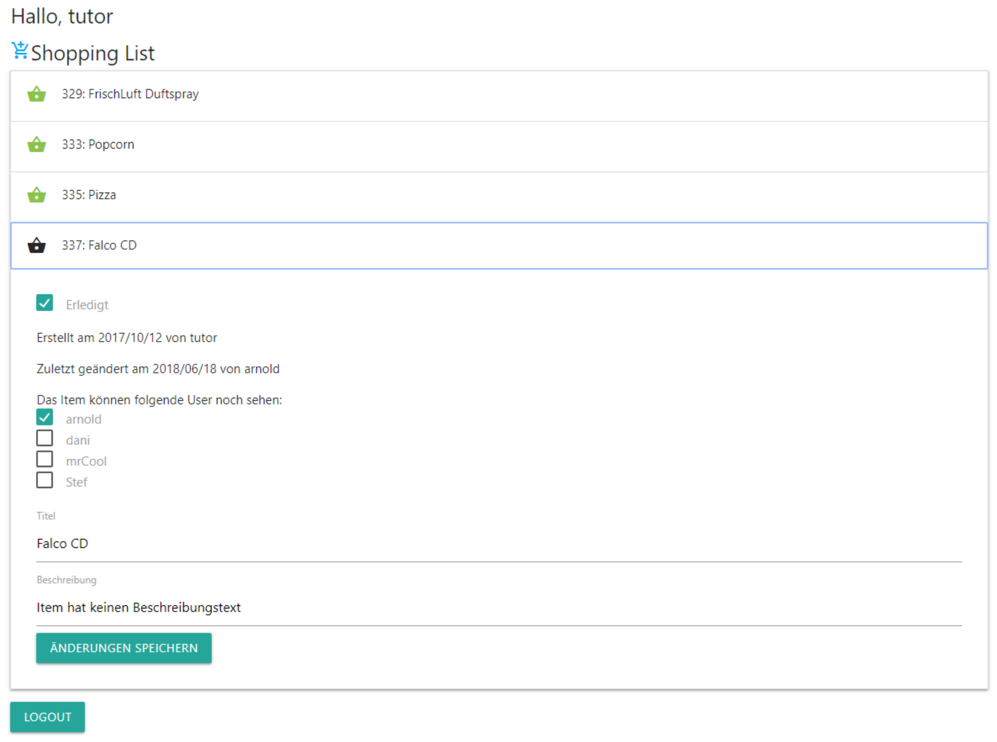
Shoppinglist Teil 3 – UE 05

Anmerkungen: Bei mir funktionierten zum Zeitpunkt der Endabgabe alle (mir bekannten) Funktionalitäten inkl. Richtigem Datenbankzugang. Sollte es dennoch Probleme mit der Datenbank geben (ich habe bereits schlechte Erfahrungen mit phpMyAdmin gemacht, obwohl ich die DB ordungsgemäß wie in der Übung besprochen exportiert habe), bitte ich um eine Verständigung per Mail.

Weitere Anmerkung: das Ausbessern der Zeilenumbrüche wurde vermieden, da der Code sonst nicht mehr gut in PHP Storm lesbar ist und Word keine gute Formatierung nicht zulässt.

Schöne Screenshots zur Darstellung der Ausgabe inkl. Datenbank


Hallo, tutor

Shopping List

329: FrischLuft Duftspray

333: Popcorn

335: Pizza

337: Falco CD

✓ Erledigt

Erstellt am 2017/10/12 von tutor

Zuletzt geändert am 2018/06/18 von arnold

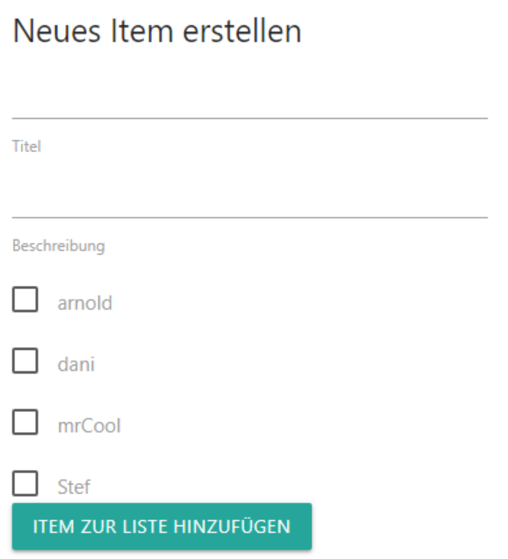
Das Item können folgende User noch sehen:
- ✓ arnold
- ☐ dani
- ☐ mrCool
- ☐ Stef

