# SCR4x Projektarbeit SS 24

☒ **Gruppe 1** DI (FH) G. Horn-V., MSc
☐ **Gruppe 2** DI (FH) I. Krammer

**Abgabetermin:** Montag, 24.06.2024, 24:00 Uhr

**Name:** Yannick Höß

**Aufwand (h):** 30

## Produktbewertungsportal mit PHP und MySQL

Zu entwickeln ist ein einfaches Produktbewertungsportal, in dem registrierte Benutzer Bewertungen für eine Reihe von Produkten abgeben können, um beliebigen Besuchern des Portals so z. B. bei Kaufentscheidungen zu helfen. Dabei sind die nachstehend beschriebenen Anforderungen zu berücksichtigen bzw. vollständig umzusetzen.

### Funktionale Anforderungen

- Das Durchsuchen der Produkte sowie das Einsehen von abgegebenen Bewertungen muss anonym und somit auch ohne Anmeldung uneingeschränkt möglich sein.
- Für das Anlegen eines neuen Produkts oder das Abgeben einer Bewertung muss sich ein Benutzer zuvor registriert und angemeldet haben.
- Neue Benutzer können sich uneingeschränkt direkt über eine eigene Seite im Portal registrieren. Für jeden Benutzer sind dabei mindestens Benutzername (muss systemweit eindeutig sein) und Passwort über eine sinnvolle Benutzeroberfläche zu erfassen.
- Auf einer Übersichtsseite sollen alle im System gespeicherten Produkte aufgelistet werden. Für jedes Produkt muss mindestens sein Name, sein Hersteller, der Benutzer, der das Produkt angelegt hat, die Anzahl der abgegebenen Bewertungen und die durchschnittliche Bewertung (zwischen 1.0 und 5.0) für das Produkt angezeigt werden. Über ein entsprechendes Formular kann ein angemeldeter Benutzer ein neues Produkt anlegen.
- Durch Anklicken eines Produkts auf der Übersichtsseite gelangt man zu einer Detailansicht, in der alle Bewertungen zu einem Produkt chronologisch absteigend geordnet aufgelistet werden. Für jede Bewertung ist zumindest der Ersteller, das Erstellungsdatum, die eigentliche Bewertung (1, 2, 3, 4 oder 5 in Schulnoten) sowie ein möglicher Kommentar in Textform ansprechend darzustellen. Über ein entsprechendes Formular kann auf einfache Art und Weise eine weitere Bewertung für das Produkt erstellt werden.
- Auf einer Suchseite soll über ein Formular durch Eingabe eines Suchbegriffs (Freitext) gezielt nach Produkten im Forum gesucht werden können. Nach Absetzen einer Suchanfrage sollen alle Produkte angezeigt werden, deren Name oder Hersteller den eingegebenen Suchbegriff enthält. Durch das Anklicken eines Suchergebnisses kann direkt zur Detailseite des entsprechenden Produkts gesprungen werden.
- Ein angemeldeter Benutzer soll seine (und nur seine) abgegebenen Bewertungen im Nachhinein auch noch korrigieren bzw. löschen können.
- Ebenso soll der Ersteller eines Produkts (und nur dieser) die Produktdaten selbst auch nachträglich noch bearbeiten können. Dazu zählt aber natürlich nicht das Bearbeiten oder Löschen von Kommentaren anderer Benutzer… ;)

## Technische Anforderungen

- Die Webseite ist mit PHP und unter Verwendung des Model-View-Controller-Entwurfsmusters zu realisieren.
- Anwendungslogik, Präsentationslogik und Infrastrukturcode müssen nach dem Prinzip der Zwiebelarchitektur sinnvoll und sauber voneinander entkoppelt werden.
- Alle zu speichernden Daten sollen in einer MySQL-Datenbank abgelegt werden. Entwickeln Sie dazu ein entsprechendes Datenbankmodell und dokumentieren Sie es ausführlich (Diagramme etc.).
- Passwörter dürfen nur in ausreichend gesicherter Form in der Datenbank abgelegt werden.
- Die Webseite soll ein einheitliches und ansprechendes Layout bieten und muss auch auf mobilen Endgeräten gut verwendbar sein. Für die Umsetzung dieser Anforderung kann ein entsprechendes UI-Framework wie z. B. Bootstrap verwendet werden.
- Die gesamte Anwendung muss möglichst robust implementiert werden. Diverse Sicherheitsprüfungen dürfen z. B. auch durch das Aufrufen von Skripts mit abgeänderten Parametern oder durch manuell abgesetzte HTTP-Anfragen nicht umgangen werden können.
- Die endgültige Lösung muss auf der in der Übung verwendeten Version von XAMPP betrieben werden können.

## Organisatorische Anforderungen

- Die Projektarbeit ist in Einzelarbeit auszuführen.
- Für die Umsetzung sind grundsätzlich nur die in der Lehrveranstaltung vorgestellten Mittel zu verwenden.
- Die Projektarbeit ist spätestens bis zum oben aufgeführten Abgabetermin im Moodle-Kurs der SCR4-Übung hochzuladen und zum im Stundenplan vorgesehenen Termin in Form einer Live-Demonstration zu präsentieren.

## Abzugebende Komponenten

- Ausführliche Systemdokumentation als PDF-Datei, mindestens mit folgendem Inhalt:
    - Allgemeine Lösungsidee
    - Datenmodell und Erstellungsskript für Datenbank
    - Architektur und Struktur der Webseite
    - Abgedruckter Code der Webseite (HTML-Seiten, PHP-Skripts, Stylesheets)
    - Testfälle inklusive Screenshots
- Ordner der entwickelten Webseite mit allen relevanten Daten (HTML-Seiten, PHP-Skripts, Stylesheets, andere Ressourcen wie Bilder etc.)
- Datenbankskript zur Erstellung der Datenbank als Textdatei
- Datenbankskript zur Erstellung von Testdaten in der Datenbank, mit denen auch bei der Entwicklung bereits getestet wurde, als Textdatei

# 1. Lösungsidee

Grundsätzlich soll die Anwendung ermöglichen, dass Nutzer Produkte erstellen, bewerten und eigens erstellte zu bearbeiten.

## 1.1. Datenmodell und Datenbankstruktur

Die Anwendung verwendet eine SQL-Datenbank mit drei Haupttabellen:

- **Users**: Speichert Informationen zu den Nutzern.
- **Products**: Speichert Informationen zu den Produkten.
- **Rating**: Verbindet Nutzer und Produkte mit Bewertungen und Kommentaren.

Jede Tabelle ist mit Primärschlüsseln und in einigen Fällen mit Fremdschlüsseln versehen, um die Integrität der Daten zu gewährleisten. Zusätzlich hilft Auto-Incremenent, um IDs sicher zu erstellen und nicht im Code mitführen zu müssen.

## 1.2. Backend-Logik und Architektur

Das Backend der Anwendung ist in PHP implementiert, wobei eine klare Trennung zwischen der Präsentationsschicht und der Geschäftslogik besteht. Hier sind einige wichtige Aspekte:

- **Autoloader**: Verwendet für die dynamische Einbindung von Klassen.
- **Service Provider**: Zentrale Komponente für Dependency Injection, die hilft, die Kopplung zwischen den Komponenten zu reduzieren.
- **Controllers**: Verwalten die Anfragen der Nutzer und leiten diese an die entsprechenden Services weiter.

Zudem ist das komplette Design responsive, wodurch die Usability der Anwendung besser gewährleistet wird.

## 1.3. Frontend-Design und Usability

Das Frontend verwendet HTML, CSS und JavaScript, um eine interaktive Benutzeroberfläche zu schaffen. Wichtige Aspekte sind:

- **Navigation**: Einfach und intuitiv, ermöglicht schnellen Zugriff auf alle wichtigen Funktionen der Webseite.
- **Formulare**: Für die Eingabe von Bewertungen, die Registrierung neuer Nutzer und das Anmelden bestehender Nutzer.

Die Ansichten sind alle im Ordner Views zusammengefasst und als .inc Files gespeichert.

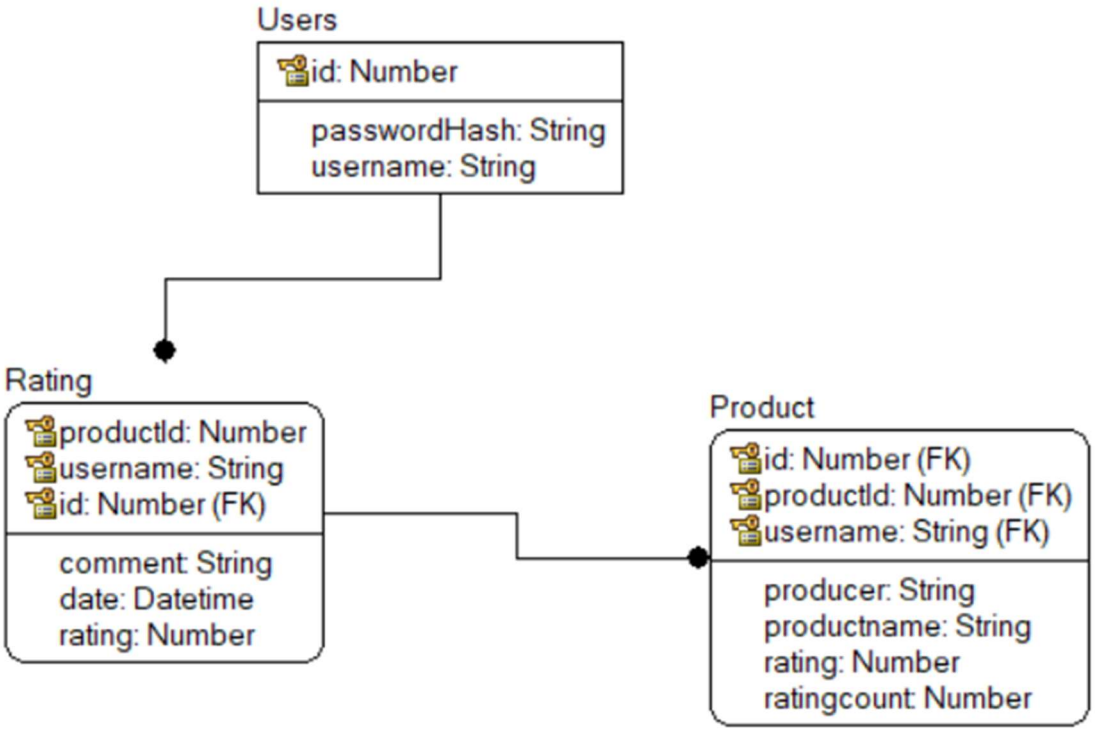
## 1.4. Sicherheit und Datenschutz

Die Anwendung sollte moderne Sicherheitspraktiken implementieren, um Nutzerdaten und -interaktionen zu schützen:

- **SQL-Injection-Schutz** durch Verwendung von Prepared Statements.
- **Password-Hashs** zum Verschlüsseln in der Datenbank. Die Anwendung entziffert die Hash-Werte selbst mittels HTML.

