Правительство Санкт-Петербурга

Комитет по науке и высшей школе

Санкт-Петербургское государственное бюджетное

профессиональное образовательное учреждение

«Политехнический колледж городского хозяйства»

**КУРСОВОЙ ПРОЕКТ**

**по профессиональному модулю**

**«ПМ.09 — Проектирование, разработка и оптимизация веб-приложений»**

**(МДК 09.01 - Проектирование и разработка веб-приложений)**

Специальность\_\_09.02.07 Информационные системы и программирование

Разработка серверного приложения REST API

«Магазин носков»

**Студент** группы **ИР-20-4**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д. В. Косинов

**Руководитель**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Л.В. Ильюшенков

**Председатель П(Ц)К**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** Л.В. Левит

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Л.В. Ильюшенков

Санкт-Петербург

2023 г.

Правительство Санкт-Петербурга

Комитет по науке и высшей школе

Санкт-Петербургское государственное бюджетное

профессиональное образовательное учреждение

«Политехнический колледж городского хозяйства»

|  |  |
| --- | --- |
| УТВЕРЖДЕНПКГХ 09.02.07 ИP-20-4 .1389-20 - ЛУ |  |

­

# КУРСОВОЙ ПРОЕКТ

Разработка серверного приложения REST API «Магазин носков»

## Техническое задание

ПКГХ 09.02.07 ИР-20-4. 1389-20 ТЗ

Листов \_\_

1. **ОБЩИЕ СВЕДЕНИЯ**

Дата выдачи курсового задания «30» октября 2023 г

Руководитель курсового проекта: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Л.В. Ильюшенков

подпись

Наименовании организации-разработчика: СПб ГБПОУ «Политехнический колледж городского хозяйства»

Условное обозначение программы: ПКГХ 09.02.07 ИР-20-4 .1389-20.П2

Рассмотрено и одобрено предметной комиссией: протокол № 3 от «27» сентября 2023 г.

Председатель П(Ц)К 09.02.07 \_\_\_\_\_\_\_\_\_\_\_ Л.В. Левит

1. **ЦЕЛИ И НАЗНАЧЕНИЕ СОЗДАНИЯ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ**

Основная задача автоматизированной системы состоит в том, чтобы автоматизировать процесс модерации сайта по продаже носков.

Программный продукт будет полезен для трёх категорий пользователей:

* Незарегистрированные пользователи (просмотр доступных товаров)
* Зарегистрированные пользователи (просмотр доступных товаров, создание учётной записи, добавление в корзину, оформление заказа)
* Администраторы (просмотр, изменение, добавление и удаление товаров, просмотр зарегистрированных пользователей и информации о них)

1. **Характеристика объектов автоматизации**

Процесс покупки товаров включает в себя: просмотр и выбор товаров, добавление товаров в корзину, редактирование корзины, оформление заказа. Пока заказ не подтвержден продавцом, покупатель может редактировать заказ или удалить.

Процесс регистрации включает в себя: ввод необходимых данных пользователем, подтверждение электронной почты, авторизация пользователя в системе.

Процесс авторизации включает в себя: ввод пользователем персональных данных, аутентификация пользователя в системе.

1. **ТРЕБОВАНИЯ К АВТОМАТИЗИРОВАННОЙ СИСТЕМЕ**

В случае ошибок связанных с валидацией данных во всех запросах необходимо возвращать следующее тело ответа:

{

“error”: {

“code”: <code>,

“message”: <message>,

“errors”: {

<key>: [ <error message>]

}

}

}

**API для обычного пользователя**

**Регистрация**

Запрос для регистрации нового пользователя в системе. При отправке запроса необходимо передать объект со следующими свойствами:

* fio – обязательное поле, строка, допустимые символы - кириллица, пробел, дефис;
* phone – обязательное, строка, только цифры;
* email – обязательное поле, строка, формат – email, уникальный;
* password – обязательное поле, строка, не менее 7 символов, обязательно: 1 цифра, 1 строчная, 1 заглавная буквы;