Titel

Falco CD

Beschreibung

Item hat keinen Beschreibungstext



Neues Item erstellen

Titel

Beschreibung

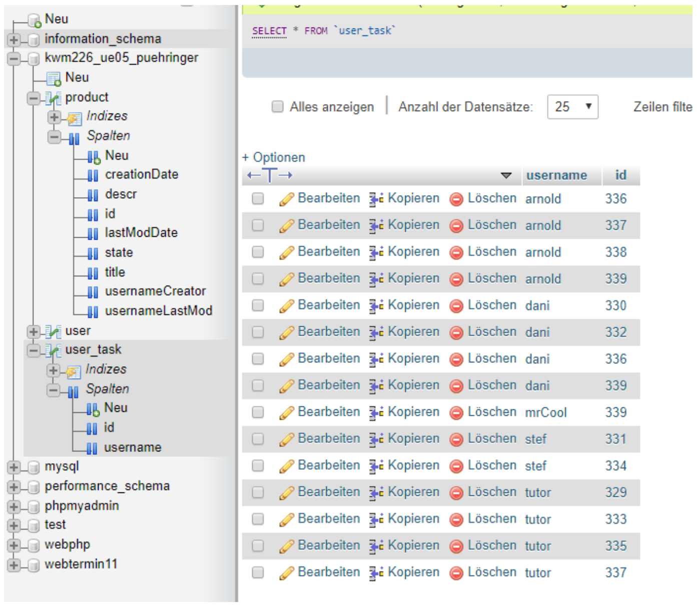
☐ arnold

☐ dani

☐ mrCool

☐ Stef

ITEM ZUR LISTE HINZUFÜGEN



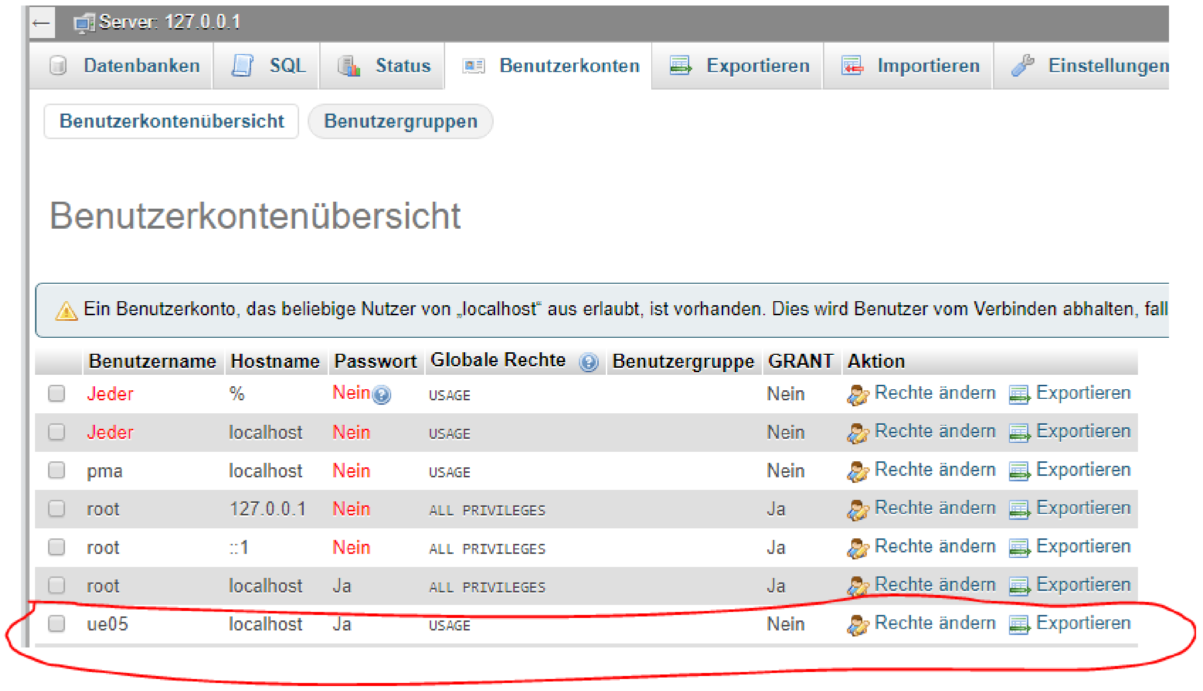
```
Neu
information_schema
kwm226_ue05_puehringer
    Neu
    product
        Indizes
        Spalten
            Neu
            creationDate
            descr
            id
            lastModDate
            state
            title
            usernameCreator
            usernameLastMod
    user
    user_task
        Indizes
        Spalten
            Neu
            id
            username
mysql
performance_schema
phpmyadmin
test
webphp
webtermin11
```

```
SELECT * FROM `user_task`
```

☐ Alles anzeigen | Anzahl der Datensätze: 25 Zeilen filte

+ Optionen

				username	id
☐	Bearbeiten	Kopieren	Löschen	arnold	336
☐	Bearbeiten	Kopieren	Löschen	arnold	337
☐	Bearbeiten	Kopieren	Löschen	arnold	338
☐	Bearbeiten	Kopieren	Löschen	arnold	339
☐	Bearbeiten	Kopieren	Löschen	dani	330
☐	Bearbeiten	Kopieren	Löschen	dani	332
☐	Bearbeiten	Kopieren	Löschen	dani	336
☐	Bearbeiten	Kopieren	Löschen	dani	339
☐	Bearbeiten	Kopieren	Löschen	mrCool	339
☐	Bearbeiten	Kopieren	Löschen	stef	331
☐	Bearbeiten	Kopieren	Löschen	stef	334
☐	Bearbeiten	Kopieren	Löschen	tutor	329
☐	Bearbeiten	Kopieren	Löschen	tutor	333
☐	Bearbeiten	Kopieren	Löschen	tutor	335
☐	Bearbeiten	Kopieren	Löschen	tutor	337



Benutzerkontenübersicht

Ein Benutzerkonto, das beliebige Nutzer von „localhost“ aus erlaubt, ist vorhanden. Dies wird Benutzer vom Verbinden abhalten, fall

	Benutzername	Hostname	Passwort	Globale Rechte	Benutzergruppe	GRANT	Aktion
	Jeder	%	Nein	USAGE		Nein	Rechte ändern Exportieren
	Jeder	localhost	Nein	USAGE		Nein	Rechte ändern Exportieren
	pma	localhost	Nein	USAGE		Nein	Rechte ändern Exportieren
	root	127.0.0.1	Nein	ALL PRIVILEGES		Ja	Rechte ändern Exportieren
	root	::1	Nein	ALL PRIVILEGES		Ja	Rechte ändern Exportieren
	root	localhost	Ja	ALL PRIVILEGES		Ja	Rechte ändern Exportieren
	ue05	localhost	Ja	USAGE		Nein	Rechte ändern Exportieren

DB_Config.php

```php
<?php

DEFINE('DB_SERVER', 'localhost');
DEFINE('DB_USER', 'ue05');
DEFINE('DB_PASSWORD', '.kwm.');
DEFINE('DB_DATABASE', 'kwm226_ue05_puehringer');

?>
```