# 2. SQL

## 2.1. Datenmodell




Die Datenbank ist so implementiert die Daten werden nicht so in der Datenbank gespeichert. Da diese Anwendung nicht sehr aufwendig ist, können mit 1-2 Queries die Foreign-Keys weggelassen werden.

## 2.2. Stellungsskrips

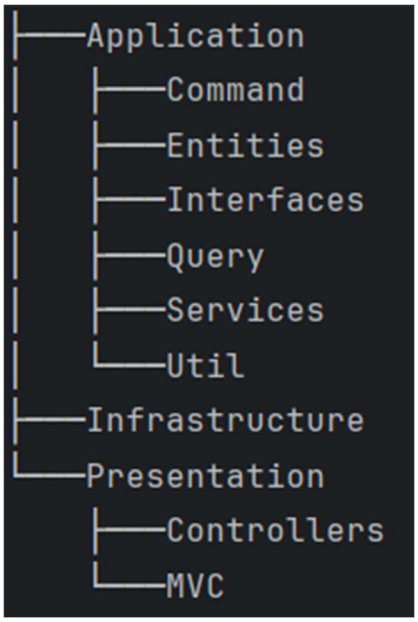
```
--
-- Tabellenstruktur für Tabelle `product`
--

CREATE TABLE `product` (
  `productname` varchar(255) NOT NULL,
  `producer` varchar(255) NOT NULL,
  `username` varchar(255) NOT NULL,
  `ratingcount` int(255) NOT NULL,
  `rating` int(1) NOT NULL,
  `id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Indizes der exportierten Tabellen
--

--
-- Indizes für die Tabelle `product`
--
ALTER TABLE `product`
  ADD PRIMARY KEY (`id`);

--
-- AUTO_INCREMENT für exportierte Tabellen
--

--
-- AUTO_INCREMENT für Tabelle `product`
--
ALTER TABLE `product`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
COMMIT;
```

```
--
-- Tabellenstruktur für Tabelle `users`
--

CREATE TABLE `users` (
  `id` int(255) NOT NULL,
  `passwordhash` varchar(255) NOT NULL,
  `username` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Indizes der exportierten Tabellen
--

--
-- Indizes für die Tabelle `users`
--
ALTER TABLE `users`
  ADD PRIMARY KEY (`id`);

--
-- AUTO_INCREMENT für exportierte Tabellen
--

--
-- AUTO_INCREMENT für Tabelle `users`
--
ALTER TABLE `users`
  MODIFY `id` int(255) NOT NULL AUTO_INCREMENT;
COMMIT;
```

```
--
-- Tabellenstruktur für Tabelle `rating`
--

CREATE TABLE `rating` (
  `username` varchar(255) NOT NULL,
  `date` date NOT NULL,
  `rating` int(1) NOT NULL,
  `comment` varchar(255) DEFAULT NULL,
  `productId` int(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Indizes der exportierten Tabellen
--

--
-- Indizes für die Tabelle `rating`
--
ALTER TABLE `rating`
  ADD PRIMARY KEY (`username`,`productId`);
COMMIT;
```

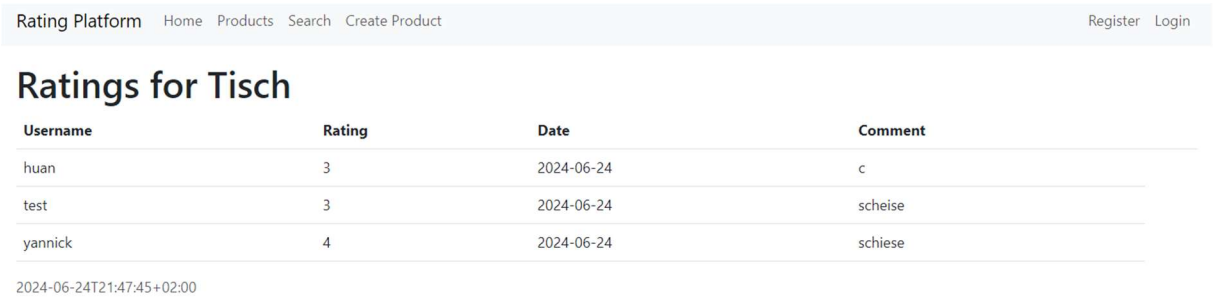
# 3. Architektur

## 3.1. Code




Die Views sind extra in einem eigenen Order zusammengefasst, da diese nicht in den „src“-Ordner gehören.

## 3.2. Website




Man kann auf die Überschrift der Website drücken und kommt auf dieselbe Seite wie Home. In den Products sieht man eine Ansicht, wo man alle Produkte sieht. Im Search-Tab kann man nach Produkten suchen, falls der Filter leer ist, werden alle Produkte angezeigt, da nach keinem gefiltert wird. In Create Produkt kann man ein Produkt erstellen, falls man eingeloggt ist. In Register kann man einen neuen User erstellen, wobei man danach nicht eingeloggt ist. Das muss man dann noch extra im Login-Tab machen.

# 4. Code

C:\xampp\htdocs\ratingportal\index.php

```php
<?php
// === register autoloader
spl_autoload_register(function ($class) {
    $file = __DIR__ . '/src/' . str_replace('\\', '/', $class) . '.php';
    if (file_exists($file)) {
        require_once($file);
    }
});

$sp = new \ServiceProvider();

$sp->register(\Presentation\MVC\MVC::class, function () {
    return new \Presentation\MVC\MVC();
}, isSingleton: true);

// PRESENTATION
// controllers
$sp->register(\Presentation\Controllers\Home::class);
$sp->register(\Presentation\Controllers\Products::class);
$sp->register(\Presentation\Controllers\User::class);
$sp->register(\Presentation\Controllers\Rating::class);


// Application
// querries
$sp->register(\Application\Query\ProductSearchQuery::class);
$sp->register(\Application\Query\SignedInUserQuery::class);
$sp->register(\Application\Query\ProductQuery::class);
$sp->register(\Application\Query\RatingQuery::class);

// commands
$sp->register(\Application\Command\SignInCommand::class);
$sp->register(\Application\Command\SignOutCommand::class);
$sp->register(\Application\Command\RegisterCommand::class);
$sp->register(\Application\Command\CreateProductCommand::class);
$sp->register(\Application\Command\CreateRatingCommand::class);
$sp->register(\Application\Command\UpdateRatingCommand::class);
$sp->register(\Application\Command\DeleteRatingCommand::class);
$sp->register(\Application\Command\IncreaseRatingCommand::class);
$sp->register(\Application\Command\UpdateProductCommand::class);

// services
$sp->register(\Application\Services\AuthenticationService::class);
// $sp->register(\Application\CategoriesQuery::class);
// $sp->register(\Application\BooksQuery::class);
// $sp->register(\Application\BookSearchQuery::class);
// $sp->register(\Application\CartSizeQuery::class);
```

```
// $sp->register(\Application\CheckoutCommand::class);
// $sp->register(\Application\AddBookToCartCommand::class);
// $sp->register(\Application\RemoveBookFromCartCommand::class);
// $sp->register(\Application\SignInCommand::class);
// $sp->register(\Application\SignedInUserQuery::class);
// $sp->register(\Application\SignOutCommand::class);

// $sp->register(\Application\Services\CartService::class);
// $sp->register(\Application\Services\AuthenticationService::class);


// INFRASTRUCTURE
// sessions
$sp->register(\Infrastructure\Session::class, isSingleton: true);
$sp->register(\Application\Interfaces\Session::class,
\Infrastructure\Session::class);

// repository
$sp->register(\Infrastructure\Repository::class, function () {
    return new \Infrastructure\Repository("localhost", "root", "",
"ratingportal");
}, isSingleton: true);
$sp->register(\Application\Interfaces\UserRepository::class,
\Infrastructure\Repository::class);
$sp->register(\Application\Interfaces\ProductRepository::class,
\Infrastructure\Repository::class);
$sp->register(\Application\Interfaces\RatingRepository::class,
\Infrastructure\Repository::class);
// $sp->register(\Application\Interfaces\CategoryRepository::class,
\Infrastructure\Repository::class);
// $sp->register(\Application\Interfaces\BookRepository::class,
\Infrastructure\Repository::class);
// $sp->register(\Application\Interfaces\OrderRepository::class,
\Infrastructure\Repository::class);

$sp->resolve(\Presentation\MVC\MVC::class)->handleRequest($sp);
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\index.php