Таблица 1 – Регистрация

|  |  |
| --- | --- |
| Request | Response |
| URL: {host}/api/register  Method: POST  Headers  - Content-Type: application/json  Body:  {  “fio”: “Борис Борисович”,  “phone”: “89917823453”,  “email”: “user@user.ru”,  “password”: “paSSword1”,  } | **-----------------------Successful---------------------**  **Successful**: Status: 204  **Body:**  {  "Successful": {  "Status": 204  }  }  **-------------------Validation error-----------------**  **Status**: 422  Content-Type: application/json  **Body**:  {  “error”: {  “code”: 422,  “message”: “Validation error”,  “errors”: {  <key>: <массив ошибок>  }  } |

**Аутентификация**

Запрос для аутентификации пользователя в системе. При отправке запроса необходимо передать объект с логином и паролем. Если клиент отправил корректные данные, то необходимо вернуть сгенерированный токен, а иначе сообщение об ошибке.

* email – обязательное поле, строка, формат – email;
* password – обязательное поле.

Таблица 2 – Аутентификация

|  |  |
| --- | --- |
| Request | Response |
| URL: {host}/api/auth  Method: POST  Headers  - Content-Type: application/json  Body:  {  “email”: “user@user.ru”  “password”: “paSSword1”,  } | **----------------------------Successful----------------------------**  **Status:** 200  Content-Type: application/json  **Body:**  {  “data”: {  “token”: <сгенерированный token>  }  }  **-------------------------Validation error-----------------------**  **Status:** 422  Content-Type: application/json  **Body:**  {  “error”: {  “code”: 422,  “message”: “Validation error”,  “errors”: {  <key>: <массив ошибок>  }  }  **--------------------------Unauthorized-------------------------**  **Status:** 401  Content-Type: application/json  **Body:**  {  “error”: {  “code”: 401,  “message”: “Unauthorized”,  “errors”: {  “error”: [ “phone or password incorrect”]  }  }  } |

**Информация о пользователе**

Запрос для получения информации о пользователе. При отправке запроса необходимо передать полученный при авторизации токен в Headers:

Таблица 3 – Информация о пользователе

|  |  |
| --- | --- |
| Request | Response |
| URL: {host}/api/user  Method: GET  Headers  - Content-Type: application/json  - Authorization: Bearer {token}  Body:  {  } | **----------------------------Successful----------------------------**  **Status:** 200  Content-Type: application/json  Body:  {  “data”: {  “user”: [  {  “id”: 1,  “fio”: “Борис Борисович”,  “phone”: “89917823453”,  “email”: “user@user.ru”  }  ]  }  }  **--------------------------Unauthorized-------------------------**  Status: 401  Content-Type: application/json  Body:  {  "error": {  "code": 401,  "message": "Unauthorized"  }  } |

**Изменение адреса электронной почты**

Запрос для изменения электронной почты пользователя. При отправке запроса необходимо передать объект со следующими свойствами:

* email – обязательное поле, строка, формат – email;

Таблица 4 – Изменение адреса электронной почты

|  |  |
| --- | --- |
| Request | Response |
| URL: {host}/api/email  Method: PATCH  Headers  - Content-Type: application/json  - Authorization: Bearer {token}  Body:  {  “email”: “newmail@user.ru”,  } | **--------------------------Successful--------------------------**  **Status**: 200  **Content-Type:** application/json  {  “data”: {  “status”: “email was changed successfully”  }  **------------------------Unauthorized------------------------**  **Status**: 401  **Content-Type:** application/json  **Body**:  {  "error": {  "code": 401,  "message": "Unauthorized"  }  }  **----------------------Validation error----------------------**  **Status**: 422  **Content-Type:** application/json  **Body**:  {  "error": {  "code": 422,  "message": "Validation error",  "error":  "The email should not be empty."  }  } |

