**Поволжский Государственный Университет Телекоммуникаций и Информатики**Кафедра «Программная инженерия»

|  |  |
| --- | --- |
| Сдана на проверку  «\_\_» \_\_\_\_2023 г. | Допустить к защите  «\_\_» \_\_\_\_\_ 2023 г.  Защищена с оценкой  «\_\_\_» \_\_\_\_\_\_\_\_ 2023 г. |

**КУРСОВАЯ РАБОТА**По дисциплине: «Прикладное программирование»  
На тему: «Разработка клиент-серверного web-приложения библиотека»  
  
Пояснительная записка

|  |  |
| --- | --- |
| Студент группы ИВТ-11\_\_\_\_\_\_\_ Ермаков А. С.  (роспись) (ФИО)  Руководитель \_\_\_\_\_\_\_\_ к.т.н., доц. Ахметшина Э. Г.  (роспись) (ФИО) | 0 190550 0  (№ зачетной книжки) |

Самара 2023

**Рецензия**

**Содержание**

[Описание предметной области. Актуальность 4](#_Toc90735856)

[Описание программы. Общие сведения 5](#_Toc90735857)

[Функциональное назначение 5](#_Toc90735858)

[Логическая модель базы данных 6](#_Toc90735859)

[Физическая модель базы данных 9](#_Toc90735860)

[Диаграмма классов 9](#_Toc90735861)

[Диаграмма компонентов 11](#_Toc90735862)

[Диаграмма вариантов использования 13](#_Toc90735863)

[Диаграмма последовательности 14](#_Toc90735864)

[Демонстрация работы приложения 14](#_Toc90735865)

[Список используемых источников 19](#_Toc90735866)

[Приложение А- Листинг программного кода 20](#_Toc90735867)

# **Описание предметной области. Актуальность**

# **Описание программы. Общие сведения**

Приложение – веб-ресурс. Установка не требуется.

Необходимые требования: доступ к интернету.

Языки программирования: Java, JavaScript.

Фреймворки/Библиотеки: Spring, React.js

Среда разработки: Intellij IDEA.

Объем проекта: 1.5 МБ (вместе с исходным кодом).

Исходный код (классы и код форм): 1350 строк.

# 

# **Функциональное назначение**

Проект "Инвентаризация товаров" создан для эффективного учета товаров на складе и обеспечения пользователей возможностью управления данными о продукции. Основные задачи включают в себя добавление, редактирование и отслеживание товаров в инвентаре, а также контроль сроков годности. Приложение предназначено для облегчения процессов управления запасами и предоставления надежной системы учета товаров.

#### Описание используемых технологий

****React.js** — это библиотека JavaScript для разработки интерактивных пользовательских интерфейсов. Созданная Facebook, она позволяет эффективно строить многокомпонентные веб-приложения, где каждый компонент управляет своим состоянием. React также обеспечивает виртуальный DOM, что повышает производительность при обновлении данных.**

****Spring Boot** — фреймворк для разработки веб-приложений на языке программирования Java. Он предоставляет удобные средства для создания микросервисов и веб-приложений с минимальными усилиями разработчика. С Spring Boot легко создавать самостоятельные, автономные приложения, что ускоряет процесс разработки.**

****RESTful (Representational State Transfer)** — стиль архитектуры взаимодействия компонентов распределенного приложения. Он основан на принципах, описанных в ограниченном и однозначном интерфейсе, ресурсах, представлении ресурсов и управлении состоянием. RESTful API предоставляет простой и эффективный способ связи между клиентом и сервером.**

****PostgreSQL** — это мощная система управления реляционными базами данных с открытым исходным кодом. Она предоставляет надежное хранение данных и поддерживает множество расширений, что делает ее превосходным выбором для приложений, требующих структурированные данные. PostgreSQL обеспечивает высокую производительность и расширенные возможности запросов.**

**Этот технологический стек обеспечивает надежность, производительность и гибкость при разработке приложения "Инвентаризация товаров". Все компоненты взаимодействуют гармонично, обеспечивая пользовательскому опыту высший уровень удовлетворения.**

# **Логическая модель базы данных**

Описание сущностей БД ( с атрибутами) и указания связей между сущностями!