```
<?php
// === register autoloader
spl_autoload_register(function ($class) {
    $file = __DIR__ . '/src/' . str_replace('\\', '/', $class) . '.php';
    if (file_exists($file)) {
        require_once($file);
    }
});

$sp = new \ServiceProvider();
```

```php
$sp->register(\Presentation\MVC\MVC::class, function(){
    return new \Presentation\MVC\MVC();
}, isSingleton: true);

// PRESENTATION
// controllers
$sp->register(\Presentation\Controllers\Home::class);
$sp->register(\Presentation\Controllers\Books::class);
$sp->register(\Presentation\Controllers\Cart::class);
$sp->register(\Presentation\Controllers\Order::class);
$sp->register(\Presentation\Controllers\User::class);


// APPLICATION
// commands and querries
$sp->register(\Application\CategoriesQuery::class);
$sp->register(\Application\BooksQuery::class);
$sp->register(\Application\BookSearchQuery::class);
$sp->register(\Application\CartSizeQuery::class);
$sp->register(\Application\CheckoutCommand::class);
$sp->register(\Application\AddBookToCartCommand::class);
$sp->register(\Application\RemoveBookFromCartCommand::class);
$sp->register(\Application\SignInCommand::class);
$sp->register(\Application\SignedInUserQuery::class);
$sp->register(\Application\SignOutCommand::class);

$sp->register(\Application\Services\CartService::class);
$sp->register(\Application\Services\AuthenticationService::class);


// INFRASTRUCTURE
// sessions
$sp->register(\Infrastructure\Session::class, isSingleton: true);
$sp->register(\Application\Interfaces\Session::class,
\Infrastructure\Session::class);

// repository
/*
$sp->register(\Infrastructure\FakeRepository::class, isSingleton: true);
$sp->register(\Application\Interfaces\CategoryRepository::class,
\Infrastructure\FakeRepository::class);
$sp->register(\Application\Interfaces\BookRepository::class,
\Infrastructure\FakeRepository::class);
$sp->register(\Application\Interfaces\OrderRepository::class,
\Infrastructure\FakeRepository::class);
$sp->register(\Application\Interfaces\UserRepository::class,
\Infrastructure\FakeRepository::class);
*/
$sp->register(\Infrastructure\Repository::class, function() {
```

```
    return new \Infrastructure\Repository("localhost", "root", "",
"bookshop");
}, isSingleton: true);
$sp->register(\Application\Interfaces\CategoryRepository::class,
\Infrastructure\Repository::class);
$sp->register(\Application\Interfaces\BookRepository::class,
\Infrastructure\Repository::class);
$sp->register(\Application\Interfaces\OrderRepository::class,
\Infrastructure\Repository::class);
$sp->register(\Application\Interfaces\UserRepository::class,
\Infrastructure\Repository::class);

$sp->resolve(\Presentation\MVC\MVC::class)->handleRequest($sp);
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\HelloWorld.php

```
<?php

class HelloWorld{
    public function __construct() {
        echo("Hello World");
    }

    public function render() {
        echo("!");
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\ServiceProvider.php

```
<?php

final class ServiceProvider
{
    private $serviceDescriptors = [];
    private $instances = [];

    private const SERVICE_TYPE = 'type';
    private const SERVICE_FACTORY = 'factory';
    private const SERVICE_IS_SINGLETON = 'singleton';

    public function register(string $serviceType, string|callable|null
$implementation = null, bool $isSingleton = false): void
    {
        $factory = match (true) {
            // implementation is string (name of other service) --> create
this service by resolving this other service
            is_string($implementation) => function () use ($implementation) {
                return $this->resolve($implementation);
            },
```

```
            // implementation is callable (factory function) --> create this
service by calling this factory function
            is_callable($implementation) => $implementation,
            // no implementation provided --> create this service by
instanticating class with service name
            default => function () use ($serviceType) {
                return $this->createInstance($serviceType);
            },
        };
        $this->serviceDescriptors[$serviceType] = [
            self::SERVICE_TYPE => $serviceType,
            self::SERVICE_FACTORY => $factory,
            self::SERVICE_IS_SINGLETON => $isSingleton,
        ];
    }

    public function resolve(string $serviceType): object
    {
        // look up service descriptor
        $sd = $this->serviceDescriptors[$serviceType] ?? null;
        if ($sd === null) {
            throw new \Exception("Service '{$serviceType}' not registered for
dependency injection.");
        }
        // check for existing instance
        $instance = $this->instances[$sd[self::SERVICE_TYPE]] ?? null;
        if ($instance === null) {
            // create instance via factory
            $instance = $sd[self::SERVICE_FACTORY]();
            // store instance when service is singleton
            if ($sd[self::SERVICE_IS_SINGLETON]) {
                $this->instances[$sd[self::SERVICE_TYPE]] = $instance;
            }
        }
        return $instance;
    }

    private function createInstance(string $className): object
    {
        $params = [];
        $ctor = (new \ReflectionClass($className))->getConstructor();
        if ($ctor !== null) {
            foreach ($ctor->getParameters() as $param) {
                $pt = $param->getType();
                if (!$pt instanceof \ReflectionNamedType) {
                    throw new \Exception("Cannot resolve constructor parameter
'{$param->getName()}' for class '$className': Parameter does not have named
type.");
                }
                $params[] = $this->resolve($pt->getName());
```

```
            }
        }
        return new $className(...$params);
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\AddBookToCartCommand.php

```php
<?php

namespace Application;

class AddBookToCartCommand {
    public function __construct(
        private Services\CartService $cartService
    ) { }

    public function execute(int $bookId): void {
        $this->cartService->addBook($bookId);
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\BookData.php

```php
<?php

namespace Application;

readonly class BookData {
    public bool $isInCart;
    public function __construct(
        public int $id,
        public string $title,
        public string $author,
        public float $price,
        public int $cartCount
    ) {
        $this->isInCart = $this->cartCount > 0;
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\BookSearchQuery.php

```php
<?php

namespace Application;

class BookSearchQuery {
    public function __construct(
        private Interfaces\BookRepository $bookRepository,
```

```
        private Services\CartService $cartService
    ) {}

    public function execute(string $filter): array {
      $res = [];

      foreach($this->bookRepository->getBooksForFilter($filter) as $b) {
          $res[] = new BookData($b->getId(), $b->getTitle(), $b->getAuthor(),
$b->getPrice(), $this->cartService->getCountForBook($b->getId()));
        }

      return $res;
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\BooksQuery.php

```
<?php

namespace Application;

class BooksQuery {
    public function __construct(
        private Interfaces\BookRepository $bookRepository,
        private Services\CartService $cartService
    ){}

    public function execute(string $categoryId): array {
        $res = [];
        foreach ($this->bookRepository->getBooksForCategory($categoryId) as
$b) {
            $res[] = new BookData($b->getId(), $b->getTitle(), $b-
>getAuthor(), $b->getPrice(), $this->cartService->getCountForBook($b-
>getId()));
        }
        return $res;
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\CartSizeQuery.php

```
<?php

namespace Application;

class CartSizeQuery {
    public function __construct(
        private Services\CartService $cartService
    ){}

    public function execute(): int {
```

```
        return $this->cartService->getSize();
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\CategoriesQuery.php

```
<?php

namespace Application;

class CategoriesQuery {
    public function __construct(private Interfaces\CategoryRepository $categoryRepository) {

    }

    public function execute(): array {
        $res = [];
        foreach ($this->categoryRepository->getCategories() as $c) {
            $res[] = new CategoryData($c->getId(), $c->getName());
        }
        return $res;
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\CategoryData.php

```
<?php

namespace Application;

readonly class CategoryData {
    public function __construct(
        public int $id,
        public string $name
    ) {}
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\CheckoutCommand.php

```
<?php

namespace Application;

class CheckoutCommand {
    const Error_NotAuthenticated = 0x01;
    const Error_CartEmpty = 0x02;
    const Error_InvalidCreditCardName = 0x04;
    const Error_InvalidCreditCardNumber = 0x08;
    const Error_CreateOrderFailed = 0x10;
```

```
    public function __construct(
        private Interfaces\OrderRepository $orderRepository,
        private Services\CartService $cartService,
        private Services\AuthenticationService $authenticationService
    ) { }

    public function execute(string $ccName, string $ccNumber, ?int &$orderId):
int {
        $ccName = trim($ccName);
        $ccNumber = str_replace(' ', '', $ccNumber);

        $errors = 0;

        $userID = $this->authenticationService->getUserId();
        if ($userID === null) {
            $errors |= self::Error_NotAuthenticated;
        }

        // check for items in cart
        if ($this->cartService->getSize() === 0) {
            $errors |= self::Error_CartEmpty;
        }

        if (strlen($ccName) === 0) {
            $errors |= self::Error_InvalidCreditCardName;
        }
        if (strlen($ccNumber) !== 16 || !ctype_digit($ccNumber)) {
            $errors |= self::Error_InvalidCreditCardNumber;
        }

        if (!$errors) {
            $cart = $this->cartService->getAllBooksWithCount();
            $orderId = $this->orderRepository->createOrder($cart, $ccName,
$ccNumber);

            if ($orderId === null) {
                $errors |= self::Error_CreateOrderFailed;
            }
        }

        return $errors;
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\RemoveBookFromCartCommand.php

```
<?php

namespace Application;
```

```php
class RemoveBookFromCartCommand {
    public function __construct(
        private Services\CartService $cartService
    ) { }

    public function execute(int $bookId): void {
        $this->cartService->removeBook($bookId);
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\SignedInUserQuery.php

```php
<?php

namespace Application;

class SignedInUserQuery {
    public function __construct(
        private Services\AuthenticationService $authenticationService,
        private Interfaces\UserRepository $userRepository
    ) { }

    public function execute(): ?UserData {
        $id = $this->authenticationService->getUserId();
        if ($id === null) {
            return null;
        }