**Просмотр всех товаров**

Запрос на получение списка всех товаров

Таблица 5 – Просмотр всех товаров

|  |  |
| --- | --- |
| Request | Response |
| URL: {host}/api/tovars  Method: GET  Headers  - Content-Type: application/json  Body:  {  } | **--------------------------Successful--------------------------**  **Status:** 200  **Content-Type:** application/json  **Body:**  {  “data”: {  [  {  "id": 10,  "name": "SinSay Polar socks",  "description": "Хлопковые мужские носки”,  "price": "4.99",  "stock": "50",  "image\_url": "image\_storage/29.jpg",  },  {  "id": 10,  "name": "SinSay Tiger socks",  "description": "Носки с рисунком тигра”,  "price": "7.00",  "stock": "90",  "image\_url": "image\_storage/9.jpg",  },…  ]  }  } |

**Просмотр одного товара**

Запрос на просмотр информации о конкретном товаре. При отправке запроса необходимо передать в URL объект со следующими свойствами:

* id – обязательное поле, число.

Таблица 6 – Просмотр одного товара

|  |  |
| --- | --- |
| Request | Response |
| URL: {host}/api/tovars/:id  Method: GET  Headers  - Content-Type: application/json  Body:  {  } | **--------------------------Successful--------------------------**  **Status**: 200  **Content-Type**: application/json  **Body**:  {  “data”: {  {  "id": 1,  "name": "Bershka Griffindor Socks",  "price": "9.00",  "description": "Всесезонные эластичные мужские носки",  "stock": "44",  "image\_url": "image\_storage/2.jpg"  }  }  }  **-----------------------------Error-----------------------------**  **Status**: 404  **Content-Type**: application/json  **Body**:  {  "error": {  "code": 404,  "message": "Product not found",  "error":  “Product not found”  }  } |

**Просмотр корзины**

Запрос при открытии пользователем корзины.

Таблица 7 – Просмотр корзины

|  |  |
| --- | --- |
| Request | Response |
| URL: {host}/api/show\_cart  Method: GET  Headers  - Content-Type: application/json  - Authorization: Bearer {token}  Body:  {  } | **--------------------------Successful--------------------------**  **Status**: 200  **Content-Type:** application/json  **Body**:  {  “data”: {  [  {  "id": 18,  "itemId": 8,  "itemName": " Bershka Griffindor Socks ",  "amount": 1,  }  ]  }  **------------------------Unauthorized------------------------**  **Status**: 401  **Content-Type**: application/json  **Body**:  {  "error": {  "code": 401,  "message": "Unauthorized"  }  } |

**Добавление товара в корзину**

Запрос для добавления товара в корзину пользователя. При отправке запроса необходимо передать объект со следующими свойствами:

* tovarId – обязательное поле, число;
* amount – обязательное поле, число;

Таблица 8 – Добавление товара в корзину

|  |  |
| --- | --- |
| Request | Response |
| URL: {host}/api/add\_cart  Method: POST  Headers  - Content-Type: application/json  - Authorization: Bearer {token}  Body:  {  "itemId": 10,  "amount": 3  } | **------------------------Successful----------------------------**  **Status**: 200  **Content-Type**: application/json  **Body:**  {  “data”: {  {  "id": 22,  }  }  **-----------------------Unauthorized-------------------------**  **Status**: 401  **Content-Type**: application/json  **Body**:  {  "error": {  "code": 401,  "message": "Unauthorized"  }  }  **-----------------------------Error-----------------------------**  **Status**: 404  **Content-Type**: application/json  **Body**:  {  "error": {  "code": 404,  "message": "Product not found",  }  }  **-----------------------------Error-----------------------------**  **Status**: 401  **Content-Type**: application/json  **Body**:  {  "error": {  "code": 401,  "message": "Amount not enough",  }  }  **--------------------Out of stock error----------------------**  **Status**: 404  **Content-Type**: application/json  **Body**:  {  "error": {  "code": 404,  "message": "Out of stock",  }  }  **---------------Repeated item in cart error---------------**  **Status**: 400  **Content-Type**: application/json  **Body**:  {  "error": {  "code": 400,  "message": "Item already exists in the cart",  }  } |

