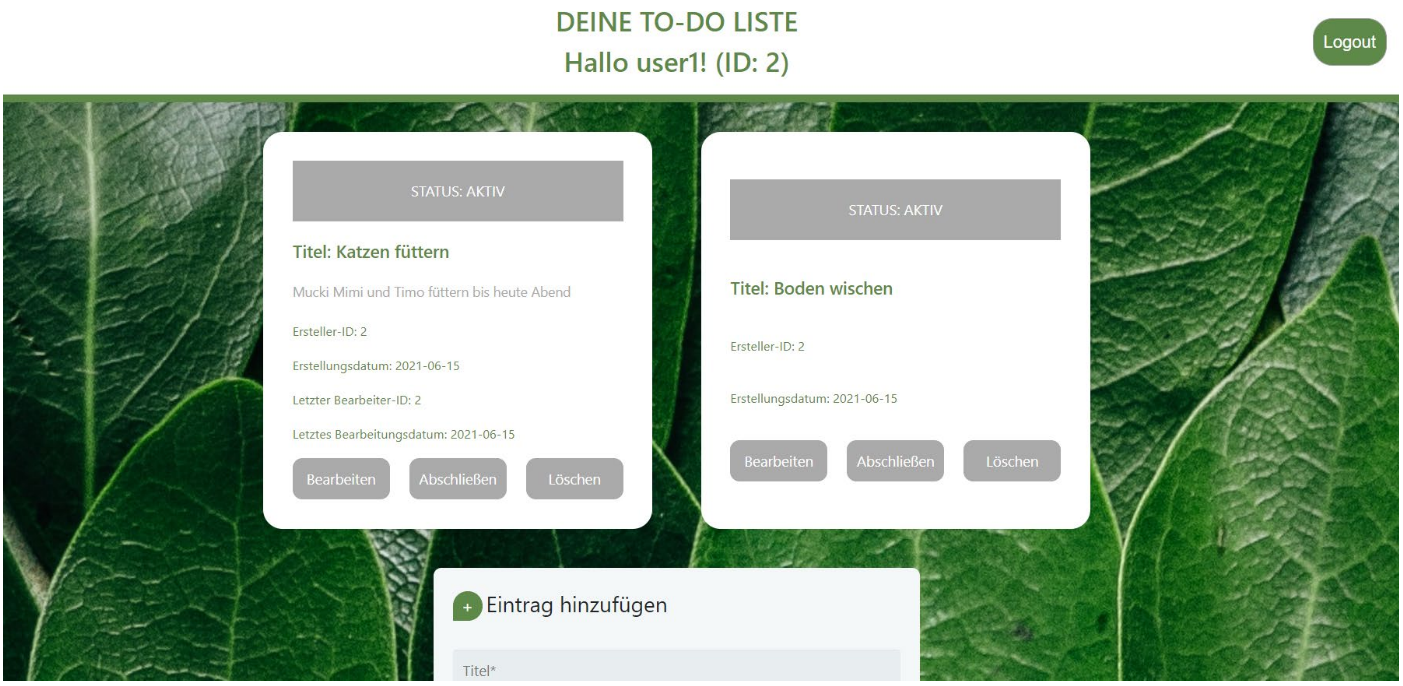
Must be a valid email address

Registrieren

Erfolgreiches Registrieren


Successfully registered! You can now login with the username Testfall

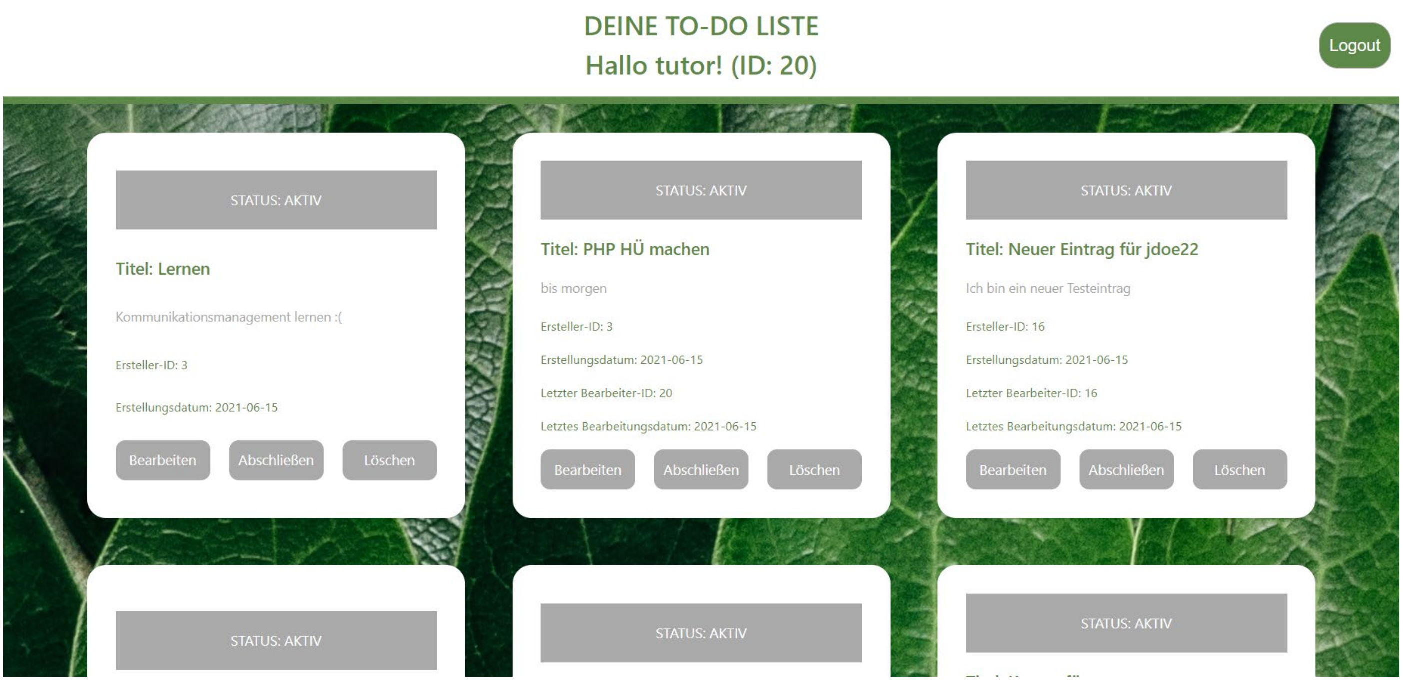
Login User: Sieht nur seine Einträge


DEINE TO-DO LISTE
Hallo user1! (ID: 2)

STATUS: AKTIV
Titel: Katzen füttern
Mucki Mimi und Timo füttern bis heute Abend
Ersteller-ID: 2
Erstellungsdatum: 2021-06-15
Letzter Bearbeiter-ID: 2
Letztes Bearbeitungsdatum: 2021-06-15

STATUS: AKTIV
Titel: Boden wischen
Ersteller-ID: 2
Erstellungsdatum: 2021-06-15

Login Admin: Sieht alle Einträge


DEINE TO-DO LISTE

Hallo tutor! (ID: 20)