        $user = $this->userRepository->getUser($id);
        if ($user === null) {
            return null;
        }

        return new UserData($user->getId(), $user->getUserName());
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\SignInCommand.php

```php
<?php

namespace Application;

class SignInCommand {
    public function __construct(
        private Services\AuthenticationService $authenticationService,
        private Interfaces\UserRepository $userRepository
    ){ }
```

```
    public function execute(string $userName, string $password): bool {
        $this->authenticationService->signOut();
        $user = $this->userRepository->getUserForUserName($userName);

        if ($user != null && password_verify($password, $user-
>getPasswordHash())) {
            $this->authenticationService->signIn($user->getId());
            return true;
        }

        return false;
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\SignOutCommand.php

```
<?php

namespace Application;

class SignOutCommand {
    public function __construct(
        private Services\AuthenticationService $authenticationService,
        private Services\CartService $cartService,
    ) { }

    public function execute() {
        $this->authenticationService->signOut();
        $this->cartService->clear();
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\UserData.php

```
<?php

namespace Application;

class UserData {
    public function __construct(
        public int $id,
        public string $userName
    ) {}
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\Entities\Book.php

```
<?php

namespace Application\Entities;
```

```php
class Book {
    public function __construct(
        private int $id,
        private string $title,
        private string $author,
        private float $price
    ) {}

    public function getId(): int {
        return $this->id;
    }

    public function getTitle(): string {
        return $this->title;
    }

    public function getAuthor(): string {
        return $this->author;
    }

    public function getPrice(): float {
        return $this->price;
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\Entities\Category.php

```php
<?php

namespace Application\Entities;

class Category {
    public function __construct(
        private int $id,
        private string $name
    ) {

    }

    public function getId(): int {
        return $this->id;
    }

    public function getName(): string {
        return $this->name;
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\Entities\User.php

```php
<?php

namespace Application\Entities;

class User {
    public function __construct(
        private int $id,
        private string $userName,
        private string $passwordHash
    ) {}

    public function getUserName(): string {
        return $this->userName;
    }

    public function getPasswordHash(): string {
        return $this->passwordHash;
    }

    public function getId(): int {
        return $this->id;
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\Interfaces\BookRepository.php

```php
<?php

namespace Application\Interfaces;

interface BookRepository {
    public function getBooksForCategory(int $categoryId): array;
    public function getBooksForFilter(string $filter): array;
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\Interfaces\CategoryRepository.php

```php
<?php

namespace Application\Interfaces;

interface CategoryRepository {
    public function getCategories(): array;
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\Interfaces\OrderRepository.php

```php
<?php

namespace Application\Interfaces;
```

```php
interface OrderRepository {
    public function createOrder(array $books, string $ccName, string
$ccNumber): ?int;
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\Interfaces\Session.php

```php
<?php

namespace Application\Interfaces;

interface Session {
    public function get(String $key): mixed;
    public function put(String $key, mixed $value): void;
    public function delete(String $key): void;

}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\Interfaces\UserRepository.php

```php
<?php

namespace Application\Interfaces;

interface UserRepository {
    public function getUser(int $id): ?\Application\Entities\User;
    public function getUserForUserName(string $userName):
?\Application\Entities\User;
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\Services\AuthenticationService.php

```php
<?php

namespace Application\Services;

class AuthenticationService {
    const SESSION_USER_ID = 'userId';

    public function __construct(
        private \Application\Interfaces\Session $session
    ){ }

    public function signIn(int $userId): void {
        $this->session->put(self::SESSION_USER_ID, $userId);
    }

    public function signOut(): void {
        $this->session->delete(self::SESSION_USER_ID);
```

```
    }

    public function getUserId(): ?int {
        return $this->session->get(self::SESSION_USER_ID);
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Application\Services\CartService.php

```
<?php
namespace Application\Services;

class CartService {
    const SESSION_CART = 'cart';

    public function __construct(private \Application\Interfaces\Session $session) { }

    public function getAllBooksWithCount(): array {
        return $this->getOrCreateCart();
    }

    public function getCountForBook(int $bookId): int {
        $cart = $this->getOrCreateCart();
        return $cart[$bookId] ?? 0;
    }

    public function addBook(int $bookId): void {
        $cart = $this->getOrCreateCart();
        if (!isset($cart[$bookId])) {
            $cart[$bookId] = 1;
        } else {
            $cart[$bookId]++;
        }
        $this->session->put(self::SESSION_CART, $cart);
    }

    public function removeBook(int $bookId): void {
        $cart = $this->getOrCreateCart();
        if (isset($cart[$bookId])) {
            if ($cart[$bookId] > 1) {
                $cart[$bookId]--;
            } else {
                unset($cart[$bookId]);
            }
        }
        $this->session->put(self::SESSION_CART, $cart);
    }

    public function getSize(): int {
```

```
        $size = 0;
        $cart = $this->getOrCreateCart();
        foreach($cart as $bookId => $count) {
            $size += $count;
        }

        return $size;
    }

    public function clear(): void {
        $this->session->delete(self::SESSION_CART);
    }

    private function getOrCreateCart(): array {
        return $this->session->get(self::SESSION_CART) ?? [];
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Infrastructure\FakeRepository.php

```
<?php

namespace Infrastructure;

class FakeRepository
    implements
    \Application\Interfaces\CategoryRepository,
    \Application\Interfaces\BookRepository,
    \Application\Interfaces\OrderRepository,
    \Application\Interfaces\UserRepository
{
    private $mockCategories;
    private $mockBooks;
    private $mockUsers;

    public function __construct()
    {
        // create mock data
        $this->mockCategories = array(
            array(1, 'Mobile & Wireless Computing'),
            array(2, 'Functional Programming'),
            array(3, 'C / C++'),
            array(4, '<< New Publications >>')
        );

        $this->mockBooks = array(
            array(1, 1, 'Hello, Android:\nIntroducing Google\'s Mobile Development Platform', 'Ed Burnette', 19.97),
            array(2, 1, 'Android Wireless Application Development', 'Shane Conder, Lauren Darcey', 31.22),
```

```
            array(5, 1, 'Professional Flash Mobile Development', 'Richard
Wagner', 19.90),
            array(7, 1, 'Mobile Web Design For Dummies', 'Janine Warner, David
LaFontaine', 16.32),
            array(11, 2, 'Introduction to Functional Programming using
Haskell', 'Richard Bird', 74.75),
            // book with title to test scripting attack
            array(12, 2, 'Scripting (Attacks) for Beginners - <script
type="text/javascript">alert(\'All your base are belong to us!\');</script>',
'John Doe', 9.99),
            array(14, 2, 'Expert F# (Expert\'s Voice in .NET)', 'Antonio
Cisternino, Adam Granicz, Don Syme', 47.64),
            array(16, 3, 'C Programming Language\n(2nd Edition)', 'Brian W.
Kernighan, Dennis M. Ritchie', 48.36),
            array(27, 3, 'C++ Primer Plus\n(5th Edition)', 'Stephan Prata',
36.94),
            array(29, 3, 'The C++ Programming Language', 'Bjarne Stroustrup',
67.49)
        );

        $this->mockUsers = array(
            array(1, 'scr4',
'$2y$10$0dhe3ngxlmzgZrX6MpSHkeoDQ.dOaceVTomUq/nQXV0vSkFojq.VG')
        );
    }

    // TODO: implementation
    public function getCategories(): array
    {
        $res = array();
        foreach ($this->mockCategories as $c) {
            $res[] = new \Application\Entities\Category($c[0], $c[1]);
        }
        return $res;
    }

    public function getBooksForCategory(int $categoryId): array
    {
        $res = array();
        foreach ($this->mockBooks as $b) {
            if ($b[1] === $categoryId) {
                $res[] = new \Application\Entities\Book($b[0], $b[2], $b[3],
$b[4]);
            }

        }
        return $res;
    }

    public function getBooksForFilter(string $filter): array
```

```
    {
        $res = array();
        foreach ($this->mockBooks as $b) {
            if ($filter == '' || stripos($b[2], $filter) !== false) {
                $res[] = new \Application\Entities\Book($b[0], $b[2], $b[3],
$b[4]);
            }
        }
        return $res;
    }

    public function createOrder(array $books, string $ccName, string
$ccNumber): ?int {
        return rand();
    }

    public function getUser(int $id): ?\Application\Entities\User {
        foreach ($this->mockUsers as $u) {
            if ($u[0] === $id) {
                return new \Application\Entities\User($u[0], $u[1], $u[2]);
            }
        }
        return null;
    }

