TodoList.php

```php
<?php

class TodoList
{
    public function __construct(
        private User $loginUser,
        private array $entries = []
    ) {
    }

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

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

    //Ein neuer Entry kann von jedem erstellt werden. Jedoch darf
    //der die Entry-ID noch nicht existieren.
    public function addEntry(TodoListItem $entry):void
    {
        if (!array_key_exists($entry->entryId, $this->entries))
            $this->entries[$entry->entryId] = $entry;
        else throw new Exception("Entry " . $entry->entryId . " does not
        exist!");
    }

    //Es ist nur für den Ersteller möglich, seine Einträge zu löschen.
    //Dieser werden hier aus dem Array mittels der Entry-ID gelöscht.
    public function deleteEntry(TodoListItem $entry):void
    {
        if($this->loginUser->userId === $entry->creatorId &&
            array_key_exists($entry->entryId, $this->entries))
            unset($this->entries[$entry->entryId]);
        else throw new Exception("Entry " . $entry->entryId . " cant be
        deleted!");
    }
```

```
    //Es ist hier wieder nur für den Ersteller möglich, seine
    //Einträge zu bearbeiten. Er kann hier den Titel sowie
    //den Text seines Eintrages ändern. Zudem ändert sich
    //dann auch die Editor-ID und das Edit-Datum auf das
    //heutige Datum.
      public function editEntry(TodoListItem $entry, string $title, string
      $text):void
    {
        if($this->loginUser->userId === $entry->creatorId &&
            array_key_exists($entry->entryId, $this->entries)) {
            $obj = $this->entries[$entry->entryId];
            $obj->title = $title;
            $obj->text = $text;
            $obj->editorId = $this->loginUser->userId;
            $obj->editDate = new DateTime();
        }
      else throw new Exception("Entry " . $entry->entryId . " cant be
      edited!");
    }

    //Es ist hier wieder nur für den Ersteller möglich, seine
    //Einträge abzuschließen. Hier ändert sich der Status.
    public function finishEntry(TodoListItem $entry):void
    {
        if($this->loginUser->userId === $entry->creatorId &&
            array_key_exists($entry->entryId, $this->entries))
            $entry->status = "abgehakt";
        else throw new Exception("Entry " . $entry->entryId . " cant be
        finished!");
    }

    //Magic Method __toString() für die Ausgabe der
    //TodoListe für den gerade angemeldeten User
    public function __toString():string
    {
        $result="<input type='submit' class='add' value='+ neuen Eintrag
        hinzufügen'>";
         $result.="<div class='welcome'>\n
         <h1>ToDoList</h1>\n
         <h2>Willkommen ".$this->loginUser->userId."!</h2>\n
         </div>\n";
         $result.="<div class='toDoList'>\n";

         foreach($this->entries as $entry) {
             $result .= $entry;
             if ($entry->creatorId === $this->loginUser->userId) {
                 $result .= "<div class='buttons'>\n<input type='submit'
                 class='edit' value='Edit'>";
                 $result .= "<input type='submit' class='delete'
                 value='Delete'>";
                 $result .= "<input type='submit' class='finish'
                 value='Finish'>\n</div>";
             }
             $result .= "\n</div>";
         }

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

TodoListItem.php

```php
<?php


class TodoListItem
{
    public function __construct(
        private string $entryId,
        private DateTime $creationDate,
        private string $creatorId,
        private string $title,
        private DateTime $editDate,
        private string $text = "",
        private string $status = "aktiv",
        private string $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'><p>Status: ".$this->status."</p></div>
            <h3>Titel: ".$this->title."</h3>";

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

        $result .= "<p>Ersteller: ".$this->creatorId."</p>
            <p>Erstellungsdatum: ".$this->creationDate-
            >format("d.m.Y")."</p>";
```

```
        //Erst wenn eine Bearbeitung stattfand soll auch das
        // Bearbeitungsdatum und die ID des Bearbeiters angezeigt werden
        if($this->editorId != "") {
            $result .= "<p>Letzter Bearbeiter: " . $this->editorId .
            "</p>";
            $result .= "<p>Letztes Bearbeitungsdatum: " . $this->editDate-
            >format("d.m.Y") . "</p>";
        }
        return $result;
    }
}
```

User.php

```
<?php


class User
{
    public function __construct(
        private string $userId,
        private string $name
    )
    {
    }

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

    public function __set(string $user, mixed $newValue): void
    {
        if (property_exists('User', $user)) {
            $this->{$user};
        } else throw new Exception("Attribute " . $user . " does not exist
in class User!");
    }
}
```

Index.php