**Удаление товара**

Запрос для удаления товара из корзины. При отправке запроса необходимо передать в URL объект со следующими свойствами:

* id – обязательное поле, число;

Таблица 9 – Удаление товара

|  |  |
| --- | --- |
| Request | Response |
| URL: {host}/api/del\_cart/:id  Method: DELETE  Headers  - Content-Type: application/json  - Authorization: Bearer {token}  Body:  {  } | **------------------------Successful----------------------------**  **Status**: 200  **Content-Type**: application/json  **Body**:  {  "code": 200,  "message": "Successful"  }  **-----------------------Unauthorized-------------------------**  **Status**: 401  **Content-Type**: application/json  **Body**:  {  "error": {  "code": 401,  "message": "Unauthorized"  }  }  **-----------------------------Error-----------------------------**  **Status**: 404  **Content-Type**: application/json  **Body**:  {  "error": {  "code": 404,  "message": "Product not found",  }  } |

**Изменение количества товара в корзине**

Запрос для изменения количества товара в корзине пользователя. При отправке запроса необходимо передать объект со следующими свойствами:

* amount – обязательное поле, число;
* id – обязательное поле, число

Таблица 10 – Изменение количества товара в корзине

|  |  |
| --- | --- |
| Request | Response |
| URL: {host}/api/amount\_cart/:id  Method: PATCH  Headers  - Content-Type: application/json  - Authorization: Bearer {token}  Body:  {  "amount": 2  } | **------------------------Successful----------------------------**  **Status**: 200  **Content-Type**: application/json  **Body**:  {  "message": "Amount changed successfully"  }  **-----------------------Unauthorized------------------------**  **Status**: 401  **Content-Type**: application/json  **Body**:  {  "error": {  "code": 401,  "message": "Unauthorized"  }  }  **-----------------------------Error-----------------------------**  **Status**: 404  **Content-Type**: application/json  **Body**:  {  "error": {  "code": 404,  "message": "Product not found",  }  }  **--------------------Out of stock error----------------------**  **Status**: 404  **Content-Type**: application/json  **Body**:  {  "error": {  "code": 404,  "message": "Out of stock",  }  } |

**Оформление заказа**

Запрос для создания заказа пользователя. При отправке запроса необходимо передать объект со следующими свойствами:

* id – обязательное поле, число;
* total\_price – обязательное поле, строка;
* tovars – обязательное поле, массив продуктов:
* amount – обязательное поле, число.

Таблица 11 – Оформление заказа

|  |  |
| --- | --- |
| Request | Response |
| URL: {host}/api/order  Method: POST  Headers  - Content-Type: application/json  - Authorization: Bearer {token}  Body:  {  “tovars”:[  {  “id”: 1,  “amount”: 3  },  {  “id”: 2,  “amount”: 2  }  ],  “address”: “ул. Коллонтай 23”  } | **------------------------Successful----------------------------**  **Status**: 200  **Content-Type**: application/json  **Body**:  {  “message”: “Order created successfully”  }  **-----------------------Unauthorized-------------------------**  **Status**: 401  **Content-Type**: application/json  **Body**:  {  “error”: {  “code”: 401,  “message”: “Unauthorized”  }  }  **---------------------Empty cart Error --------------------**  **Status**: 400  **Content-Type**: application/json  **Body**:  {  “error”: {  “code”: 400,  “message”: “Bad Request: Cart is empty”  }  }  **------------------Empty address Error -------------------**  **Status**: 400  **Content-Type**: application/json  **Body**:  {  “error”: {  “code”: 400,  “message”: “Bad Request: “address” parameter is empty”  }  } |

**Получение заказов определённого пользователя**

Запрос для просмотра в личном кабинете пользователя всех сделанных заказов.