    public function getUserForUserName(string $userName):
?\Application\Entities\User {
        foreach ($this->mockUsers as $u) {
            if ($u[1] === $userName) {
                return new \Application\Entities\User($u[0], $u[1], $u[2]);
            }
        }
        return null;
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Infrastructure\Repository.php

```
<?php

namespace Infrastructure;

class Repository
    implements
    \Application\Interfaces\BookRepository,
    \Application\Interfaces\CategoryRepository,
    \Application\Interfaces\OrderRepository,
    \Application\Interfaces\UserRepository
{
```

```
    private $server;
    private $userName;
    private $password;
    private $database;

    public function __construct(string $server, string $userName, string
$password, string $database)
    {
        $this->server = $server;
        $this->userName = $userName;
        $this->password = $password;
        $this->database = $database;
    }

    // === private helper methods ===

    private function getConnection()
    {
        $con = new \mysqli($this->server, $this->userName, $this->password,
$this->database);
        if (!$con) {
            die('Unable to connect to database. Error: ' .
mysqli_connect_error());
        }
        return $con;
    }

    private function executeQuery($connection, $query)
    {
        $result = $connection->query($query);
        if (!$result) {
            die("Error in query '$query': " . $connection->error);
        }
        return $result;
    }

    private function executeStatement($connection, $query, $bindFunc)
    {
        $statement = $connection->prepare($query);
        if (!$statement) {
            die("Error in prepared statement '$query': " . $connection-
>error);
        }
        $bindFunc($statement);
        if (!$statement->execute()) {
            die("Error executing prepared statement '$query': " . $statement-
>error);
        }
        return $statement;
    }
```

```
    public function getBooksForCategory(int $categoryId): array {
        $books = [];

        $con = $this->getConnection();
        $stat = $this->executeStatement(
            $con,
            "SELECT id, title, author, price FROM books WHERE categoryId = ?",
            function($s) use ($categoryId) {
                $s->bind_param('i', $categoryId);
            });
        $stat->bind_result($id, $title, $author, $price);
        while($stat->fetch()) {
            $books[] = new \Application\Entities\Book($id, $title, $author,
$price);
        }
        $stat->close();
        $con->close();

        return $books;
    }

    public function getBooksForFilter(string $filter): array {
        $filter = "%$filter%";
        $books = [];

        $con = $this->getConnection();
        $stat = $this->executeStatement(
            $con,
            "SELECT id, title, author, price FROM books WHERE title LIKE ?",
            function($s) use ($filter) {
                $s->bind_param('s', $filter);
            });
        $stat->bind_result($id, $title, $author, $price);
        while($stat->fetch()) {
            $books[] = new \Application\Entities\Book($id, $title, $author,
$price);
        }
        $stat->close();
        $con->close();

        return $books;
    }

    public function getCategories(): array {
        $categories = [];

        $con = $this->getConnection();
        $res = $this->executeQuery($con, "SELECT id, name FROM categories");
        while ($cat = $res->fetch_object()) {
```

```
            $categories[] = new \Application\Entities\Category($cat->id, $cat->name);
        }
        $res->close();
        $con->close();

        return $categories;
    }

    public function createOrder(array $books, string $ccName, string $ccNumber): ?int {
        $con = $this->getConnection();
        $con->autocommit(false);
        $stat = $this->executeStatement(
            $con,
            "INSERT INTO orders (userId, creditCardHolder, creditCardNumber) VALUES (?, ?, ?)",
            function($s) use ($userId, $creditCardName, $creditCardNumber) {
                $s->bind_param('iss', $userId, $creditCardName, $creditCardNumber);
            }
        );
        $orderId = $stat->insert_id;
        $stat->close();

        foreach ($books as $bookId => $count) {
            for ($i = 0; $i < $count; $i++) {
                $this->executeStatement($con,
                "INSERT INTO orderedBooks (orderId, bookId) VALUES (?, ?)",
                function($s) use ($orderId, $bookId) {
                    $s->bind_param("ii", $orderId, $bookId);
                })->close();
            }
        }
        $con->commit();
        $con->close();
        return $orderId;
    }

    public function getUser(int $id): ?\Application\Entities\User {
        $user = null;
        $con = $this->getConnection();
        $stat = $this->executeStatement($con,
            'SELECT id, userName, passwordHash FROM users WHERE id = ?',
            function($s) use ($id) {
                $s->bind_param('i', $id);
            }
        );
        $stat->bind_result($id, $userName, $passwordHash);
        if ($stat->fetch()) {
```

```
            $user = new \Application\Entities\User($id, $userName,
$passwordHash);
        }
        $stat->close();
        $con->close();
        return $user;
    }

    public function getUserForUserName(string $userName):
?\Application\Entities\User {
        $user = null;
        $con = $this->getConnection();
        $stat = $this->executeStatement($con,
            'SELECT id, userName, passwordHash FROM users WHERE userName = ?',
            function($s) use ($userName) {
                $s->bind_param('s', $userName);
            }
        );
        $stat->bind_result($id, $userName, $passwordHash);
        if ($stat->fetch()) {
            $user = new \Application\Entities\User($id, $userName,
$passwordHash);
        }
        $stat->close();
        $con->close();
        return $user;
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Infrastructure\Session.php

```
<?php

namespace Infrastructure;

class Session implements \Application\Interfaces\Session {
    public function __construct() {
        session_start();
    }

    public function get(string $key): mixed {
        return $_SESSION[$key] ?? null;
    }

    public function put(string $key, mixed $value): void {
        $_SESSION[$key] = $value;
    }

    public function delete(string $key): void {
        unset($_SESSION[$key]);
    }
```

```
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Presentation\Controllers\Books.php

```php
<?php
namespace Presentation\Controllers;

class Books extends \Presentation\MVC\Controller {
    const PARAM_CATEGORY_ID = 'cid';
    const PARAM_FILTER = 'f';

    public function __construct(
        private \Application\CategoriesQuery $categoriesQuery,
        private \Application\BooksQuery $booksQuery,
        private \Application\BookSearchQuery $bookSearchQuery,
        private \Application\SignedInUserQuery $signedInUserQuery
    ) {

    }

    public function GET_Index(): \Presentation\MVC\ActionResult {
        return $this->view('bookList', [
            'user' => $this->signedInUserQuery->execute(),
            'categories' => $this->categoriesQuery->execute(),
            'selectedCategoryId' => $this->tryGetParam(self::PARAM_CATEGORY_ID, $value) ? $value : null,
            'books' => $this->tryGetParam(self::PARAM_CATEGORY_ID, $value) ? $this->booksQuery->execute($value) : null,
            'context' => $this->getRequestUri()
        ]);
    }

    public function GET_Search(): \Presentation\MVC\ActionResult {
        return $this->view('bookSearch', [
            'user' => $this->signedInUserQuery->execute(),
            'filter' => $this->tryGetParam(self::PARAM_FILTER, $value) ? $value : '',
            'books' => $this->tryGetParam(self::PARAM_FILTER, $value) ? $this->bookSearchQuery->execute($value) : null,
            'context' => $this->getRequestUri()
        ]);
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Presentation\Controllers\Cart.php

```php
<?php

namespace Presentation\Controllers;
```

```
class Cart extends \Presentation\MVC\Controller {
    const PARAM_BOOK_ID = 'bid';
    const PARAM_CONTEXT = 'ctx';

    public function __construct(
        private \Application\AddBookToCartCommand $addBookToCartCommand,
        private \Application\RemoveBookFromCartCommand
$removeBookFromCartCommand
    ) {}

    public function POST_Add(): \Presentation\MVC\ActionResult {
        $this->addBookToCartCommand->execute($this-
>getParam(self::PARAM_BOOK_ID));
        //return $this->view('home');
        return $this->redirectToUri($this->getParam(self::PARAM_CONTEXT));
    }

    public function POST_Remove(): \Presentation\MVC\ActionResult {
        $this->removeBookFromCartCommand->execute($this-
>getParam(self::PARAM_BOOK_ID));
        return $this->redirectToUri($this->getParam(self::PARAM_CONTEXT));
    }
}

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-
main\src\Presentation\Controllers\Home.php
<?php
namespace Presentation\Controllers;

class Home extends \Presentation\MVC\Controller {
    public function __construct (
        private \Application\SignedInUserQuery $signedInUserQuery
    ) { }

    public function GET_index() : \Presentation\MVC\ActionResult {
        return $this->view('home', [
            'user' => $this->signedInUserQuery->execute()
        ]);
    }
}

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-
main\src\Presentation\Controllers\Order.php
<?php

namespace Presentation\Controllers;

class Order extends \Presentation\MVC\Controller {
    const PARAM_ORDER_ID = 'oid';
    const PARAM_CARD_NUMBER = 'cn';
```

```
    const PARAM_NAME_ON_CARD = 'noc';
    const PARAM_USER_NAME = 'un';

    public function __construct (
        private \Application\CartSizeQuery $cartSizeQuery,
        private \Application\CheckoutCommand $checkoutCommand,
        private \Application\SignedInUserQuery $signedInUserQuery
    ) {

    }

    public function GET_Create() : \Presentation\MVC\ActionResult {
        $user = $this->signedInUserQuery->execute();
        if ($user === null) {
            return $this->redirect('User', 'Login');
        }

        $cartSize = $this->cartSizeQuery->execute();

        if ($cartSize !== 0) {
            return $this->view('orderForm', [
                'user' => $this->signedInUserQuery->execute(),
                'cartSize' => $cartSize
            ]);
        }

        return $this->view('orderFormEmptyCart', ['user' => $this-
>signedInUserQuery->execute(),]);
    }

    public function POST_Create() : \Presentation\MVC\ActionResult {
        $ccName = $this->getParam(self::PARAM_NAME_ON_CARD);
        $ccNumber = $this->getParam(self::PARAM_CARD_NUMBER);

        $result = $this->checkoutCommand->execute($ccName, $ccNumber,
$orderId);

        echo $ccName;
        echo $ccNumber;
        echo $result;

        if ($result !== 0) {
            if ($result &
\Application\CheckoutCommand::Error_NotAuthenticated) {
                return $this->redirect('Order', 'Create');
            }

            if ($result & \Application\CheckoutCommand::Error_CartEmpty) {
                return $this->redirect('Order', 'Create');
            }
```

