**Государственное бюджетное общеобразовательное учреждение города Москвы «Школа № 2098 «Многопрофильный образовательный центр» имени Героя Советского Союза Л.М. Доватора»**

**FDB-ar — ВЕБ-САЙТ ПО УЧЁТУ КРЕПЕЖА НА СКЛАДЕ**

Выполнил:

Ученик 10”Т” класса

ГБОУ Школа № 2098 имени Героя Советского Союза Л.М. Доватора

Бардин Константин Алексеевич

Научный руководитель:

Учитель информатики

ГБОУ Школа № 2098 имени Героя Советского Союза Л.М. Доватора

Гоптарь Евгений Андреевич

**Москва, 2024 год**

Оглавление:

1. Введение (Актуальность работы, Обоснование выбора темы)
2. Цель работы
3. Задачи работы
4. Гипотеза
5. Методика выполнения работы
6. Результаты
7. Практическая и теоретическая значимость
8. Перспективы развития
9. Выводы
10. Список используемой литературы

1. Введение

В современном мире робототехнические системы стали неотъемлемой частью нашей жизни, а информация является ключевым источником знаний. Люди, владеющие информацией, способны результативно выполнять задачи, достигая намеченной цели, а визуализация информации играет в этом решающую роль, обеспечивая не только понимание, но и наиболее эффективную работу человека по изучению данных.

2. Цель работы

Именно поэтому целью моего проекта является создание веб-сайта, который станет виртуальной платформой для контроля и учета различных товаров на складе, включая крепежные элементы. Этот проект нацелен на обеспечение максимальной эффективности работы и улучшения качества регулирования используемых компанией материалов.

3. Задачи работы

1. Разработать удобный интерфейс, который будет вдохновлять пользователей на эффективное сотрудничество и новые достижения.
2. Создать систему учета, которая позволит легко и точно отслеживать наличие товаров.
3. Добавить функционал, который позволит пользователям вносить изменения в базу данных, делая процесс управления еще более гибким.
4. Интегрировать возможность формирования листа закупок на основе актуального ассортимента товаров на складе.

4. Гипотеза

Я предполагаю, что создание веб-сайта для учета и контроля товаров на складе, включая даже самые мелкие детали, такие как крепёж, улучшит процессы управления материалами, сократит время на учет и заказы товаров и стимулирует производительность и сотрудничество в команде.

5. Методика выполнения работы

Я использую методику гибкой разработки ПО (Agile), которая позволяет быстро адаптироваться к изменениям и реагировать на потребности пользователей. В рамках этого метода я разрабатываю серверную часть веб-сайта с использованием языка программирования Python (Back-end). Python обеспечивает высокую производительность и удобство в разработке серверных приложений.

Для обработки данных и взаимодействия с базой данных я использую различные библиотеки и фреймворки Python, такие как Flask и его составляющую часть Jinja2 (шаблонизатор). Эти инструменты позволяют создавать мощные и надежные веб-приложения, способные эффективно обрабатывать информацию о товарах и заказах. Для создания веб-сайта также используются современные технологии веб-разработки, такие как HTML, CSS, JavaScript (Front-end), а также базы данных для хранения информации о товарах и заказах.

Мои эксперименты включают в себя тестирование различных методов хранения и обработки данных, оптимизацию производительности сервера и интерфейса, а также анализ поведения пользователей для улучшения пользовательского опыта. Я стремлюсь создать веб-сайт, который не только удовлетворяет текущие потребности компании, но и готов к масштабированию и дальнейшему развитию.

6. Результаты

Я ожидаю, что внедрение нового веб-сайта с функционалом учета и контроля товаров на складе значительно упростит процессы управления материалами и сократит время, затрачиваемое на заказ и учет товаров. В мои планы входит обучение персонала компании перед использованием новой системы, а также её регулярное обновление при наличии изменений ассортимента и потребностей компании.

7. Практическая и теоретическая значимость

Разработка веб-сайта для учета и контроля товаров на складе представляет собой важный шаг в оптимизации процессов управления материалами. Этот проект направлен на улучшение эффективности учета и заказов товаров, что в свою очередь способствует повышению производительности и сокращению времени, затрачиваемого на административные процессы. Обучение персонала и регулярное обновление системы гарантируют, что веб-сайт будет актуален и полезен в повседневной деятельности компании.

С точки зрения теоретической значимости, использование гибкой методологии разработки ПО (Agile) демонстрирует применение современных подходов к управлению проектами, а также способствует адаптации к изменяющимся требованиям пользователей. Применение различных библиотек и фреймворков Python, таких как Flask и Jinja2, подчеркивает важность использования современных технологий в веб-разработке.

В моём проекте присутствуют данные файлы JavaScript:

* script1.js — в нём написаны функции, отвечающие за перенаправление на другие страницы сайта, которые активируются при нажатии на кнопки и текстовые ссылки на страницах сайта. Также в этом скрипте прописаны некоторые общие скрипты, которые должны работать на всех страницах сайта, такие как:
* запрет делать скриншот на странице (в данный момент закомментирована, но может быть в любой момент активирована)
* запрет на нажатие правой кнопки мыши на сайте
* запрет комбинации клавиш Ctrl + U (просмотр кода страницы) и Ctrl + S (сохранить страницу как) и Ctrl + P (печать страницы)
* script2.js, script2-3\_3.js, script2-1-2-3\_1-2.js — все эти скрипты имеют общий смысл, но разделены конкретные страницы.

script2.js – общий скрипт, который применяется для всех страниц и отвечает за изменения background-color у выбранного блока с class=” grid-item”. Также в нём присутствует функции getCurrentBlockId() и getСurrentPageUrl(), первая для получения текущего ID выбранного блока, вторая для получения ссылки на текущую страницу.