Index.php

```
<?php
session_name("hue05");
session_start();

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

Database::loadConfig("db_config.php");
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8" />
    <title>PHP OOP - UE05</title>


    <!--This is where the magic happens!-->
    <!--Import Google Icon Font-->
    <link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

    <!--Import materialize.css-->
    <link type="text/css" rel="stylesheet" href="css/materialize.css"/>
    <script src="js/main.js"></script>
    <script src="js/materialize.js"></script>
</head>

<body>
<div class="container">
<?php

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

    //constants which can be easily changed
    define("PASSWORD_MIN_LENGTH", 3);
    define("INPUT_MIN_LENGTH", 1);


    echo("<h1 class='header center orange-text'>HUE05 - ShoppingList 3</h1>");

    /*for UE03: run these two methods to test the functionality
    testShoppingListItem();
    testShoppingList();*/

    //for UE04:
    //list of users: Daniel, H.P.Baxxter, Ben
    /*$_SESSION["userCredentials"] = array("Daniel"=>md5("test"), "H.P.Baxxter"=>md5("test"),
"Ben"=>md5("test"));
    $oShoppingList = new ShoppingList();
    addDummyDataToFillShoppingList();//filling shoppinglist with items
    //this is where the magic happens!
    manageSession();*/



    //for UE05:
    //creating one account for the tutor--> username="tutor", password: "tutor1234"

    $oShoppingList = new ShoppingList();

    //this is only do init database with users!
    if(!checkIfUserIsAlreadyRegistered("tutor")){
        registerUser("tutor", "tutor1234");
        echo("<p>Tutor wurde wie in Angabe gefordert angelegt!");
    }


    //TODO ACHTUNG ACHTUNG, diese Methode ist wichtig!!!
    //TODO ACHTUNG ACHTUNG, diese Methode ist wichtig!!!
    //TODO ACHTUNG ACHTUNG, diese Methode ist wichtig!!!
    //TODO ACHTUNG ACHTUNG, diese Methode ist wichtig!!!
    //TODO ACHTUNG ACHTUNG, diese Methode ist wichtig!!!
    //TODO ACHTUNG ACHTUNG, diese Methode ist wichtig!!!
    //this is only do init database with products!
    //TODO @Tutoren: bitte führt diese auskommentierte methode lediglich dann aus, wenn ihr mehr daten in die
datenbank laden möchtet! (wenn ihr zb den gesamten inhalt der DB löscht)
    //addDummyDataToFillShoppingList();
    manageSession();


    function testShoppingListItem(){
        echo("<br/><br/><h5 class='center'>Testing Class ShoppingListItem</h5>");

        //testing __construct
        $oListItem1 = new ShoppingListItem("Prod1", new DateTime('2018-05-05'), "Daniel", "FrischLuft
Duftspray",
            new DateTime('2018-05-05'), "Daniel", true);

        $oListItem2 = new ShoppingListItem("Prod2", new DateTime('2018-05-10'), "Daniel", "Choco Cookies",
            new DateTime('2018-05-15'), "Max Mustermann", false, "Dieser Text beschreibt den Geschmack der
super     Kekse!");

        //testing __toString
        echo("<ul class='collapsible'>".$oListItem1."</ul>");
        echo("<ul class='collapsible'>".$oListItem2."</ul>");

        //testing __get
        try{
            //valid method call
            echo("<p>Success: ".$oListItem1->sId."</p>");
        }catch(Exception $oException){
            echo("<p>Error: ".$oException->getMessage()."</p>");
        }
        echo("<p></p>");
        try{
            //invalid method call --> __get throws exception, the exception gets caught here
            echo("<p>Success: ".$oListItem1->sIDN."</p>");
        }catch(Exception $oException){
            echo("<p>Error: ".$oException->getMessage()."</p>");
        }

        //testing __set
        try{
            //valid method call
            echo("<p>Success: ".$oListItem1->sId = 'newID'."</p>");
        }catch(Exception $oException){
            echo("<p>Error: ".$oException->getMessage()."</p>");
        }
        try{
            //invalid method call --> __set throws exception, the exception gets caught here
            echo("<p>Success: ".$oListItem1->sIDN = 'newID'."</p>");
        }catch(Exception $oException){
            echo("<p>Error: ".$oException->getMessage()."</p>");
        }
    }

    function testShoppingList(){
        echo("<br/><br/><h5 class='center'>Testing Class ShoppingList</h5>");
        $oShoppingListTest = new ShoppingList();

        $oListItem1 = new ShoppingListItem("Prod1", new DateTime('2018-05-05'), "Daniel", "FrischLuft
Duftspray",
            new DateTime('2018-05-05'), "Daniel", true);

        $oListItem2 = new ShoppingListItem("Prod2", new DateTime('2018-05-10'), "Daniel", "Choco Cookies",
            new DateTime('2018-05-15'), "Max Mustermann", false, "Dieser Text beschreibt den Geschmackt der
super Kekse!");

        $oListItem3 = new ShoppingListItem("Prod3", new DateTime('2012-05-10'), "H.P.Baxxter", "Fish",
            new DateTime('2018-05-15'), "H.P.Baxxter", false);

        $oListItem4 = new ShoppingListItem("Prod4", new DateTime('2017-07-10'), "Ben", "Ben&Jerry´s",
            new DateTime('2014-12-26'), "Jerry", true, "Best Ice Cream Ever!");


        //testing addItem
        $oShoppingListTest->addItem($oListItem1);
        echo($oShoppingListTest);

        //testing editItem
        try{
            //testing editItem --> valid method call
            $oShoppingListTest->editItem($oListItem1, "sUserIdCreator", "Satoshi Nakamoto");
            echo($oShoppingListTest);
        }catch(Exception $oException){
            echo("<p>Error: ".$oException->getMessage()."</p>");
        }

        try{
            //testing editItem --> invalid method call --> editItem throws exception
            $oShoppingListTest->editItem($oListItem1, "sLeeroyJenkins", "Satoshi Nakamoto");
            echo($oShoppingListTest);
        }catch(Exception $oException){
            echo("<p>Error: ".$oException->getMessage()."</p>");
        }

        //adding more items
        $oShoppingListTest->addItem($oListItem2);
        echo($oShoppingListTest);

        $oShoppingListTest->addItem($oListItem3);
        echo($oShoppingListTest);

        $oShoppingListTest->addItem($oListItem4);
        echo($oShoppingListTest);

        //testing delete method --> entry gets done and therefore can be deleted from the shoppingList
        try{
            //valid method call
            $oShoppingListTest->deleteItem($oListItem3);
            echo($oShoppingListTest);
        }catch(Exception $oException){
            echo("<p>Error: ".$oException->getMessage()."</p>");
        }
    }

/**
 * @param string $sKey key of user
 * @return bool
 */
    function checkIfUserIsAlreadyRegistered(string $sUserNameAttempt):bool{
        $sSelectQuery = "SELECT username FROM user WHERE username='".$sUserNameAttempt."';";
        $oResult = Database::selectQuery($sSelectQuery);
        if($oResult && $oResult->num_rows > 0){
            $aRow = $oResult->fetch_assoc();
            return true;
        }
        return false;
    }

/**
 * @param string $sPasswordAttempt
 * @param string $sPasswordRepeatAttempt
 * @return bool
 */
    function checkIfPasswordIsValid(string $sPasswordAttempt, string $sPasswordRepeatAttempt):bool{
        return(($sPasswordAttempt === $sPasswordRepeatAttempt) && strlen($sPasswordAttempt) >
PASSWORD_MIN_LENGTH);
    }

/** this method uses the filter we used in the course
 * @param string $sEmail
 * @return bool
 */
    function checkIfEmailIsValid(string $sEmail):bool{
        if(filter_var($sEmail, FILTER_VALIDATE_EMAIL) !== false){
            return true;
        }
        return false;
    }

/** input_min_length is a constant!!
 * @param string $input
 * @return bool
 */
    function checkIfInputIsLongEnough(string $input):bool{
        return strlen($input) >= INPUT_MIN_LENGTH;
    }

/**
 * method for creating hard coded elements and adding them into the shoppinglist
 */
    function addDummyDataToFillShoppingList(){
        global $oShoppingList;

        $sSelectQuery = "SELECT username FROM user WHERE username='Dani';";
        $oResult = Database::selectQuery($sSelectQuery);
        if($oResult && $oResult->num_rows > 0){
            $aRow = $oResult->fetch_assoc();
        }
        $oListItem1 = new ShoppingListItem("Prod1", new DateTime('2018-05-05'), "tutor", "FrischLuft
Duftspray",
            new DateTime('2018-05-05'), "dani", true);

        $oListItem2 = new ShoppingListItem("Prod2", new DateTime('2018-05-10'), "dani", "Choco Cookies",
            new DateTime('2018-06-20'), "tutor", false, "Dieser Text beschreibt den Geschmackt der super
Kekse!", ["tutor", "tutor"]);

        $oListItem3 = new ShoppingListItem("Prod3", new DateTime('2012-05-10'), "stef", "Fish",
            new DateTime('2018-04-25'), "stef", true, "How much is the fish?", ["tutor"]);

        $oListItem4 = new ShoppingListItem("Prod4", new DateTime('2017-07-10'), "dani", "Ben&Jerry´s",
            new DateTime('2014-12-26'), "tutor", true, "Best Ice Cream Ever!", ["tutor"]);

        $oListItem5 = new ShoppingListItem("Prod5", new DateTime('2016-07-10'), "tutor", "Popcorn");
        $oListItem6 = new ShoppingListItem("Prod6", new DateTime('2018-02-10'), "stef", "Gemüse");
        $oListItem7 = new ShoppingListItem("Prod7", new DateTime('2017-04-05'), "tutor", "Pizza");
        $oListItem8 = new ShoppingListItem("Prod8", new DateTime('2017-06-15'), "arnold", "Schallplatten");
        $oListItem9 = new ShoppingListItem("Prod9", new DateTime('2017-10-12'), "tutor", "Falco CD");