Таблица 12 – Получение заказов определённого пользователя

|  |  |
| --- | --- |
| Request | Response |
| URL: {host}/api/orders  Method: GET  Headers  - Content-Type: application/json  - Authorization: Bearer {token}  Body:  {  } | **------------------------Successful----------------------------**  **Status**: 200  **Content-Type**: application/json  **Body**:  {  “data”: {  [  {  “order\_id”: 99,  “status”: “Завершён”,  “total\_cost”: “18.00”,  “order\_date”: “2023-03-01 16:32:40”,  “address”: “ул. Будапештская, дом 9, к2, кв. 8»,  «tovars»: {  “name”: “St. Friday socks”,  “price”: “9”  “amount”: “2”,  },  },  {  “order\_id”: 101,  “status”: “В обработке”,  “total\_cost”: “28.00”,  “order\_date”: “2023-03-01 17:12:09”,  “address”: “ул. Будапештская, дом 9, к2, кв. 8»,  «tovars»: {  [  {  “name”: “St. Friday socks”,  “price”: “9.00”  “amount”: “2”,  },  {  “name”: “Party Socks”,  “price”: “10.00”  “amount”: “1”,  },  ],  },  ],  }  **-----------------------Unauthorized-------------------------**  **Status**: 401  **Content-Type**: application/json  **Body**:  {  “error”: {  “code”: 401,  “message”: “Unauthorized”  }  }  **-----------------------------Error-----------------------------**  **Status**: 404  **Content-Type**: application/json  **Body**:  {  “error”: {  “code”: 404,  “message”: “Orders not found”,  } } |

**Получение определённого заказа пользователя:**

Запрос для просмотра конкретного заказа. При отправке запроса необходимо передать объект со следующими свойствами:

* id – обязательное поле, число;

Таблица 13 – Получение определённого заказа

|  |  |
| --- | --- |
| Request | Response |
| URL: {host}/api/order/:id  Method: GET  Headers  - Content-Type: application/json  - Authorization: Bearer {token}  Body:  {  } | **------------------------Successful----------------------------**  **Status**: 200  **Content-Type**: application/json  **Body**:  {  “data”: {  {  “order\_id”: 101,  “status”: “В обработке”,  “total\_cost”: “28.00”,  “order\_date”: “2023-03-01 17:12:09”,  «address»: «ул. Будапештская, дом 9, к2, кв. 8»,  “tovars”: {  [  {  “name”: “St. Friday socks”,  “price”: “9.00”  “amount”: “2”,  },  {  “name”: “Party Socks”,  “price”: “10.00”  “amount”: “1”,  },  ],  },  }  **-----------------------Unauthorized-------------------------**  **Status**: 401  **Content-Type**: application/json  **Body**:  {  “error”: {  “code”: 401,  “message”: “Unauthorized”  }  }  **-----------------------------Error-----------------------------**  **Status**: 404  **Content-Type**: application/json  **Body**:  {  “error”: {  “code”: 404,  “message”: “Order not found”  }  } |

**API для администратора**

**Добавление нового товара**

Запрос для добавления нового товара. При отправке запроса необходимо передать объект со следующими свойствами:

* name – обязательное поле, строка;
* description – обязательное поле, строка;
* price – обязательное поле, число;
* stock – обязательное поле, число;
* image – обязательное поле, изображение, формат: jpg или bmd, не более 2 мб

Таблица 14 – Добавление нового товара

|  |  |
| --- | --- |
| Request | Response |
| URL: {host}/api/tovars  Method: POST  Headers  - Content-Type: multipart/form-data  - Authorization: Bearer {token}  Body:  {  “name”: “SinSay Polar socks”,  “description”: “Хлопковые мужские носки”,  “price”: “4.99”,  “stock”: “50”,  “image”: “29.jpg”,  } | **--------------------------Successful--------------------------**  **Status**: 200  **Content-Type:** application/json  {  “data”: {  “status”: “New item added successfully”  }  **----------------------Validation error----------------------**  **Status**: 422  **Content-Type:** application/json  **Body**:  {  “error”: {  “code”: 422,  “message”: “Validation error”,  “error”:  “Validation is invalid”  }  }  **-----------------------Unauthorized-------------------------**  **Status**: 401  **Content-Type**: application/json  **Body**:  {  “error”: {  “code”: 401,  “message”: “Unauthorized”  }  } |