Logout

STATUS: AKTIV

Titel: Lernen

Kommunikationsmanagement lernen :(

Ersteller-ID: 3

Erstellungsdatum: 2021-06-15

Bearbeiten Abschließen Löschen

STATUS: AKTIV

Titel: PHP HÜ machen

bis morgen

Ersteller-ID: 3

Erstellungsdatum: 2021-06-15

Letzter Bearbeiter-ID: 20

Letztes Bearbeitungsdatum: 2021-06-15

Bearbeiten Abschließen Löschen

STATUS: AKTIV

Titel: Neuer Eintrag für jdoe22

Ich bin ein neuer Testeintrag

Ersteller-ID: 16

Erstellungsdatum: 2021-06-15

Letzter Bearbeiter-ID: 16

Letztes Bearbeitungsdatum: 2021-06-15

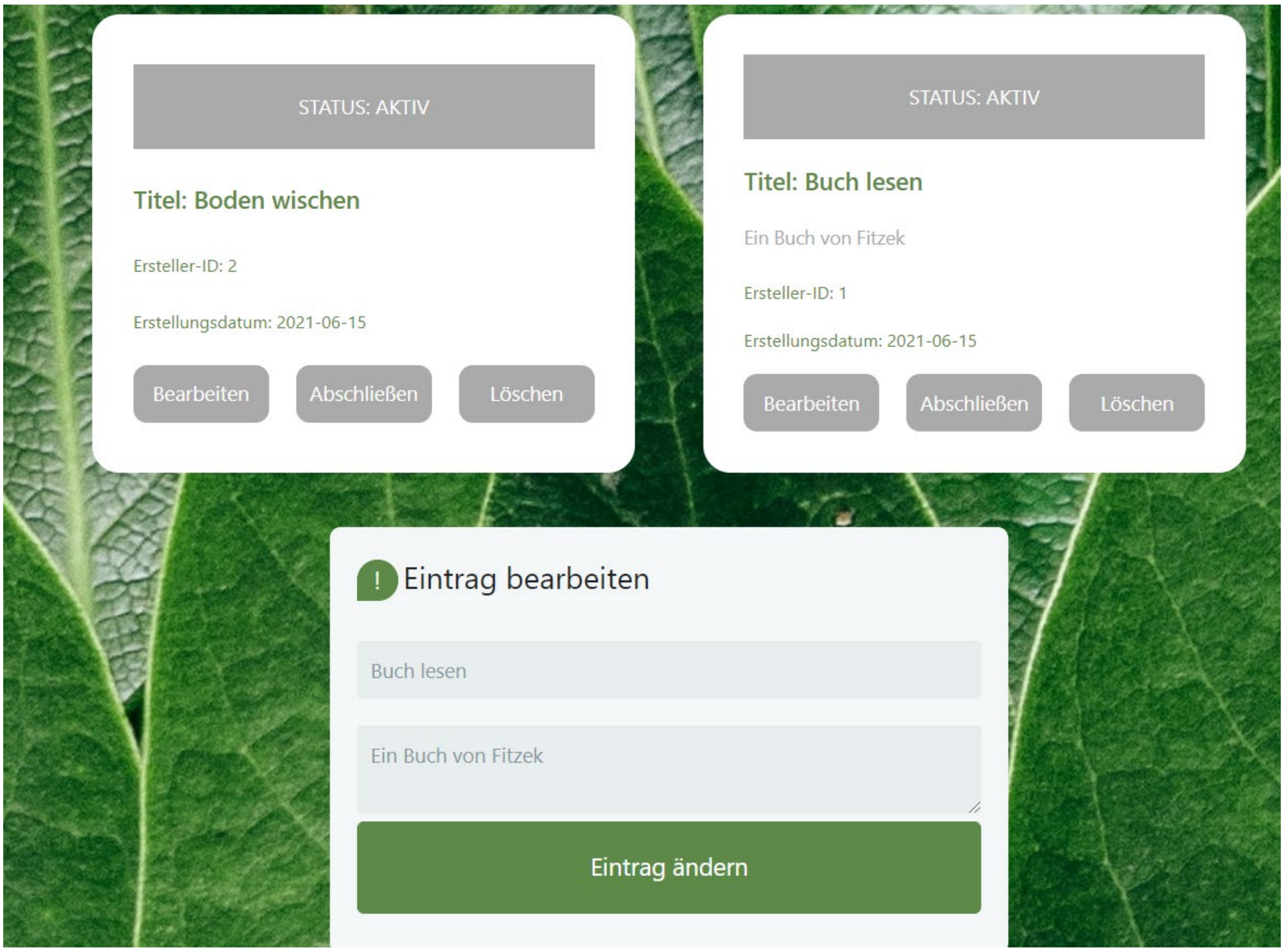
Bearbeiten Abschließen Löschen

STATUS: AKTIV

STATUS: AKTIV

STATUS: AKTIV

```
Eintrag bearbeiten: Eintrag hinzufügen ändert sich zu Eintrag bearbeiten, füllt sich
mit derzeitigen Daten
```