        $oListItem10 = new ShoppingListItem("Prod10", new DateTime('2018-01-03'), "arnold", "Swiffers");

        $aTempArray = Array($oListItem1, $oListItem2, $oListItem3, $oListItem4, $oListItem5, $oListItem6,
$oListItem7, $oListItem8, $oListItem9, $oListItem10);
        foreach ($aTempArray as $oCurrentElem){
            registerProduct($oCurrentElem->dCreation, $oCurrentElem->sUserIdCreator, $oCurrentElem->sTitle,
$oCurrentElem->dLastMod, $oCurrentElem->sUserIdLastMod, $oCurrentElem->bState);
        }
    }


    function printLoginForm(string $sDefaultUser = ""){
        ?>
        <div class="row">
```

```
                <div class="col s4">
                    <h5 class='left-align'>LOGIN</h5>
                    <form method="post" action="<?php echo($_SERVER['PHP_SELF']); ?>">
                        <input type="text" value="<?php echo($sDefaultUser) ?>" name="username" id="username"/>
                        <label for="username">Username</label>

                        <input type="password" value="" name="password" id="password"/>
                        <label for="password">Password</label>
                        <p></p>
                        <input type="hidden" name="action" value="login">
                        <input class="btn" type="submit" value="login"/>
                    </form>
                </div>
            </div>

            <?php
    }

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

    function printRegistrationForm(string $sUsername = "", string $sEmail = ""){
            ?>
            <div class="row">
                <div class="col s4">
                    <h5 class='left-align'>REGISTRIEREN</h5>
                    <form method="post" action="<?php echo($_SERVER['PHP_SELF']); ?>">
                        <input type="text" value="<?php echo($sUsername) ?>" name="username" id="username"/>
                        <label for="username">Username</label>

                        <input type="password" value="" name="password" id="password"/>
                        <label for="password">Passwort</label>

                        <input type="password" value="" name="passwordRepeat" id="passwordRepeat"/>
                    <label for="passwordRepeat">Passwort wiederholen</label>

                    <input type="email" value="<?php echo($sEmail) ?>" name="email" id="email"/>
                    <label for="email">E-Mail Adresse</label>
                    <p></p>

                    <input class="btn" type="submit" value="Registrieren"/>
                    <input type="hidden" name="action" value="register"/>
                </form>
            </div>
        </div>
        <?php
    }

    function printCreateNewItemForm(string $sUserIdCreator, string $sTitle="", string $sDescr="",
$aSharedUserIds=array()){
    ?>
        <div class="row">
            <div class="col s4">
                <h5 class='left-align'>Neues Item erstellen</h5>
                <form method="post" action="<?php echo($_SERVER['PHP_SELF']); ?>">
                    <!--<input type="text" value="*/" name="sId" id="sId"/>
                    <label for="sId">ID Nummer</label>-->

                    <input type="text" value="<?php echo($sTitle) ?>" name="sTitle" id="sTitle"/>
                    <label for="sTitle">Titel</label>

                    <input type="text" value="<?php echo($sDescr) ?>" name="sDescr" id="sDescr"/>
                    <label for="sDescr">Beschreibung</label>

                    <div>
                    <?php
                    $aAllUsers = ShoppingListItem::loadAllUserArray();
                    foreach ($aAllUsers as $iKey => $sCurrentUser){
                        if($sCurrentUser != $sUserIdCreator){
                            echo("<label for='create".$sCurrentUser."'><br/><input type='checkbox'
class='filled-in' id='create".$sCurrentUser."' name='aUserCredentialsAddItem[]' value='".$sCurrentUser."' />");
                            echo("<span>".$sCurrentUser."</span><br/></label>");
                        }
                    }
                    ?>
                    </div>
                    <input class="btn" type="submit" value="Item zur Liste hinzufügen"/>
                    <input type="hidden" name="action" value="addItem"/>
                </form>
            </div>
        </div>
    <?php
    }

    function checkIfRightCredentials(string $sUsernameAttempt, string $sPasswordAttempt):bool{
        $sSelectQuery = "SELECT userpassword FROM user WHERE username='".$sUsernameAttempt."';";
        $oResult = Database::selectQuery($sSelectQuery);
        if($oResult && $oResult->num_rows > 0){
            $aRow = $oResult->fetch_assoc();
            if(password_verify($sPasswordAttempt, $aRow["userpassword"])){
                return true;
            }else{
                return false;
            }
        }
        return false;
    }

    function registerUser($sUsername, $sPassword):int{
        $sHashedPassword = password_hash($sPassword, PASSWORD_DEFAULT);
        $sInsertQuery = "INSERT INTO user(username, userpassword) VALUES ('".$sUsername."',
'".$sHashedPassword."');";
        $iResult = Database::insertQuery($sInsertQuery);
        return $iResult;
    }

    function registerProduct(DateTime $dCreation, string $sUserIdCreator, string $sTitle,
                             DateTime $dLastMod = NULL,
                             string $sUserIdLastMod = "",
                             bool $bState = true,
                             string $sDescr = "Item hat keinen Beschreibungstext"){
        if($sUserIdLastMod == ""){
            $sUserIdLastMod = $sUserIdCreator;
        }
        $sInsertQuery = "INSERT INTO product(creationDate, usernameCreator, title, descr, lastModDate,
usernameLastMod, state) VALUES (
         '".date_format($dCreation, 'Y-m-d')."', '".$sUserIdCreator."', '".$sTitle."', '".$sDescr."' ,
'".date_format($dLastMod, 'Y-m-d')."', '".$sUserIdLastMod."', '".$bState."');";
        $iResult = Database::insertQuery($sInsertQuery);


        $sInsertQuery = "INSERT INTO user_task(username, id) VALUES ('".$sUserIdCreator."', '".$iResult."');";
        $iResult = Database::insertQuery($sInsertQuery);
    }

    function printShoppingListForSpecificUser($sUsername){
        $sSelectQuery = "SELECT DISTINCT * FROM user INNER JOIN user_task USING(username) INNER JOIN product
USING (id) WHERE username='".$sUsername."';";

        $oResult = Database::selectQuery($sSelectQuery);
        $sPrintResult = "<h5><span class='center light-blue-text'><i class='material-
icons'>add_shopping_cart</i></span>Shopping List</h5><ul class='collapsible'>";
        if($oResult && $oResult->num_rows > 0) {
            while ($aRow = $oResult->fetch_assoc()) {
                $oShoppingItem = new ShoppingListItem($aRow["id"], new DateTime($aRow["creationDate"]),
$aRow["usernameCreator"], $aRow["title"], new DateTime($aRow["lastModDate"]),
                    $aRow["usernameLastMod"], $aRow["state"], $aRow["descr"]);
                $sPrintResult.=$oShoppingItem;
            }
        }
        echo $sPrintResult."</ul>";
    }

    function updateItemData($oShoppingListItem){
        $sUpdatedId = $oShoppingListItem->sId;
        $sUpdatedTitle = $oShoppingListItem->sTitle;
        $sUpdatedDescr = $oShoppingListItem->sDescr;
        $sUpdatedLastModDAte = date_format(new DateTime(), 'Y-m-d');
        $sUpdatedUsernameLastMod = $_SESSION["username"];
        $bUpdatedState = $oShoppingListItem->bState;
        $sUpdateQuery = "UPDATE product SET title = '".$sUpdatedTitle."', descr='".$sUpdatedDescr."',
lastModDate='".$sUpdatedLastModDAte."
        ', usernameLastMod='".$sUpdatedUsernameLastMod."', state = '".$bUpdatedState."' WHERE
id=".$sUpdatedId.";";
```

```
        $oResult = Database::updateQuery($sUpdateQuery);
    }

    function deleteUserFromProduct($sUsername, $iProductId){
        $oDeleteQuery = "DELETE FROM user_task WHERE id='".$iProductId."' and username='".$sUsername."';";
        $bResult = Database::deleteQuery($oDeleteQuery);
    }

    function addUserToProduct($sUsername, $iProduktId){
        $oInsertQuery = "INSERT INTO user_task (id, username) VALUES (".$iProduktId.", '".$sUsername."')";
        $iResult = Database::insertQuery($oInsertQuery);
    }

/**
 * this method manages all the sessions. therefore it hast to read from $_SESSION and $_REQUEST
 */
    function manageSession(){
        global $oShoppingList;
        if(!isset($_SESSION["username"])){
            //user filled out form and wants to login
            if(isset($_REQUEST["action"]) && $_REQUEST["action"] == "login"){
                $sUsernameAttempt = $_REQUEST["username"];
                $sPasswordAttempt = $_REQUEST["password"];
                if(checkIfUserIsAlreadyRegistered($sUsernameAttempt)){
                    if(checkIfRightCredentials($sUsernameAttempt, $sPasswordAttempt)){
                        $_SESSION["username"] = $sUsernameAttempt;
                        echo("<h5>Hallo, ".$_SESSION["username"]."</h5>");
                        printShoppingListForSpecificUser($_SESSION["username"]);
                        printLogoutForm();
                        printCreateNewItemForm($_SESSION["username"]);
                    }else{
                        echo("<h5>Fehler: falsches Passwort!</h5><p class='btn-floating btn-large red pulse'><i
class='material-icons'>error</i></p>");
                        printLoginForm();