В script2-3\_3.js раньше был написан код, который отдельно отвечает за страницу "/map/stellazh3/shelf3", так как она отвечает за полку стеллажа с необычными ячейками. В течении преобразований и изменений кода, данный скрипт не имеет функциональности и код в нём закомментирован.

В script2-1-2-3\_1-2.js написана функция для генерации сетки с номерами блоков. Данный скрипт запускается при загрузке страниц всех полок стеллажей (у 1-ого стеллажа нет полок, поэтому при загрузке страницы "/map/stellazh1" данный скрипт тоже выполняется). В данную функцию generateGrid(columns, rows) поступают значения columns и rows, которые прописываются в коде HTML отдельно для каждой страницы.

* script3-database.js, script3-database-page3.js — эти скрипты отвечают за работу с данными из базы данных.

script3-database-page3.js работает только для страницы "/page3" и функции в нём вызываются при нажатии на кнопку с id="btn-confirm-of-input-line\_number-container". Функция submitAnswers\_page3() отвечает за получение данных с сервера, на котором осуществляется получение данных из базы данных, что заключается в отправке на сервер line\_number (номер строки базы данных, которое вбили в окно ввода <input type="number" id="line\_number">, чтобы получить на сайт значения, записанные для этой ячейки) и возвращение на сайт данных в виде словаря "data", содержащего данные об этой ячейке.

script3-database.js является самым объёмным и сложным. Его описание будет представлено в виде блок схемы, которая будет находиться ниже в конце данного пункта “Практическая и теоретическая значимость” документации (также это файл будет храниться в папке “resources” моего проекта под названием “Функциональная\_схема.svg”)

Также в моём проекте присутствуют файлы HTML, каждый из которых создан отдельно для каждой из страниц сайта.

Помимо этого, есть файлы CSS, такие как:

* style\_base.css — в нём прописаны стилевые свойства для блоков, находящихся на всех страниц, так как этот стилевой файл используется в base.html, который в свою очередь при помощи Jinja2 используется в качестве шаблона для всех остальных страниц класса и расширяется с помощью конструкции {% block content %} {% endblock %} всеми остальными страницами.
* style\_stellazh.css — в данном стилевом файле прописываются свойства для блоков, находящихся на страницах относящимся к полок (в случае Стеллажа 1 прописан именно для него, так как у него нет полок)

Также в моём проекте присутствует Python часть кода, в которой происходит работа с серверной частью, соответственно при запуске этого файла активируется локальный сервер. Его описание, а также описание работы script3-database.js будет представлено в виде блок схемы, которая будет находиться ниже в конце данного пункта “Практическая и теоретическая значимость” документации (также это файл будет храниться в папке “resources” моего проекта под названием “Функциональная\_схема.svg”)

8. Перспективы развития

Планируется расширение функционала веб-сайта для автоматизации процессов заказа и поставки продукции, а также для предоставления клиентам актуальной информации о наличии товаров и состоянии заказов. Также возможно развитие системы аналитики и отчетности для управленческого контроля за процессами складского учета и управления запасами.

В настоящее время для ускорения процесса разработки веб-сайта и упрощения работы с данными я использую CSV-файлы в качестве базы данных. Планируется разработка функциональности, которая автоматически будет преобразовывать данные из CSV в базу данных с использованием SQLAlchemy. Это поможет улучшить управление данными и обеспечит более эффективную работу с сайтом.

9. Выводы

Создание веб-сайта для учета и контроля товаров на складе является важным шагом в оптимизации процессов управления материалами компании. Применение современных методологий разработки и технологий веб-разработки позволяет создать мощный и гибкий инструмент, способный эффективно управлять информацией о товарах и заказах. Расширение функционала веб-сайта может значительно улучшить не только внутренние процессы компании, но и взаимодействие с клиентами и поставщиками.

10. Список используемой литературы

1. Flask Documentation [Электронный ресурс]. URL: <https://flask.palletsprojects.com/> (дата обращения: 18.02.2024).
2. Jinja2 Documentation [Электронный ресурс]. URL: <https://crossjs.com/jinja2/> (дата обращения: 18.02.2024).
3. Jinja2 Documentation [Электронный ресурс]. URL: <https://jinja.palletsprojects.com/_/downloads/en/stable/pdf/> (дата обращения: 18.02.2024).
4. SQLAlchemy Documentation [Электронный ресурс]. URL: <https://docs.sqlalchemy.org/> (дата обращения: 18.02.2024).
5. Python Documentation [Электронный ресурс]. URL: <https://docs.python.org/> (дата обращения: 18.02.2024).
6. HTML Documentation [Электронный ресурс]. URL: <https://developer.mozilla.org/ru/docs/Web/HTML> (дата обращения: 18.02.2024).
7. CSS Documentation [Электронный ресурс]. URL: <https://developer.mozilla.org/ru/docs/Web/CSS> (дата обращения: 18.02.2024).
8. JavaScript Documentation [Электронный ресурс]. URL: <https://developer.mozilla.org/ru/docs/Web/JavaScript> (дата обращения: 18.02.2024).