STATUS: AKTIV

Titel: Boden wischen

Ersteller-ID: 2

Erstellungsdatum: 2021-06-15

Bearbeiten Abschließen Löschen

STATUS: AKTIV

Titel: Buch lesen

Ein Buch von Fitzek

Ersteller-ID: 1

Erstellungsdatum: 2021-06-15

Bearbeiten Abschließen Löschen

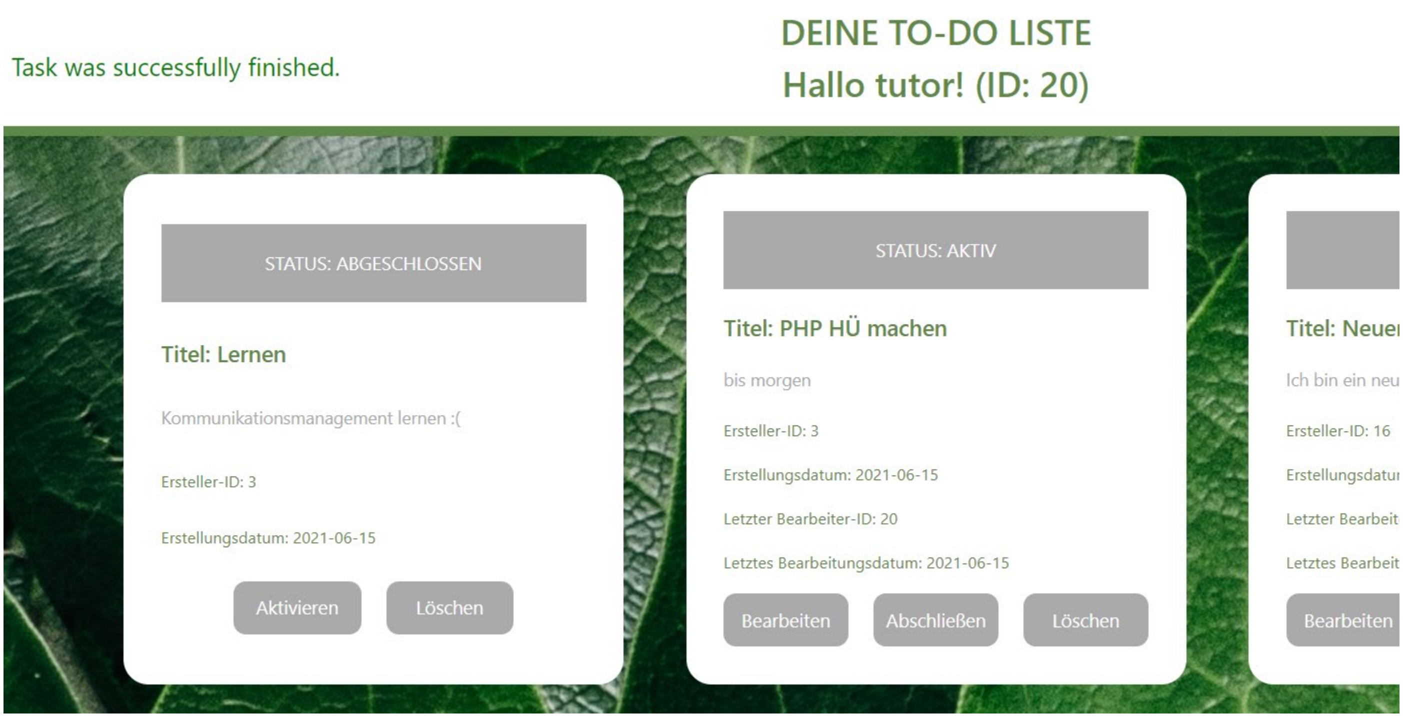
Eintrag bearbeiten

Buch lesen

Ein Buch von Fitzek

Eintrag ändern

Eintrag abschließen: Kann man aber wieder reaktivieren


Task was successfully finished.