                        printRegistrationForm();
                    }
                }else{
                    echo("<h5>Fehler: kein Nutzer mit diesem Namen vorhanden!</h5><p class='btn-floating btn-
large red pulse'><i class='material-icons'>error</i></p>");
                    printLoginForm();
                    printRegistrationForm();
                }
            }else{
                if(isset($_REQUEST["action"]) && $_REQUEST["action"] == "register"){
                    $sUsernameAttempt = $_REQUEST["username"];
                    $sPasswordAttempt = $_REQUEST["password"];
                    $sPasswordRepeatAttempt = $_REQUEST["passwordRepeat"];
                    $sEmail = $_REQUEST["email"];

                    if(!checkIfUserIsAlreadyRegistered($sUsernameAttempt) &&
checkIfInputIsLongEnough($sUsernameAttempt)){
                        //username is okay
                        if(checkIfPasswordIsValid($sPasswordAttempt, $sPasswordRepeatAttempt)){
                            //password is okay
                            if(checkIfEmailIsValid($sEmail)){
                                //email is okay
                                $_SESSION["userCredentials"][$sUsernameAttempt] = md5($sPasswordAttempt);
                                $iResult = registerUser($sUsernameAttempt, $sPasswordAttempt);
                                if($iResult != -1){
                                    echo("<h5>Registrierung erfolgreich</h5><p class='btn-floating btn-large
green'><i class='material-icons'>done</i></p>");
                                }else{
                                    echo("<h5>Registrierung nicht erfolgreich!</h5><p class='btn-floating btn-
large red pulse'><i class='material-icons'>error</i></p>");
                                }
                                printLoginForm();
                                printRegistrationForm();
                            }else{
                                echo("<h5>Fehler: ungültige E-Mail Adresse!</h5><p class='btn-floating btn-
large red pulse'><i class='material-icons'>error</i></p>");
                                printLoginForm();
                                printRegistrationForm($sUsernameAttempt);
                            }
                        }else{
                            echo("<h5>Fehler: falsches Passwort!</h5><p class='btn-floating btn-large red
pulse'><i class='material-icons'>error</i></p>");
                            printLoginForm();
                            printRegistrationForm($sUsernameAttempt, $sEmail);
                        }
                    }else{
                        if(strlen($sUsernameAttempt)< INPUT_MIN_LENGTH){
                            echo("<h5>Fehler: zu kurzer Username!</h5><p class='btn-floating btn-large red 
pulse'><i class='material-icons'>error</i></p>");
                        }else{
                            echo("<h5>Fehler: der Benutzername ist nicht mehr frei, wählen Sie einen 
anderen!</h5><p class='btn-floating btn-large red pulse'><i class='material-icons'>error</i></p>");
                        }
                        printLoginForm();
                        printRegistrationForm("", $sEmail);
                    }
                }else{
                    printLoginForm();
                    printRegistrationForm();
                }
            }
        }else {
            //there is a session with a username
            if (isset($_REQUEST["action"]) && $_REQUEST["action"] == "logout") {
                unset($_SESSION["username"]);
                printLoginForm();
                printRegistrationForm();
            } else {
                if(isset($_REQUEST["action"]) && $_REQUEST["action"] == "addItem"){
                    //$sFormItemId = $_REQUEST["sId"];
                    $sFormItemTitle = $_REQUEST["sTitle"];
                    $sFormItemDescr = $_REQUEST["sDescr"];
                    if($sFormItemDescr == ""){
                        $sFormItemDescr = "Item has no Description";
                    }
                    $aFormItemSharedUserIds = $_REQUEST["aUserCredentialsAddItem"];
                    $dFormItemCreation = new dateTime();
                    $sFormItemUserIdLastMod = $_SESSION["username"];
                    $bFormItemState = true;
                    $dFormItemLastMod = new dateTime();
                    if(!checkIfInputIsLongEnough($sFormItemTitle)){//description can be empty and will not be 
checked
                        echo(printShoppingListForSpecificUser($_SESSION["username"]));
                        printLogoutForm();
                        echo("<h5>Fehler: bitte füllen Sie das Feld 'Titel' auch aus!</h5><p class='btn-
floating btn-large red pulse'><i class='material-icons'>error</i></p>");
```

```
                                        printCreateNewItemForm($_SESSION["username"], "", $sFormItemDescr);
                                }else{
                                        echo("<h5>Hallo, " . $_SESSION["username"] . "</h5><h6>Dein Produkt wurde erfolgreich
angelegt!</h6>");

                                        //registering the product with sql statement
                                        registerProduct($dFormItemCreation, $_SESSION["username"], $sFormItemTitle,
$dFormItemLastMod,
                                            $sFormItemUserIdLastMod, $bFormItemState, $sFormItemDescr,
$aFormItemSharedUserIds);


                                        $aAddUsersToThisProduct = $_REQUEST["aUserCredentialsAddItem"];
                                        $iIdOfCurrentlyRegisteredProduct = ShoppingListItem::getIdOfProduct($sFormItemTitle,
$sFormItemDescr);
                                        foreach ($aAddUsersToThisProduct as $iKey => $sSelectedUser){
                                            addUserToProduct($sSelectedUser, $iIdOfCurrentlyRegisteredProduct);
                                        }
                                        echo("<h5></h5>");
                                        echo(printShoppingListForSpecificUser($_SESSION["username"]));
                                        printLogoutForm();
                                        printCreateNewItemForm($_SESSION["username"]);
                                }
                            }else{
                                if(isset($_REQUEST["action"]) && $_REQUEST["action"] == "editItem"){
                                        $sEditItemId = $_REQUEST["itemId"];

                                        $oOldItem = $oShoppingList->getItemById($sEditItemId);
                                        if($oOldItem==null){//to avoid nullpointer
                                            $dEditCreation = new datetime();
                                            $sEditUserIdCreator = $_SESSION["username"];
                                        }else{
                                            $dEditCreation = $oOldItem->dCreation;
                                            $sEditUserIdCreator = $oOldItem->sUserIdCreator;
                                        }

                                        $sEditItemDescr = $_REQUEST["sDescr"];
                                        $sEditItemTitle = $_REQUEST["sTitle"];
                                        $dEditLastMod = new DateTime();//current time
                                        $sEditLastModUser = $_SESSION["username"];
```

```
                        $aEditUserIds = @$_REQUEST["aUserIds"];
                        if(!isset($_REQUEST['sState'])){
                            $bEditState = true;
                        }else{
                            $bEditState= false;
                        }


                        $aAllUsers = ShoppingListItem::loadAllUserArray();
                        $aPrevUsers = ShoppingListItem::loadSharedUserArray($sEditItemId);

                        $sCreatorOfProduct = ShoppingListItem::getCreatorOfProduct($sEditItemId);

                        foreach ($aAllUsers as $iKey => $oCurrentUser){
                            if(@in_array($oCurrentUser, $aPrevUsers) && @!in_array($oCurrentUser,
$aEditUserIds)){
                                //delete old one
                                if($oCurrentUser != $sCreatorOfProduct){//active user is not in array and must
not be deleted
                                    deleteUserFromProduct($oCurrentUser, $sEditItemId);
                                }
                            }
                            if(@!in_array($oCurrentUser, $aPrevUsers) && @in_array($oCurrentUser,
$aEditUserIds)){
                                //add new one
                                addUserToProduct($oCurrentUser, $sEditItemId);
                            }
                        }



                        $oEditItem = new ShoppingListItem($sEditItemId, $dEditCreation, $sEditUserIdCreator,
$sEditItemTitle,
                        $dEditLastMod, $sEditLastModUser, $bEditState, $sEditItemDescr, $aEditUserIds);
                        updateItemData($oEditItem);
                    }
                    echo("<h5>Hallo, " . $_SESSION["username"] . "</h5>");
                    echo(printShoppingListForSpecificUser($_SESSION["username"]));
                    printLogoutForm();
                    printCreateNewItemForm($_SESSION["username"]);
                    }
               }
          }
     }
?>
</div>
</body>
</html>
```