```
            $errors = [];
            if ($result &
\Application\CheckoutCommand::Error_InvalidCreditCardName) {
                $errors[] = "Invalid name on card";
            }
            if ($result &
\Application\CheckoutCommand::Error_InvalidCreditCardNumber) {
                $errors[] = "Invalid credit card number (must be 16 digits)";
            }
            if (sizeof($errors) == 0) {
                $errors[] = "Order creation failed";
            }

            return $this->view('orderForm', [
                'user' => $this->signedInUserQuery->execute(),
                "cartSize" => $this->cartSizeQuery->execute(),
                "nameOnCard" => $ccName,
                "cardNumber" => $ccNumber,
                "errors" => $errors
            ]);
        } else {
            return $this->redirect('Order', 'ShowSummary',
[self::PARAM_ORDER_ID => $orderId]);
        }
    }

    public function GET_ShowSummary() : \Presentation\MVC\ActionResult {
        return $this->view('orderSummary', [
            'user' => $this->signedInUserQuery->execute(),
            'orderId' => $this->getParam(self::PARAM_ORDER_ID)
        ]);
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Presentation\Controllers\User.php

```
<?php

namespace Presentation\Controllers;

class User extends \Presentation\MVC\Controller {
    const PARAM_USER_NAME = 'un';
    const PARAM_PASSWORD = 'password';

    public function __construct(
        private \Application\SignInCommand $signInCommand,
        private \Application\SignOutCommand $signOutCommand,
        private \Application\SignedInUserQuery $signedInUserQuery
    ) {}
```

```
    public function GET_LogIn(): \Presentation\MVC\ActionResult {
        return $this->view('login', [
            'user' => $this->signedInUserQuery->execute(),
            'userName' => $this->tryGetParam(self::PARAM_USER_NAME, $value) ?
$value : ''
        ]);
    }

    public function POST_LogIn(): \Presentation\MVC\ActionResult {
        if (!$this->signInCommand->execute($this-
>getParam(self::PARAM_USER_NAME), $this->getParam(self::PARAM_PASSWORD))) {
            return $this->view('login', [
                'user' => $this->signedInUserQuery->execute(),
                'userName' => $this->getParam(self::PARAM_USER_NAME),
                'errors' => ['Invalid username or password']
            ]);
        } else {
            return $this->redirect('Home', 'Index');
        }
    }

    public function POST_LogOut(): \Presentation\MVC\ActionResult {
        $this->signOutCommand->execute();
        return $this->redirect('Home', 'Index');
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Presentation\MVC\ActionResult.php

```
<?php

namespace Presentation\MVC;

abstract class ActionResult
{
    public abstract function handle(MVC $mvc): void;
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Presentation\MVC\Controller.php

```
<?php

namespace Presentation\MVC;

abstract class Controller
{
    public final function getParam(string $id): mixed
    {
```

```
        return $_REQUEST[$id];
    }

    public final function tryGetParam(string $id, mixed &$value): bool
    {
        if (isset($_REQUEST[$id])) {
            $value = $_REQUEST[$id];
            return true;
         }
         return false;
    }

    public final function getRequestUri(): string {
        return $_SERVER['REQUEST_URI'];
    }

    public final function view(string $view, array $data = []): ViewResult
    {
        return new ViewResult($view, $data);
    }

    public final function redirectToUri(string $uri): RedirectToUriResult
    {
        return new RedirectToUriResult($uri);
    }

    public final function redirect(string $controller, string $action, array
$params = []): RedirectResult
    {
        return new RedirectResult($controller, $action, $params);
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Presentation\MVC\MVC.php

```
<?php

namespace Presentation\MVC;

use Exception;

final class MVC
{
    public function __construct(
        private string $viewPath = 'views/',
        private string $controllerNamespace = 'Presentation\\Controllers',
        private string $defaultController = 'Home',
        private string $defaultAction = 'Index',
        private string $controllerParameterName = 'c',
        private string $actionParameterName = 'a',
```

```
    ) {
    }

    public function getViewPath(): string
    {
        return $this->viewPath;
    }

    public function getControllerParameterName(): string
    {
        return $this->controllerParameterName;
    }

    public function getActionParameterName(): string
    {
        return $this->actionParameterName;
    }

    public function buildActionLink(?string $controller, ?string $action,
array $params): string
    {
        $res = '?' . rawurlencode($this->controllerParameterName)
            . '=' . rawurlencode($controller ?? $this->defaultController)
            . '&' . rawurlencode($this->actionParameterName)
            . '=' . rawurlencode($action ?? $this->defaultAction);
        foreach ($params as $name => $value) {
            $res .= '&' . rawurlencode($name) . '=' . rawurlencode($value);
        }
        return $res;
    }

    public function handleRequest(\ServiceProvider $serviceProvider): void
    {
        // determine controller class
        $controllerName = $_REQUEST[$this->controllerParameterName] ?? $this-
>defaultController;
        $controller = $this->controllerNamespace . "\\$controllerName";
        // determine HTTP method and action
        $method = $_SERVER['REQUEST_METHOD'];
        $action = $_REQUEST[$this->actionParameterName] ?? $this-
>defaultAction;
        // instanciate controller and call according action method
        $m = $method . '_' . $action;
        $res = $serviceProvider->resolve($controller)->$m();
        if (!is_a($res, ActionResult::class)) {
            throw new Exception("Return value of controller action
'$controllerName:$m' is not an instance of ActionResult.");
        }
        // handle result
        $res->handle($this);
```

```
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Presentation\MVC\RedirectResult.php

```php
<?php

namespace Presentation\MVC;

final class RedirectResult extends ActionResult
{
    public function __construct(
        private string $controller,
        private string $action,
        private array $params,
    ) {
    }

    public function handle(MVC $mvc): void
    {
        $location = $mvc->buildActionLink($this->controller, $this->action, $this->params);
        header("Location: $location");
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Presentation\MVC\RedirectToUriResult.php

```php
<?php

namespace Presentation\MVC;

final class RedirectToUriResult extends ActionResult
{
    public function __construct(
        private string $uri
    ) {
    }

    public function handle(MVC $mvc): void
    {
        header("Location: $this->uri");
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Presentation\MVC\ViewRenderer.php

```php
<?php

namespace Presentation\MVC;

final class ViewRenderer
{
    private function __construct()
    {
    }

    public static function render(MVC $mvc, string $view, mixed $data): void
    {
        // define helper functions for view rendering
        $render = function (string $view, mixed $data) use ($mvc) {
            self::render($mvc, $view, $data);
        };
        $htmlOut = function (mixed $value) {
            echo nl2br(htmlentities($value));
        };
        $beginForm = function (string $controller, string $action, array
$params = [], string $method = 'get', ?string $cssClass = null) use ($mvc) {
            $cc = $cssClass !== null ? " class=\"$cssClass\"" : '';
            echo "<form method=\"$method\" action=\"?\"$cc>";
            foreach ($params as $name => $value) {
                echo ("<input type=\"hidden\" name=\"$name\"
value=\"$value\">");
            }
            echo "<input type=\"hidden\" name=\"{$mvc-
>getControllerParameterName()}\" value=\"$controller\">";
            echo "<input type=\"hidden\" name=\"{$mvc-
>getActionParameterName()}\" value=\"$action\">";
        };
        $endForm = function () {
            echo '</form>';
        };
        $link = function (string $content, string $controller, string $action,
array $params = [], ?string $cssClass = null) use ($mvc, $htmlOut) {
            $cc = $cssClass != null ? " class=\"$cssClass\"" : '';
            $url = $mvc->buildActionLink($controller, $action, $params);
            echo "<a href=\"$url\"$cc>";
            $htmlOut($content);
            echo '</a>';
        };

        // render view
        require($mvc->getViewPath() . "$view.inc");
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\src\Presentation\MVC\ViewResult.php

```php
<?php

namespace Presentation\MVC;

final class ViewResult extends ActionResult
{
    public function __construct(
        private string $view,
        private array $data
    ) {
    }

    public function handle(MVC $mvc): void
    {
        ViewRenderer::render($mvc, $this->view, $this->data);
    }
}
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\views\bookList.inc

```php
<?php $render('partial/header', $data); ?>

        <h1>Books by category</h1>

        <nav class="nav nav-pills my-3">
            <?php foreach($data['categories'] as $cat) { ?>
                <?php $link($cat->name, 'Books', 'Index', ['cid' => $cat->id], 'nav-link' . ($cat->id == $data['selectedCategoryId'] ? ' active': '')); ?>
            <?php } ?>
        </nav>
<?php if($data['selectedCategoryId'] !== null): ?>
    <?php if(sizeof($data['books']) > 0) { ?>
        <?php $render('partial/books', array(
            'books' => $data['books'],
            'context' => $data['context']
        )); ?>
    <?php } else { ?>
        <p>No books in this category.</p>
    <?php } ?>
<?php else: ?>
    <p>Please select a category.</p>
<?php endif; ?>
<?php $render('partial/footer', $data); ?>
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\views\bookSearch.inc

```php
<?php $render ('partial/header', $data); ?>

    <h1>Search Site</h1>
```

```
    <div class="my-3">
        <?php $beginForm('Books', 'Search'); ?>
            <div class="row g-3">
                <div class="col-auto">
                    <input class="form-control" name="f" value="<?php
$htmlOut($data['filter'])?>" />
                </div>
                <div class="col-auto">
                    <button class="btn btn-primary">Search</button>
                </div>
            </div>
        <?php $endForm(); ?>
    </div>

    <?php if ($data['books'] != null): ?>
        <?php if (sizeof($data['books']) > 0) {
            $render('partial/books', array(
                'books' => $data['books'],
                'context' => $data['context']
            ));
        } else { ?>
            <p>No matching books found</p>
        <?php } ?>
    <?php endif; ?>

<?php $render ('partial/footer', $data); ?>
```

```
C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\views\home.inc
<?php $render('partial/header', $data); ?>

        <h1>Welcome</h1>
        <p>Welcome to the SCR4 book shop.</p>

<?php $render('partial/footer', $data); ?>

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\views\login.inc
<?php $render('partial/header', $data); ?>