```
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>ToDoList</title>
    <link rel="stylesheet" type="text/css" href="style.css"/>
</head>
<body>
<?php

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

try{
    //Neue Benutzer anlegen
    $user = new User("PlannerGirl3000", "Verena Schickmair");
    $user1 = new User("PhpLover1", "Maxi Mustermann");
    //Ein User ist in der TodoListe angemeldet
    $toDoList = new ToDoList($user);

    //Erstellen der unterschiedlichen Einträge
      $entry = new ToDoListItem("e1", new DateTime("2021-02-01"), $user-
      >userId, "Hausarbeit", new DateTime("2021-02-01"));
      $entry1 = new ToDoListItem("e2", new DateTime("2021-04-05"), $user1-
      >userId, "Hausübung machen", new DateTime("2021-04-02"), "Dringend
      die PHP Hausübung abgeben :D");
      $entry2 = new ToDoListItem("e3", new DateTime("2021-03-06"), $user1-
      >userId, "Yoga", new DateTime("2021-03-02"));
      $entry3 = new ToDoListItem("e4", new DateTime("2021-08-05"), $user-
      >userId, "Prokrastinieren", new DateTime("2021-03-02"), "Ich bin eine
      Notiz, hallo!");

    //Hinzufügen in die TodoListe
    $toDoList->addEntry($entry);
    $toDoList->addEntry($entry1);
    $toDoList->addEntry($entry2);
    $toDoList->addEntry($entry3);

    //Bearbeiten eines Eintrages (Ändern des Title und des Textes)
       $toDoList->editEntry($entry, "Bearbeiteter Titel", "Bearbeiteter
      Text");

    //Einen Eintrag abschließen
    $toDoList->finishEntry($entry);

    //Einen Eintrag löschen
    $toDoList->deleteEntry($entry3);
    $toDoList->deleteEntry($entry);

    //Ausgabe der TodoListe - möglich durch __toString
    echo($toDoList);
```

```
}
catch (Exception $e){
    echo("Caught exception: ".$e->getMessage());
}
?>
</body>
</html>
```

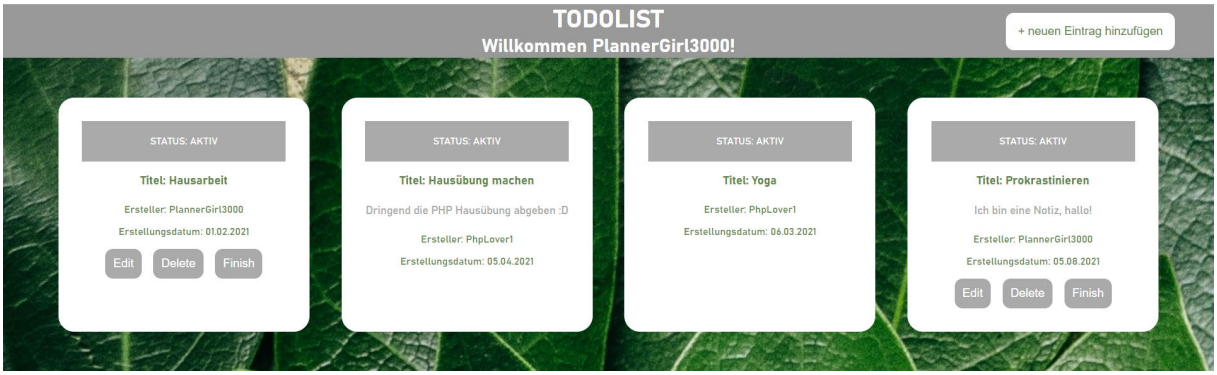
Style.css

```
body{
    font-family: "Bahnschrift", sans-serif;
    margin: 0 auto;
    box-sizing: border-box;
}

.welcome{
    width: 100%;
    background-color: #999;
    text-align: center;
    color: #fff;
}

h1, h2{
    margin: 0;
}

h1{
    top: 5px;
    color: #fff;
    background-color: #999;
    text-transform: uppercase;
}

.add{
    position: absolute;
    right: 50px;
    top: 10px;
    padding: 15px;
    background-color: #fff;
    color: #5e8949;
    font-size: 15px;
    border-radius: 10px;
    border: none;
    cursor: pointer;
}

.toDoList{
    display: flex;
    justify-content: center;
    gap: 3%;
    padding: 50px;
    flex-wrap: wrap;
    background: url("img/bg.jpg");
}
```

```css
.entry{
    text-align: center;
    background-color: #fff;
    color: #5e8949;
    padding: 30px;
    border-radius: 20px;
    width: 20%;
    font-size: 13px;
}