ShoppingList.php

```php
<?php
/**
 * Created by PhpStorm.
 * User: S1610456027
 * Date: 04.05.2018
 * Time: 07:42
 */

class ShoppingList{//Author: Daniel Pühringer

    private $aEntries;//stores all items

    public function __construct(){
        $this->aEntries = array();
    }

    /**
     * @param $oItem item to add
     */
    public function addItem($oItem){
        array_push($this->aEntries, $oItem);
    }

    /**
     * @param ShoppingListItem $oItemToEdit the item which needs to be edited
     * @param $mAttributeToEdit the value of this attribute gets changed
     * @param $mValueToEdit this is the new value of the changed attribute
     * @throws Exception if the attribute is not found, or the object is null
     */
    public function editItem(ShoppingListItem $oItemToEdit, $mAttributeToEdit, $mValueToEdit){//TODO deprecated
        try{
            if(property_exists("ShoppingListItem", $mAttributeToEdit)){
                if(is_null($oItemToEdit)){
                    throw new Exception("The given object is null!");
                }else{
                    $oItemToEdit->$mAttributeToEdit = $mValueToEdit;
                }
            }else{
                throw new Exception("Setter-Attribute ".$mAttributeToEdit." does not exist!");
            }
        }catch(Exception $e){
            echo("Exception: ". $e->getMessage());
        }
    }

    public function __get($sName){
        return $this->{$sName};
    }//public function __get($sName)


    /**
     * @param ShoppingListItem $oItemToDelete this item gets deleted from shoppingList
     * @throws Exception if the element is not found in the list
     */
    public function deleteItem(ShoppingListItem $oItemToDelete){
        foreach ($this->aEntries as $iKey => $oCurrentItem){
            if($oCurrentItem == $oItemToDelete){
                array_splice($this->aEntries, $iKey, 1);
                return;
            }
        }
        throw new Exception("The item is not in the shopping list!");
    }

    public function checkIfUserCanSeeOtherSharedItems($oItem, $sUserId){
        $aSharedUserForItem = $oItem->aSharedUserIds;
        foreach ($aSharedUserForItem as $iKey => $mValue){
            if($mValue == $sUserId){
                return true;
            }
        }
        return false;
    }

    /**
     * @return string returns all stored items of the list
     */
    public function  __toString():string{
        $sResult = "<h5><span class='center light-blue-text'><i class='material-
icons'>add_shopping_cart</i></span>Shopping List</h5><ul class='collapsible'>";
        foreach ($this->aEntries as $iKey => $oCurrentItem){
            $sResult.= $oCurrentItem;
        }
        return $sResult."<br/><br/></ul>";
    }

    public function printShoppingListForSpecificUser($sIdCreator){
        $sResult = "<h5><span class='center light-blue-text'><i class='material-
icons'>add_shopping_cart</i></span>Shopping List</h5><ul class='collapsible'>";
        //get rows
        foreach ($this->aEntries as $iKey => $oCurrentItem){
            if($oCurrentItem->sUserIdCreator == $sIdCreator || $this->checkIfUserCanSeeOtherSharedItems($oCurrentItem,
$sIdCreator)){
                $sResult.= $oCurrentItem;
            }
        }
        return $sResult."<br/><br/></ul>";
    }

    public function getItemById(string $sItemIdOfSearchedItem){
        $oSearchedItem = "";
        foreach ($this->aEntries as $iKey => $oCurrentItem){
            if($oCurrentItem->sId == $sItemIdOfSearchedItem){
             return $oCurrentItem;
            }
        }
        return $oSearchedItem;
    }

    public function refreshObject($oShoppinglistItem){
        for($i = 0; $i < count($this->aEntries); $i++){
            if($oShoppinglistItem->sId == $this->aEntries[$i]->sId){
                $this->aEntries[$i] = $oShoppinglistItem;
                return;
            }
        }
    }
```

```
}
```