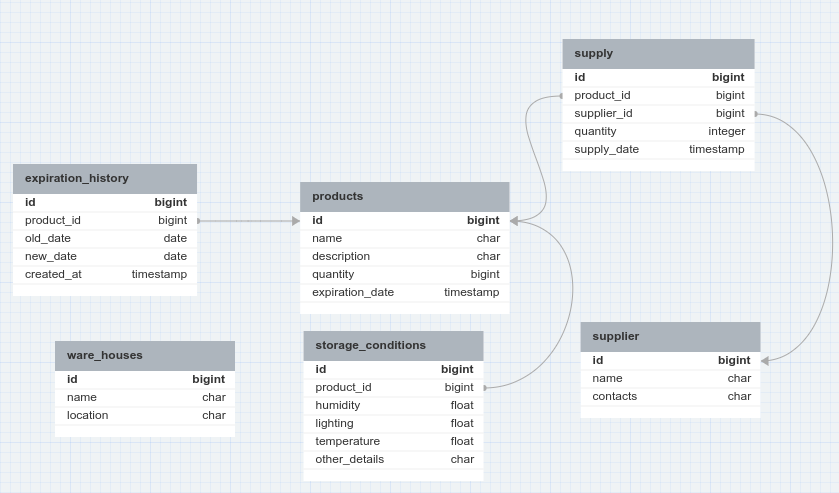


рис. - логическая модель базы данных

# **Физическая модель базы данных**

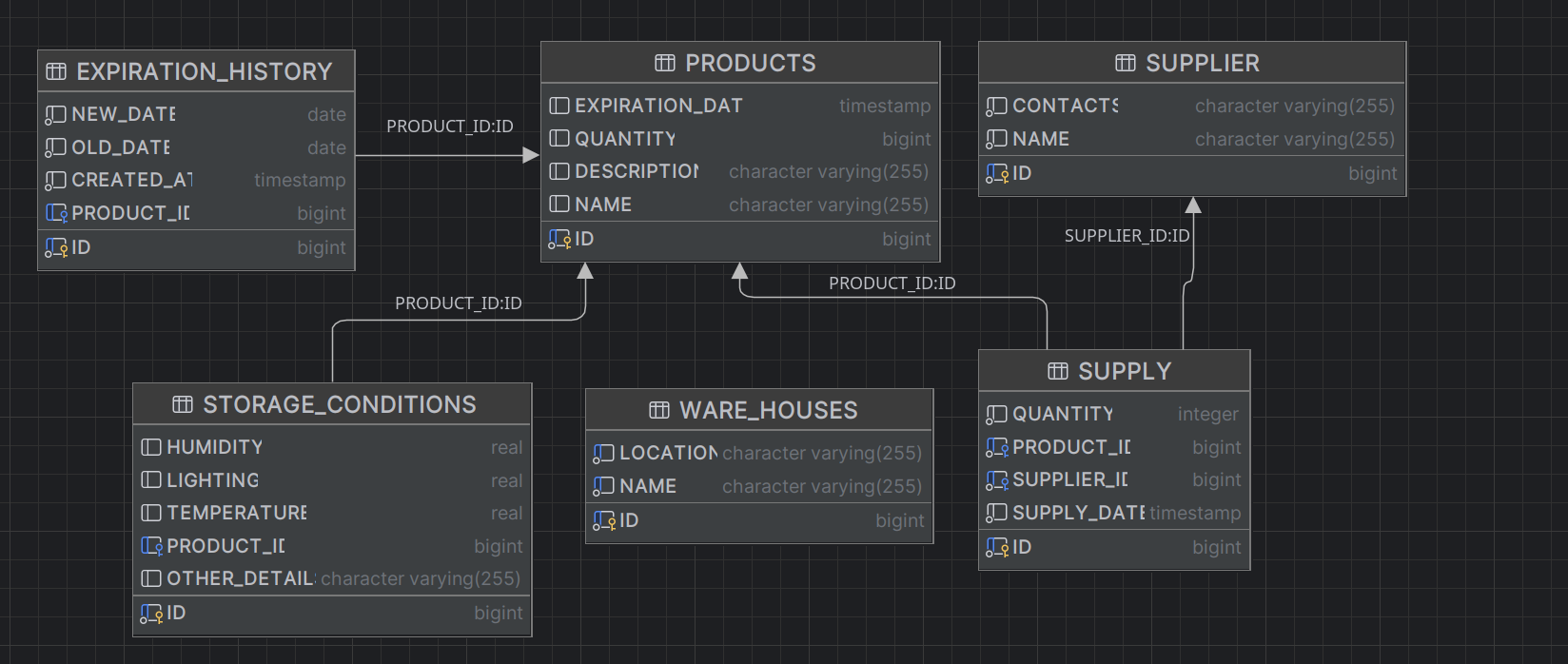
****

рис. - физическая модель базы данных

# **Диаграмма классов**

Диаграмма классов – это UML-диаграмма (англ. Unified Modeling Language – рус. Унифицированный язык моделирования), которая описывает связи между классами.

# **Диаграмма компонентов**

Диаграмма компонентов представляет из себя средство для визуализации, позволяющее получить представление об организации компонентов системы и взаимосвязей между ними.