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

.status{
    background-color: #aaa;
    color: #fff;
    padding: 5px;
    text-transform: uppercase;
}

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

.buttons{
    display: flex;
    justify-content: center;
    flex-wrap: wrap;
    gap: 15px;
}

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

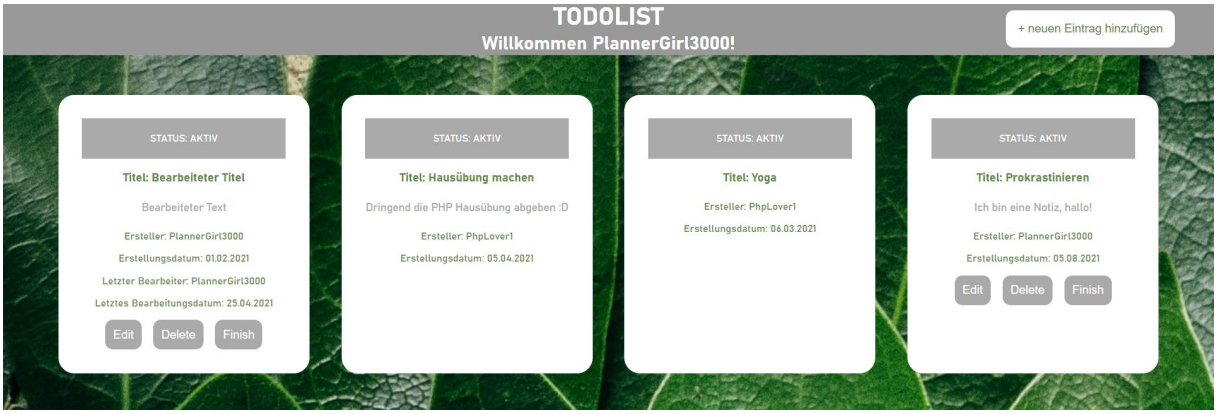
Testfälle

```
    //Hinzufügen in die TodoListe
    $toDoList->addEntry($entry);
    $toDoList->addEntry($entry1);
    $toDoList->addEntry($entry2);
    $toDoList->addEntry($entry3);
```



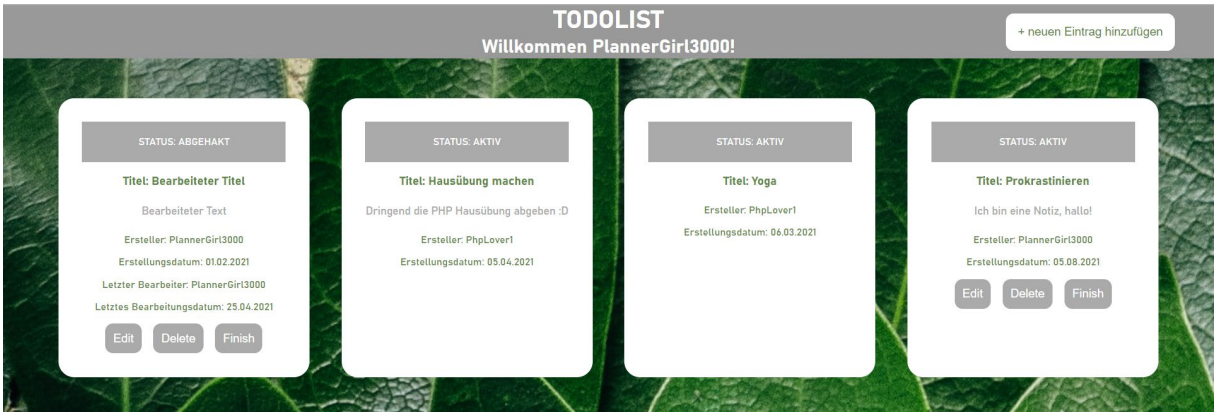

```
//Bearbeiten eines Eintrages (Ändern des Title und des Textes)
$toDoList->editEntry($entry, "Bearbeiteter Titel", "Bearbeiteter Text");
$toDoList->editEntry($entry2, "Das funktioniert nicht", "Gehört nicht zu
mir");
```

➔ Nur das Entry vom Benutzer selbst wurde geändert



```php
//Einen Eintrag abschließen
    $toDoList->finishEntry($entry);
    $toDoList->finishEntry($entry1);
```

➔ Nun ist der Status des ersten Eintrags „abgehakt“




```php
//Einen Eintrag löschen
    $toDoList->deleteEntry($entry3);
    $toDoList->deleteEntry($entry2);
    $toDoList->deleteEntry($entry);
```

➔ Löscht nur die eigenen Einträge