**Удаление товара**

Запрос для удаления товара. При отправке запроса необходимо передать в URL объект со следующими свойствами:

* id – обязательное поле, число;

Таблица 15 – Удаление товара

|  |  |
| --- | --- |
| Request | Response |
| URL: {host}/api/tovars/:id  Method: DELETE  Headers  - Content-Type: application/json  - Authorization: Bearer {token}  Body:  {  } | **--------------------------Successful--------------------------**  **Status**: 200  **Content-Type:** application/json  {  “data”: {  “status”: “Item deleted successfully”  }  **-----------------------Unauthorized-------------------------**  **Status**: 401  **Content-Type**: application/json  **Body**:  {  “error”: {  “code”: 401,  “message”: “Unauthorized”  }  }  **-----------------------------Error-----------------------------**  **Status**: 404  **Content-Type**: application/json  **Body**:  {  “error”: {  “code”: 404,  “message”: “Item not found”,  “error”:  “Item not found”  }  } |

**Изменение товара**

Запрос для изменения какой-либо информации о товаре (Кроме изображения). При отправке запроса необходимо передать объект со следующими свойствами:

* param – название поля которое нужно изменить.

Таблица 16 – Изменение товара

|  |  |
| --- | --- |
| Request | Response |
| URL: {host}/api/tovars/{param}  Method: PATCH  Headers  - Content-Type: application/json  - Authorization: Bearer {token}  Body:  {  “{param}”: “значение”  } | **--------------------------Successful--------------------------**  **Status**: 200  **Content-Type:** application/json  {  “data”: {  “status”: “Item changed successfully”  }  **-----------------------Unauthorized-------------------------**  **Status**: 401  **Content-Type**: application/json  **Body**:  {  “error”: {  “code”: 401,  “message”: “Unauthorized”  }  }  **-----------------------------Error-----------------------------**  **Status**: 404  **Content-Type**: application/json  **Body**:  {  “error”: {  “code”: 404,  “message”: “{param} not found”,  “error”:  “{param} not found”  }  } |

1. **СОСТАВ И СОДЕРЖАНИЕ РАБОТ ПО СОЗДАНИЮ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ**
2. Проектирование базы данных
3. Развертывание Yii2 на сервере
4. Реализация REST API
5. Тестирование REST API
6. **ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ**

Контроль качества и приемка программного средства будет производиться при помощи программы Postman. Требуется разработать запросы к API. Проверке подлежат удачные ответы, ошибки валидации, авторизации и пр. На каждую автоматизированную функцию не менее трех запросов. Проект необходимо разместить на github.

1. **ТРЕБОВАНИЯ К СОСТАВУ И СОДЕРЖАНИЮ РАБОТ ПО ПОДГОТОВКЕ ОБЬЕКТА АВТОМАТИЗАЦИИ К ВВОДУ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ В ДЕЙТСВИЕ**

Автоматизированная система будет доступна по адресу: [k-kosinov.xn--80ahdri7a.site](http://k-kosinov.сделай.site)

Чтобы запустить автоматизированную систему нужно:

* Развернуть базу данных, создать таблица и связи, или загрузить бэкап;
* Загрузить файлы на сервер;
* Настроить перенаправления путей. Настройки подключения к базе данных.

1. **ТРЕБОВАНИЯ К ДОКУМЕНТРИВАНИЮ**
2. Разработка пояснительной записки, ГОСТ 34.201 — 2020
3. Разработка руководства администратора ГОСТ Р ИСО/МЭК 25051—2017
4. Источники, используемые в разработке