# **Диаграмма вариантов использования**

Пример диаграммы последовательностей



# **Диаграмма последовательности**

Диаграмма последовательности – это схема, иллюстрирующая как происходит взаимодействие различных частей системы друг с другом, также она показывает порядок выполнения определенных функций в проекте. Данный вид диаграммы приведен на рисунке под номером 8.

# **Демонстрация работы приложения**

# **Список используемых источников**

1. Бен Фрейн (Ben Frain) - "HTML5 и CSS3. Разработка сайтов для любых браузеров и устройств", Питер, 2017, 272 стр. (ориг. название: "Responsive Web Design with HTML5 and CSS3", Packt Publishing)
2. Лия Веру - "Секреты CSS. Идеальные решения ежедневных задач", Питер, 2017, 336 стр. (ориг. название: "CSS Secrets. Better solutions to everyday web design problems", O'Reilly)
3. Эрик А. Майер - "CSS. Карманный справочник", Вильямс, 2017, 288 стр. (ориг. название: "CSS Pocket Reference", O'Reilly)
4. Джон Дакетт - "HTML и CSS. Разработка и дизайн веб-сайтов", Эксмо, 2017, 480 стр. (ориг. название: "HTML и CSS: Design and Build Websites", John Wiley & Sons)
5. Крис Минник, Эд Титтел - "HTML5 и CSS3 для чайников ", Диалектика, 2016, 400 стр. (ориг. название: "Beginning HTML5 and CSS3 For Dummies", John Wiley & Sons)
6. А. Хрусталев, А. Кириченко "HTMLS + CSS3. Основы современного WEB-дизайна", Наука и Техника, 2018, 352 стр.

# **Приложение А – Листинг программного кода**

Change.php

<?php

    require\_once '../config/connect.php';

    $id = $\_POST['id'];

    $title = $\_POST['title'];

    $subtitle = $\_POST['subtitle'];

    $description = $\_POST['description'];

    $SQL = "UPDATE `book` SET `title` = '$title', `sub\_title` = '$subtitle', `description` = '$description'

                WHERE `book`.`id` = '$id'";

    mysqli\_query($connect, $SQL);

    header('Location: ../index.php');

?>

CreateBook.php

<?php

    require\_once '../config/connect.php';

    $title = $\_POST['title'];

    $subtitle = $\_POST['subtitle'];

    $author = $\_POST['author'];

    $genre = $\_POST['genre'];

    $description = $\_POST['description'];

    $SQL = "INSERT INTO `book` (`id`, `title`, `sub\_title`, `genre\_name`, `author`, `description`) VALUES (NULL, '$title', '$subtitle', '$genre', '$author', '$description');";

    mysqli\_query($connect, $SQL);

    header('Location: ../index.php');

?>

CreateAuthor.php

<?php

    require\_once '../config/connect.php';

    $author\_name = $\_POST['author'];

    $authorInfo = mysqli\_query($connect, "SELECT \* FROM `author`");

    $authorInfo = mysqli\_fetch\_all($authorInfo);

    $isChange = false;

    foreach($authorInfo as $item){

        if($item[1] != $author\_name){

            $isChange = true;

        }

    }

    if($isChange){

        mysqli\_query($connect, "INSERT INTO `author` (`id`, `name`)

VALUES (NULL, '$author\_name')");

    }

    if(!$isChange){

        echo "<script>alert('Такой автор уже имеется в базе данных!')</script>";

    }

    header('Location: author.php');

?>

CreateGenre.php

<?php

    require\_once '../config/connect.php';

    $genre\_name = $\_POST['genre'];

    $genreInfo = mysqli\_query($connect, "SELECT \* FROM `genre`");

    $genreInfo = mysqli\_fetch\_all($genreInfo);

    $isChange = false;

    foreach($genreInfo as $item){

        if($item[1] != $genre\_name){

            $isChange = true;

        }

    }

    if($isChange){

        mysqli\_query($connect, "INSERT INTO `genre` (`id`, `name`) VALUES (NULL, '$genre\_name')");

    }

    if(!$isChange){

        echo "<script>alert('Такой автор уже имеется в базе данных!')</script>";

    }

    header('Location: genre.php');

?>

Delete.php

<?php

    require\_once '../config/connect.php';

    $id = $\_GET['id'];

    mysqli\_query($connect, "DELETE FROM `book` WHERE `book`.`id` = '$id'");

    header('Location: ../index.php');

?>

Connect.php

<?php

    $connect = mysqli\_connect('localhost', 'root', '', 'books');

    if(!$connect){

        echo 'Failed to connect bd';

    }

?>