ShoppingListItem.php

```php
<?php
/**
 * Created by PhpStorm.
 * User: S1610456027
 * Date: 03.05.2018
 * Time: 20:14
 */

class ShoppingListItem{//Author: Daniel Pühringer

    private $sId;
    private $dCreation;
    private $sUserIdCreator;
    private $sTitle;
    private $sDescr;
    private $dLastMod;
    private $sUserIdLastMod;
    private $bState;

    private $aSharedUserIds;


    /**
     * ShoppingListItem constructor.
     * @param string $sId id of the item
     * @param date $dCreation date of creation
     * @param string $sUserIdCreator name of user who created this item
     * @param string $sTitle title of the item
     * @param date $dLastMod date of last modification
     * @param string $sUserIdLastMod user of last modification
     * @param bool $bState state(active or not active) of the item; true->item is active; false->item is not active
     * @param string $sDescr is an optional parameter description of the item --> is the only optional attribute of
this class!
     */
    public function __construct(string $sId, DateTime $dCreation, string $sUserIdCreator, string $sTitle,
                                DateTime $dLastMod = NULL,
                                string $sUserIdLastMod ="tutor",
                                bool $bState = true,
                                string $sDescr = "Item hat keinen Beschreibungstext",
                                $aSharedUserIds = array()){

        $this->sId = $sId;
        $this->dCreation = $dCreation;
        $this->sUserIdCreator = $sUserIdCreator;
        $this->sTitle = $sTitle;
        if($dLastMod == NULL){
            $this->dLastMod = new DateTime();
        }else{
            $this->dLastMod = $dLastMod;
        }
        $this->sUserIdLastMod = $sUserIdLastMod;
        $this->bState = $bState;
        $this->sDescr = $sDescr;
        $this->aSharedUserIds = $aSharedUserIds;//TODO
    }

    public function addUserForSharing($sUserId){
        array_push($this->aSharedUserIds, $sUserId);
    }

    /**
     * @return string of current item
     */
    public function __toString(): string
    {
        $sResult = "<li class='shoppingListItem'>";
        if($this->bState){
            $sResult.="<div class='collapsible-header'><span class='center light-green-text'><i class='material-
icons'>shopping_basket</i></span>".$this->sId.": ".$this->sTitle."</div>";
        }else{
            $sResult.="<div class='collapsible-header'><i class='material-icons'>shopping_basket</i>".$this->sId.":
".$this->sTitle."</div>";
        }
        $sResult .= "<div class='collapsible-body'><form>";
        if(!$this->bState){//items which are still active should not be displayed as checked
            $sResult .= "<label for='".$this->sTitle."'><input type='checkbox' class='filled-in' id='".$this->sTitle."'
name='sState' value='notActive' checked='checked'/>";
            $sResult .= "<span>Erledigt</span></label>";
        }else{
            $sResult .= "<label for='".$this->sTitle."'><input type='checkbox' class='filled-in' id='".$this->sTitle."'
name='sState' value='active' />";
            $sResult .= "<span>Noch nicht erledigt</span></label>";
        }

        $sResult .= "<p>Erstellt am " . date_format($this->dCreation, "Y/m/d");
        $sResult .= " von " . $this->sUserIdCreator . "</p>";

        if($this->dLastMod != NULL && self::isSecondDateAfterFirstDate($this->dCreation, $this->dLastMod)){
            $sResult.="<p>Zuletzt geändert am ".date_format($this->dLastMod, "Y/m/d");
            $sResult.=" von ".$this->sUserIdLastMod."</p>";
        }
        $sResult.="<p>Das Item können folgende User noch sehen: <br/>";
        $aSharedUserArray = self::loadSharedUserArray($this->sId);
        $aAllUserArray = self::loadAllUserArray();
        if($aAllUserArray == null || $aSharedUserArray == null){
            return null;
        }
        foreach($aAllUserArray as $iKey => $oCurrentUser){
            $bPrintCurrentUserAsChecked = false;
            if(in_array($oCurrentUser, $aSharedUserArray)){
                $bPrintCurrentUserAsChecked = true;
            }
            if($bPrintCurrentUserAsChecked){
                if($oCurrentUser != $_SESSION["username"] && $oCurrentUser != $this->sUserIdCreator){
                    $sResult .= "<label for='".$oCurrentUser."_".$this->sId."'><input type='checkbox' class='filled-in'
id='".$oCurrentUser."_".$this->sId."' name='aUserIds[]' value='".$oCurrentUser."' checked='checked'/>";
                    $sResult .= "<span>".$oCurrentUser."</span></label><br/>";
                }else {
                    if ($oCurrentUser != $this->sUserIdCreator) {
                        $sResult .= "<label for='".$oCurrentUser."_".$this->sId."'><input type='checkbox'
class='filled-in' id='".$oCurrentUser."_".$this->sId."' name='aUserIds[]' value='" . $oCurrentUser . "'
checked='checked'/>";
                        $sResult .= "<span>" . $oCurrentUser . " (Das sind Sie!)</span></label><br/>";
                    } else {
                        //No printing because it is the creator of the item!
                    }
                }
            }else{
                if($oCurrentUser != $this->sUserIdCreator){
                    $sResult .= "<label for='".$oCurrentUser."_".$this->sId."'><input type='checkbox' class='filled-in'
id='".$oCurrentUser."_".$this->sId."' name='aUserIds[]' value='".$oCurrentUser."'/>";
                    $sResult .= "<span>".$oCurrentUser."</span></label><br/>";
                }else{
                    //No printing because it is the creator of the item!
                }
            }
        }
        $sResult.="</p>";
        $sResult.= "<label for='".$this->sTitle."'>Titel</label>";
        $sResult.= "<input type='text' value='".$this->sTitle."' name='sTitle' id='".$this->sTitle."'/>";
        $sResult.= "<br/>";

        $sResult.= "<label for='".$this->sDescr."'>Beschreibung</label>";
        $sResult.= "<input type='text' value='".$this->sDescr."' name='sDescr' id='".$this->sDescr."'/>";
        $sResult.= "<br/>";


        $sResult.= "<input type='hidden' name='itemId' value='".$this->sId."'/>";
        $sResult.= "<input type='hidden' name='action' value='editItem'/>";
        $sResult.= "<input class='btn' type='submit' value='Änderungen speichern' />";

        return $sResult."</form></div></li>";
    }


    /** loads the array of shared users for a given product id
     * @param int $sId
     * @return array
     */
    public static function loadSharedUserArray(int $sId){
        $sSelectQuery = "SELECT username FROM user_task INNER JOIN product USING(id) WHERE id = '".$sId."';";
        $oResult = Database::selectQuery($sSelectQuery);
        $aResultArrayWithNamesOfSharedUsers = array();
        if($oResult && $oResult->num_rows > 0) {
            while ($aRow = $oResult->fetch_assoc()) {
                array_push($aResultArrayWithNamesOfSharedUsers, $aRow["username"]);
            }
        }
        return $aResultArrayWithNamesOfSharedUsers;
    }

    /** returns all users which are stored in the database within the table "user"
     * @return array
     */
    public static function loadAllUserArray(){
        $sSelectQuery = "SELECT username FROM user;";
        $oResult = Database::selectQuery($sSelectQuery);
        $aResultArrayWithAllRegisteredUsers = array();
        if($oResult && $oResult->num_rows > 0) {
            while ($aRow = $oResult->fetch_assoc()) {
                array_push($aResultArrayWithAllRegisteredUsers, $aRow["username"]);
            }
        }
        return $aResultArrayWithAllRegisteredUsers;
    }

    /** gets creator of certain product by productid
     * @param $iId
     * @return string
     */
    public static function getCreatorOfProduct($iId):string{
        $sSelectQuery = "SELECT usernameCreator FROM product WHERE id='".$iId."';";
        $oResult = Database::selectQuery($sSelectQuery);
        if($oResult && $oResult->num_rows > 0) {
            return $oResult->fetch_assoc()["usernameCreator"];
        }
    }

    /** returns id of product which is searched by title and description
     * @param $sTitle
     * @param $sDescr
     * @return int
     */
    public static function getIdOfProduct($sTitle, $sDescr):int{
        $sSelectQuery = "SELECT id FROM product WHERE title='".$sTitle."' AND descr='".$sDescr."';";
        $oResult = Database::selectQuery($sSelectQuery);
        if($oResult && $oResult->num_rows > 0) {
            return $oResult->fetch_assoc()["id"];
        }
    }

    /**
     * @param $name name of the attribute which value should be returned
     * @return mixed value of attribute
     * @throws Exception if attribute does not exist
     */
    public function __get($name){
        if(property_exists("ShoppingListItem", $name)) {
            return $this->$name;
        }else{
            throw new Exception("Getter-Attribute ".$name." does not exist!");
        }
    }

    /**
     * @param $name name of the attribute which needs to be set
     * @param $value value which the attribute should get
     * @throws Exception if the attribute does not exist
     */
    public function __set($name, $value){
        if(property_exists("ShoppingListItem", $name)){
            $this->$name = $value;
        }else{
            throw new Exception("Setter-Attribute ".$name." does not exist!");
        }
    }

    /**
     * @param datetime $oFirstDate
     * @param datetime $oSecondDate
     * @return bool true if the second date is after the first date; otherwise false
     */
    public static function isSecondDateAfterFirstDate(datetime $oFirstDate, datetime $oSecondDate):bool{
        return $oFirstDate < $oSecondDate;
    }

}
```