DEINE TO-DO LISTE

Hallo tutor! (ID: 20)

STATUS: ABGESCHLOSSEN

Titel: Lernen

Kommunikationsmanagement lernen :(

Ersteller-ID: 3

Erstellungsdatum: 2021-06-15

STATUS: AKTIV

Titel: PHP HÜ machen

bis morgen

Ersteller-ID: 3

Erstellungsdatum: 2021-06-15

Letzter Bearbeiter-ID: 20

Letztes Bearbeitungsdatum: 2021-06-15

Titel: Neue

Ich bin ein neu

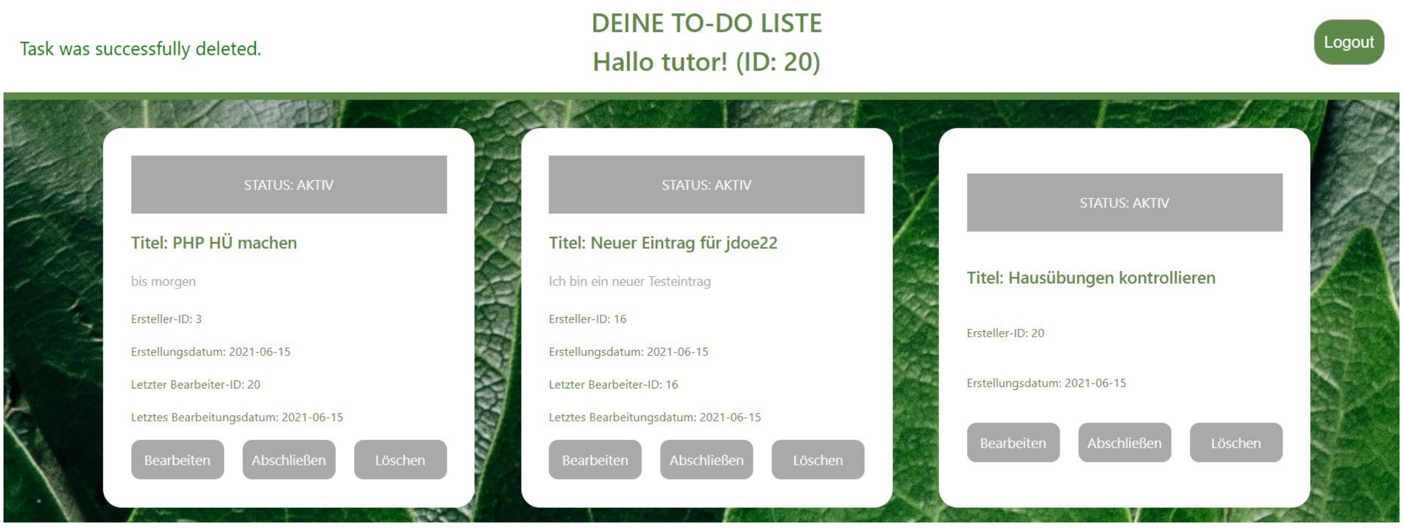
Ersteller-ID: 16

Erstellungsdatu

Letzter Bearbeit

Letztes Bearbeit

Task löschen


Task was successfully deleted.

DEINE TO-DO LISTE

Hallo tutor! (ID: 20)