    <h1>Login</h1>

    <?php $beginForm('User', 'LogIn', method: 'post'); ?>
```

```
        <div class="mb-3">
            <label for="uesrName" class="form-label">User name</label>
            <input class="form-control" id="username" name="un" value="<?php
$htmlOut($data['userName']); ?>"/>
        </div>
        <div class="mb-3">
            <label for="password" class="form-label">Password</label>
            <input class="form-control" id="password" name="password" />
        </div>
        <button class="btn btn-primary">Login</button>
    <?php $endForm(); ?>


<?php $render('partial/footer', $data); ?>


C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\views\orderForm.inc
<?php $render('partial/header', $data); ?>

    <h1>Checkout</h1>

    <p>You have <strong><?php $htmlOut($data['cartSize']) ?></strong> item(s)
in your cart.</p>

    <p>Please provide your credit card details for payment</p>

    <?php $beginForm('Order', 'Create', method: 'post') ?>
        <div class="mb-3">
            <label class="form-label" for="nameOnCard">Name on card</label>
            <input class="form-label" type="text" id="nameOnCard" name="noc"
/>
        </div>

        <div class="mb-3">
            <label class="form-label" for="cardNumber">Card number</label>
            <input class="form-label" type="text" id="cardNumber" name="cn" />
        </div>

        <button class="btn btn-primary" type="submit">Place order</button>
    <?php $endForm() ?>

<?php $render('partial/footer', $data); ?>

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\views\orderFormEmptyCart.inc
<?php $render('partial/header', $data); ?>

    <h1>Checkout</h1>

    <p>You have no item(s) in your cart. Please add som items in your cart</p>
```

```
<?php $render('partial/footer', $data); ?>

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\views\orderSummary.inc
<?php $render('partial/header', $data); ?>

    <h1>Order Summary</h1>
    <p>Thank you for your purchase</p>
    <p>Your order number is <?php $htmlOut($data['orderId']); ?>.</p>



<?php $render('partial/footer', $data); ?>

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\views\partial\books.inc
<table class="table align-middle">
    <thead>
        <tr>
            <th>Title</th>
            <th>Author</th>
            <th>Price</th>
            <th colspan="2">In cart</th>
        </tr>
    </thead>
    <tbody>
        <?php foreach ($data['books'] as $book) : ?>
            <tr <?php if($book->isInCart) { ?> class="table-active" <?php }
?>>
                <td><?php $htmlOut($book->title); ?></td>
                <td><?php $htmlOut($book->author); ?></td>
                <td><?php $htmlOut($book->price); ?></td>
                <td><?php $htmlOut($book->cartCount) ?></td>
                <td>
                    <div class="d-flex">
                        <?php $beginForm('Cart', 'Add', ['bid' => $book->id,
'ctx' => $data['context']], 'post', 'form-inline'); ?>
                            <button class="btn btn-link">Add</button>
                        <?php $endForm(); ?>

                        <?php $beginForm('Cart', 'Remove', ['bid' => $book-
>id, 'ctx' => $data['context']], 'post', 'form-inline'); ?>
                        <button class="btn btn-link" <?php if(!$book-
>isInCart){?> disabled <?php } ?>>Remove</button>
                        <?php $endForm(); ?>
                    </div>
                </td>
            </tr>
        <?php endforeach; ?>
    </tbody>
</table>
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\views\partial\errors.inc

```
<div class="alert alert-danger">
    <p>
        Please correct the following and try again:
    </p>
    <ul>
        <?php foreach ($data as $errMsg): ?>
            <li><?php $htmlOut($errMsg); ?></li>
        <?php endforeach; ?>
    </ul>
</div>
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\views\partial\footer.inc

```
    </div>
    <footer class="container">
        <p class="text-muted p-0"><?php echo htmlentities(date('c')); ?></p>
    </footer>
    <script src="js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\views\partial\header.inc

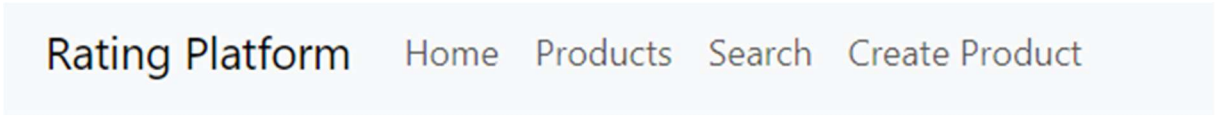
```
<!doctype html>
<html lang="en" data-bs-theme="light">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link href="css/bootstrap.min.css" rel="stylesheet">
    <title>SCR4 Book Shop</title>
</head>
<body>
    <nav class="navbar navbar-expand-lg bg-body-tertiary mb-3">
        <div class="container">
            <?php $link('SCR4 Book Shop', 'Home', 'Index', cssClass: 'navbar-
brand'); ?>
            <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbar">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbar">
                <nav class="navbar-nav me-auto">
                    <?php $link('Home', 'Home', 'Index', cssClass: 'nav-
link'); ?>
                    <?php $link('Books', 'Books', 'Index', cssClass: 'nav-
link'); ?>
                    <?php $link('Search', 'Books', 'Search', cssClass: 'nav-
link'); ?>
                    <?php $link('Checkout', 'Order', 'Create', cssClass: 'nav-
link'); ?>
```

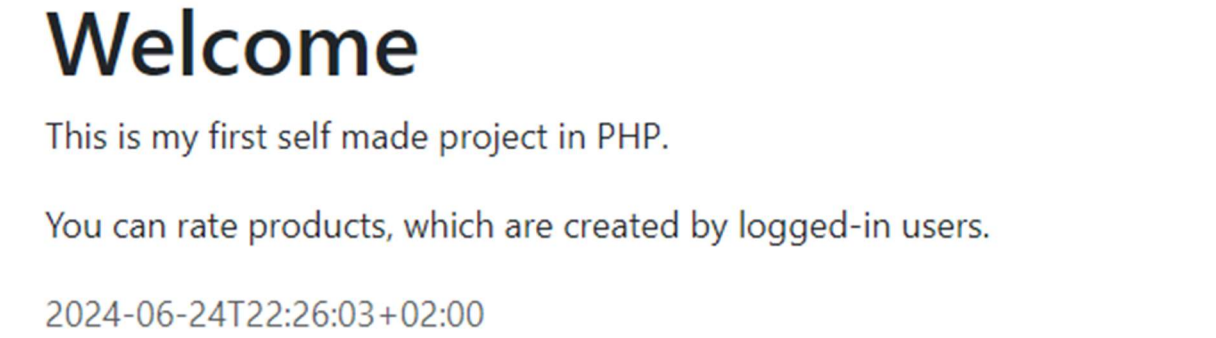
```
                </nav>
                <?php $render('partial/user', $data['user']); // TODO ?>
                <!-- TODO: navigation user -->
            </div>
        </div>
    </nav>
    <div class="container mb-3">

        <?php
            if(isset($data['errors'])) {
                $render('partial/errors', $data['errors']);
            }
        ?>
```

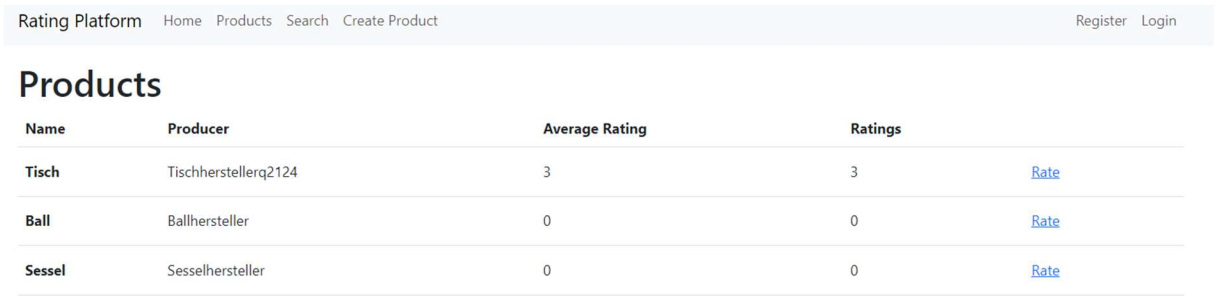
C:\xampp\htdocs\ratingportal\SCR4_Bookshop-main\views\partial\user.inc

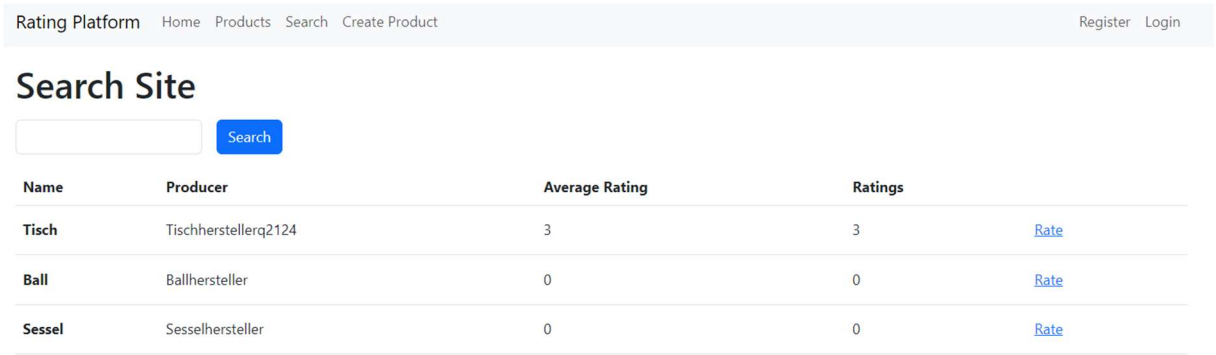
```
<?php if(!isset($data)): ?>
    <nav class="navbar-nav">
        <?php $link('Login', 'User', 'LogIn', cssClass: 'nav-link') ?>
    </nav>
<?php else: ?>
    <?php $beginForm('User', 'LogOut', method: 'post', cssClass: 'form-
inline'); ?>
        <span class="navbar-text me-2">Welcome, <strong><?php $htmlOut($data-
>userName); ?></strong></span>
        <button class="btn btn-secondary">Log out</button>
    <?php $endForm(); ?>
<?php endif; ?>
```