Database.php

```php
<?php
class Database{
    private static $oMySQLi;

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

    public static function selectQuery(string $sQuery):mysqli_result{
        if(Database::connect()){
            $oResult = Database::$oMySQLi->query($sQuery);
            Database::disconnect(); //TODO: check return value
            return $oResult;
        }
        else return null;
    }
    //returns id of inserted entry or 0 if not inserted
    public static function insertQuery(string $sQuery):int{
        if(Database::connect()){
            $oResult = Database::$oMySQLi->query($sQuery);
            $iID = Database::$oMySQLi->insert_id;
            Database::disconnect(); //TODO: check return value
            return $iID;
        }
        else return -1;
    }

    //updates certain line
    public static function updateQuery(string $sQuery):bool{
        if(Database::connect()){
            $oResult = Database::$oMySQLi->query($sQuery);
            Database::disconnect();
            return true;
        }else{
            return false;
        }
    }

    public static function deleteQuery(string $sQuery):bool{
        if(Database::connect()){
            $oResult = Database::$oMySQLi->query($sQuery);
            Database::disconnect(); //TODO: check return value
            return $oResult;
        }
        else return false;
    }

    private static function connect():bool{
        Database::$oMySQLi = @new mysqli(DB_SERVER, DB_USER, DB_PASSWORD, DB_DATABASE);
        if(Database::$oMySQLi->connect_error){
            return false; //TODO: could throw exception here
        }
        return true;
    }

    private static function disconnect():bool{
        if(Database::$oMySQLi != null){
            return (Database::$oMySQLi->close());
        }
        return true;
    }
}

?>
```