STATUS: AKTIV

Titel: PHP HÜ machen

bis morgen

Ersteller-ID: 3

Erstellungsdatum: 2021-06-15

Letzter Bearbeiter-ID: 20

Letztes Bearbeitungsdatum: 2021-06-15

STATUS: AKTIV

Titel: Neuer Eintrag für jdoe22

Ich bin ein neuer Testeintrag

Ersteller-ID: 16

Erstellungsdatum: 2021-06-15

Letzter Bearbeiter-ID: 16

Letztes Bearbeitungsdatum: 2021-06-15

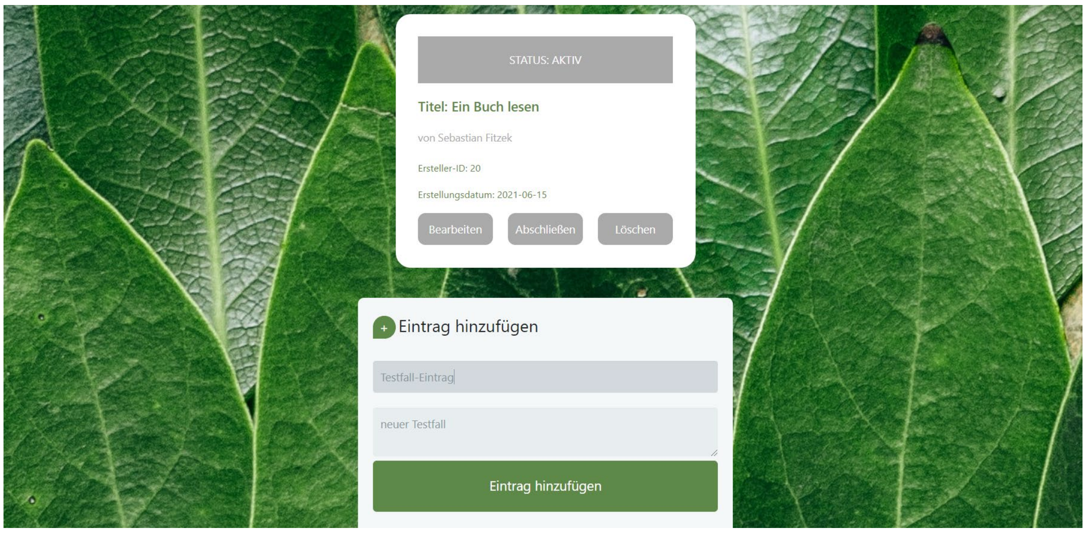
STATUS: AKTIV

Titel: Hausübungen kontrollieren

Ersteller-ID: 20

Erstellungsdatum: 2021-06-15

Task hinzufügen:


STATUS: AKTIV

Titel: Ein Buch lesen

von Sebastian Fitzek

Ersteller-ID: 20

Erstellungsdatum: 2021-06-15

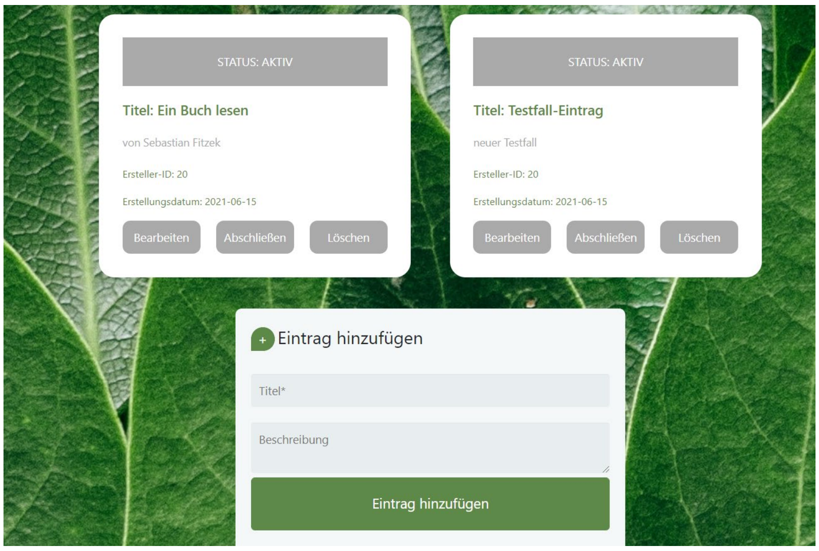
Bearbeiten Abschließen Löschen

Eintrag hinzufügen

Testfall-Eintrag

neuer Testfall

Eintrag hinzufügen


STATUS: AKTIV

Titel: Ein Buch lesen

von Sebastian Fitzek

Ersteller-ID: 20

Erstellungsdatum: 2021-06-15

Bearbeiten Abschließen Löschen

STATUS: AKTIV

Titel: Testfall-Eintrag

neuer Testfall

Ersteller-ID: 20

Erstellungsdatum: 2021-06-15

Bearbeiten Abschließen Löschen

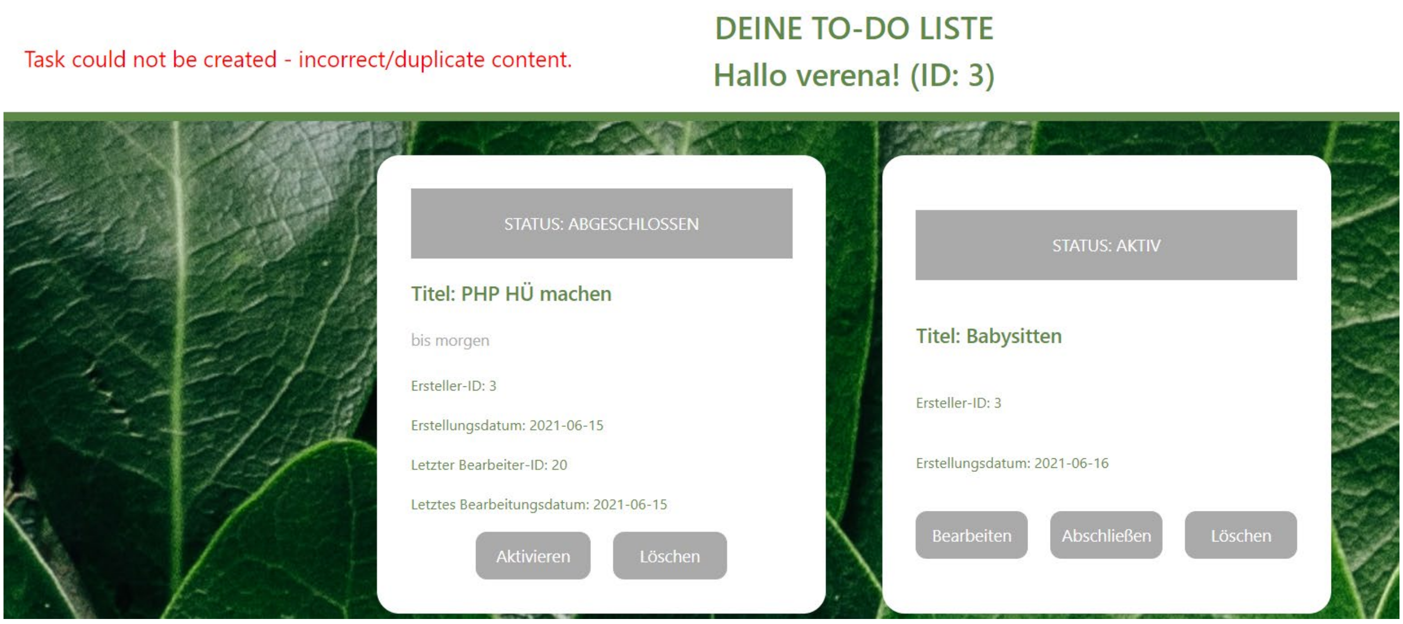
Eintrag hinzufügen

Titel*

Beschreibung

Eintrag hinzufügen

Doppelte Titel der Einträge verboten


Task could not be created - incorrect/duplicate content.

DEINE TO-DO LISTE

Hallo verena! (ID: 3)

STATUS: ABGESCHLOSSEN

Titel: PHP HÜ machen

bis morgen

Ersteller-ID: 3

Erstellungsdatum: 2021-06-15

Letzter Bearbeiter-ID: 20

Letztes Bearbeitungsdatum: 2021-06-15

Aktivieren Löschen

STATUS: AKTIV

Titel: Babysitten

Ersteller-ID: 3

Erstellungsdatum: 2021-06-16

Bearbeiten Abschließen Löschen