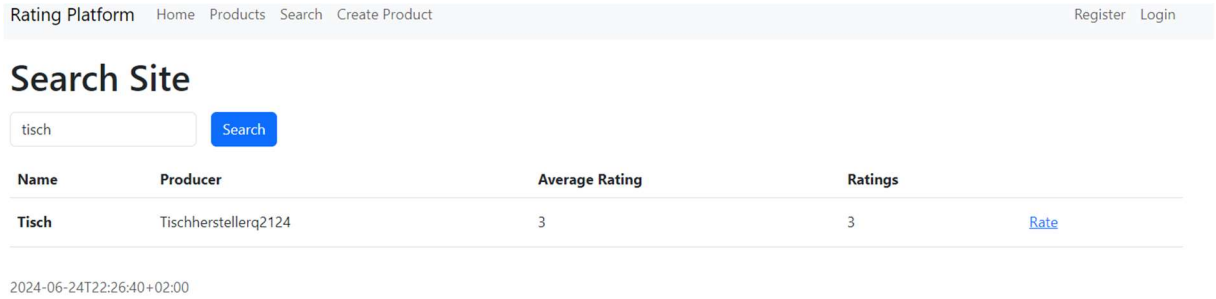
# 5. Testfälle

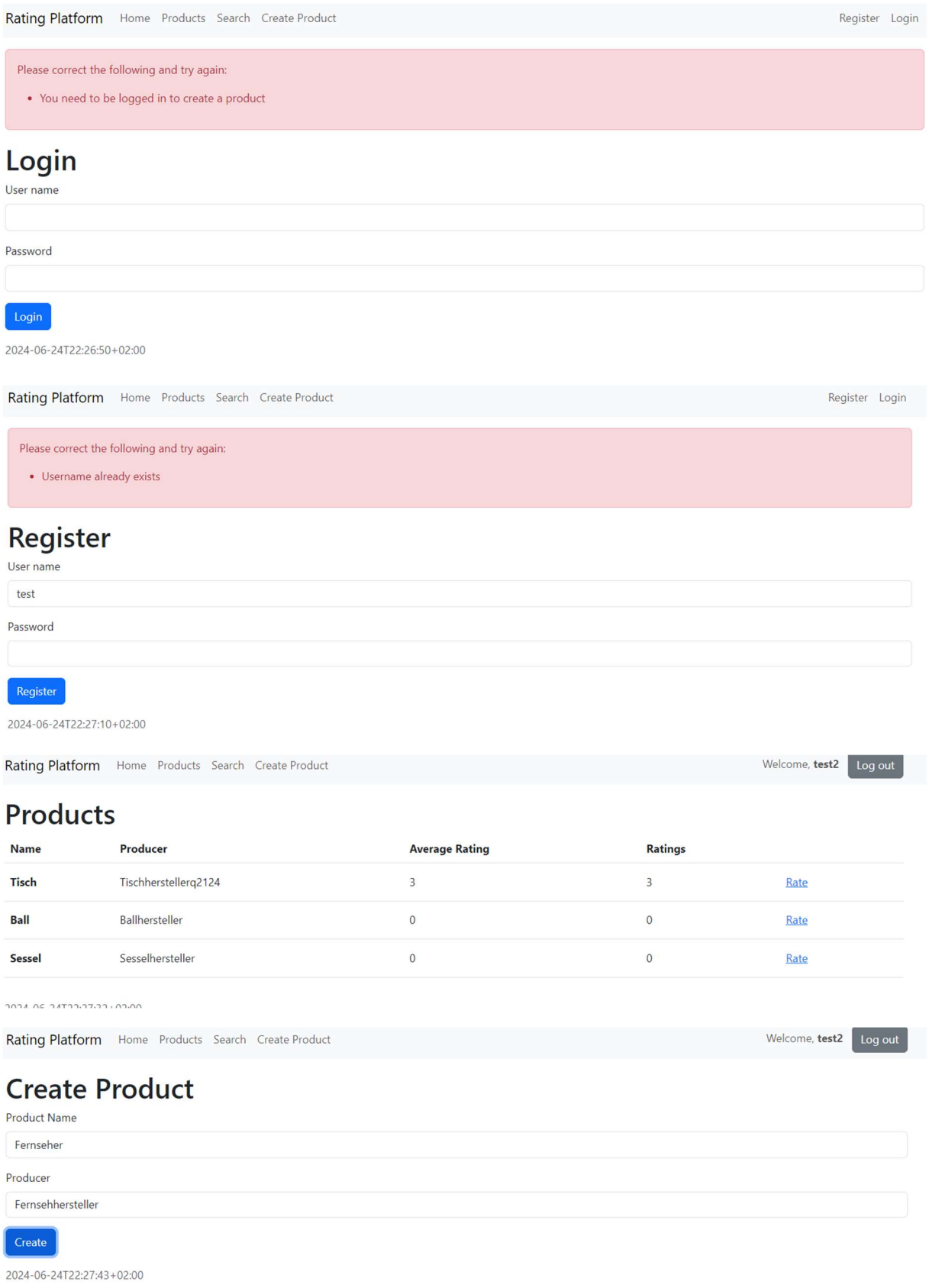

Rating Platform
Home
Products
Search
Create Product


Welcome
This is my first self made project in PHP.
You can rate products, which are created by logged-in users.
2024-06-24T22:26:03+02:00


Rating Platform
Home
Products
Search
Create Product
Register
Login
Products
Name
Producer
Average Rating
Ratings
Tisch
Tischherstellerq2124
3
3
Rate
Ball
Ballhersteller
0
0
Rate
Sessel
Sesselhersteller
0
0
Rate


Rating Platform
Home
Products
Search
Create Product
Register
Login
Search Site
Search
Name
Producer
Average Rating
Ratings
Tisch
Tischherstellerq2124
3
3
Rate
Ball
Ballhersteller
0
0
Rate
Sessel
Sesselhersteller
0
0
Rate


Rating Platform
Home
Products
Search
Create Product
Register
Login
Search Site
tisch
Search
Name
Producer
Average Rating
Ratings
Tisch
Tischherstellerq2124
3
3
Rate
2024-06-24T22:26:40+02:00


Rating Platform
Home
Products
Search
Create Product
Register
Login
Please correct the following and try again:
You need to be logged in to create a product
Login
User name
Password
Login
2024-06-24T22:26:50+02:00
Rating Platform
Home
Products
Search
Create Product
Register
Login
Please correct the following and try again:
Username already exists
Register
User name
test
Password
Register
2024-06-24T22:27:10+02:00
Rating Platform
Home
Products
Search
Create Product
Welcome, test2
Log out
Products
Name
Producer
Average Rating
Ratings
Tisch
Tischherstellerq2124
3
3
Rate
Ball
Ballhersteller
0
0
Rate
Sessel
Sesselhersteller
0
0
Rate
Rating Platform
Home
Products
Search
Create Product
Welcome, test2
Log out
Create Product
Product Name
Fernseher
Producer
Fernsehhersteller
Create
2024-06-24T22:27:43+02:00

Rating Platform Home Products Search Create Product Welcome, **test2** Log out

# Products

| Name | Producer | Average Rating | Ratings | | |
|---|---|---|---|---|---|
| **Tisch** | Tischherstellerq2124 | 3 | 3 | Rate | |
| **Ball** | Ballhersteller | 0 | 0 | Rate | |
| **Sessel** | Sesselhersteller | 0 | 0 | Rate | |
| **Fernseher** | Fernsehhersteller | 0 | 0 | Rate | Edit |

2024-06-24T22:28:05+02:00

Rating Platform Home Products Search Create Product Welcome, **test2** Log out

Please correct the following and try again:

- Rating must be between 1 and 5

# Rate Ball:

Rating -1

Comment

Add

2024-06-24T22:28:21+02:00

Rating Platform Home Products Search Create Product Welcome, **test2** Log out

# Ratings for Ball

| Username | Rating | Date | Comment |
|---|---|---|---|
| test2 | 4 | 2024-06-24 | abc |

2024-06-24T22:28:37+02:00

Rating Platform Home Products Search Create Product Welcome, **test2** Log out

# Rate Ball:

Rating 4

Comment

abc

Update

Delete

2024-06-24T22:28:49+02:00

Rating Platform Home Products Search Create Product Welcome, **test2** Log out

# Edit Product

Product Name

Fernseher

Producer

Fernsehhersteller

Edit

2024-06-24T22:29:07+02:00

Rating Platform Home Products Search Create Product Welcome, **test2** Log out

# Products

| Name | Producer | Average Rating | Ratings | | |
|---|---|---|---|---|---|
| **Tisch** | Tischherstellerq2124 | 3 | 3 | Rate | |
| **Ball** | Ballhersteller | 4 | 1 | Rate | |
| **Sessel** | Sesselhersteller | 0 | 0 | Rate | |
| **Fernseher** | Fernsehherstell | 0 | 0 | Rate | Edit |

2024-06-24T22:29:17+02:00

Rating Platform Home Products Search Create Product Register Login

# Products

| Name | Producer | Average Rating | Ratings | |
|---|---|---|---|---|
| **Tisch** | Tischherstellerq2124 | 3 | 3 | Rate |
| **Ball** | Ballhersteller | 4 | 1 | Rate |
| **Sessel** | Sesselhersteller | 0 | 0 | Rate |
| **Fernseher** | Fernsehherstell | 0 | 0 | Rate |

2024-06-24T22:29:32+02:00