МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра математического и компьютерного моделирования

Разработка и проектирование информатиционной системы интернет магазина «Набор для мини стройки»

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 441 группы

направление 09.03.03 — Прикладная информатика

механико-математического факультета

Никитина Олега Дмитриевича

Научный руководитель
доцент, к.ф-м.н., доцент _____ Е.Ю.Крылова

Зав. кафедрой
зав. каф., д.ф.-м.н., доцент _____ Ю.А. Блинков

Саратов 2025

Введение. В условиях стремительного роста интереса к ручному труду, хобби и образовательным игрушкам, создание доступной и функциональной онлайн-платформы становится актуальной задачей. Наборы для мини-сройки представляют собой уникальные изделия, сочетающие в себе элементы игры, обучения и творчества. Они позволяют пользователям, как детям, так и взрослым, развивать мелкую моторику, пространственное мышление и навыки конструирования. Однако на рынке ощущается дефицит качественных решений для продажи подобной продукции через интернет, особенно в сегменте малых и средних производителей. Целью данной работы является разработка удобного, интуитивно понятного и технологически современного веб-приложения для представления и продажи наборов для мини-стройки. Основное внимание уделяется пользовательскому опыту, скорости работы приложения, надежности хранения данных и безопасности пользователей.

Для достижения нужного результата предстоит решить следующие задачи:

-Проанализировать требования к целевой аудитории и функционалу системы;

-Спроектировать архитектуру веб-приложения с учетом современных требований;

-Разработать структуру базы данных для хранения информации о пользователях, товарах и заказах;

-Реализовать клиентскую и серверную части системы;

-Провести тестирование работоспособности приложения и подготовить рекомендации по его дальнейшему развитию.

Таким образом, разработка веб-приложения для продажи наборов для мини-стройки представляет собой актуальную и практически значимую задачу, решение которой требует комплексного применения знаний в области веб-технологий, баз данных, проектирования интерфейсов и программной инженерии.

Описание структуры. Работа состоит из введения, пяти разделов, в которых содержится теоретический материал, заключения.

Первый раздел содержит анализ требований к веб-приложению.

Основные функциональные требования к веб-приложению включают:

-Каталог товаров;

-Отображение категорий товаров;

-Просмотр списка товаров в выбранной категории;

-Просмотр подробной информации о товаре;

-Пользовательская регистрация и авторизация;

-Возможность создания учетной записи;

-Вход в личный кабинет;

-Выход из системы;

-Корзина покупок;

-Добавление товара в корзину;

-Удаление товара из корзины;

-Просмотр содержимого корзины;

-Учет пользователей;

-Хранение информации о пользователях в базе данных;

-Разграничение прав доступа (пользователь, администратор);

-Система управления товарами (для администратора);

-Добавление новых товаров;

-Редактирование информации о товарах;

-Удаление товаров;

-Страница "О нас"с описанием компании;

-Страница "Контакты"с информацией для связи.

Второй раздел посвящен проектированию архетиктуры веб-приложения.

Целевой аудиторией веб-приложения являются:

1)Покупатели-конечные пользователи:

- Возраст: от 8 до 65 лет;

- Интересы: моделизм, рукоделие, DIY-хобби, коллекционирование;

- Технологическая подготовка: от новичков до уверенных пользователей интернета и смартфонов;

- Мотивы: желание купить набор для творческого досуга, подарить оригинальный сувенир, украсить интерьер ручной работой.

2. Администраторы контента:

- Роль: сотрудники компании, отвечающие за наполнение каталога, актуализацию цен, загрузку новых изображений;

- Навыки: уверенное знание REST, умение работать в админ-панели или прямо в базе данных;

- Задачи: создавать/редактировать/удалять категории и товары, контролировать целостность данных, следить за загрузкой изображений.

3. Менеджеры по работе с клиентами:

- Роль: специалисты поддержки и обработки заказов;

- Навыки: работа с CRM-системами (или планируемая интеграция), коммуникация с покупателями по email/телефону;

- Задачи: отслеживать статусы заказов, запрашивать у клиентов уточняющие данные, организовывать доставку.

Основные бизнес-процессы, поддерживаемые системой:

1)Процесс покупки:

Пользователь выбирает товар, добавляет его в корзину, оформляет заказ.

2)Процесс регистрации нового пользователя:

Заполнение регистрационной формы с указанием личных данных.

3)Процесс администрирования контента:

Администратор добавляет и обновляет товары, редактирует категории.

Эти процессы обеспечивают базовую работу интернет-магазина и позволяют расширять функционал в будущем.Эти процессы мы можем наблюдать в соответствии с рисунками 1,2 и 3.

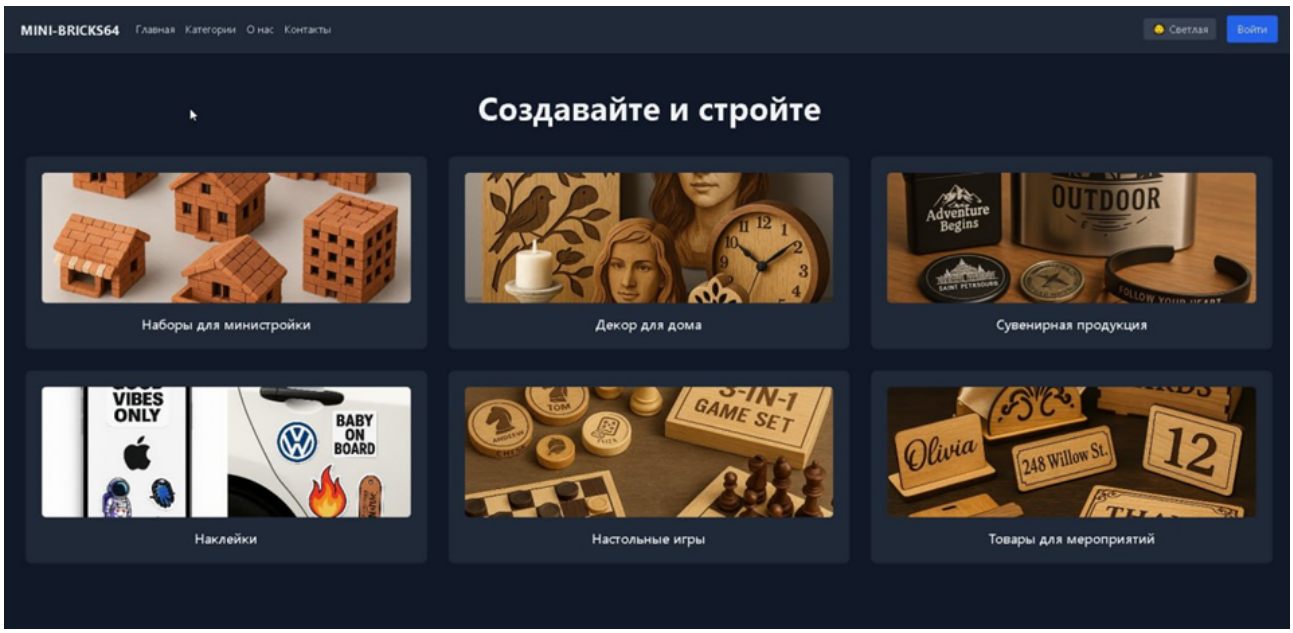

Рисунок 1 — Главное меню.

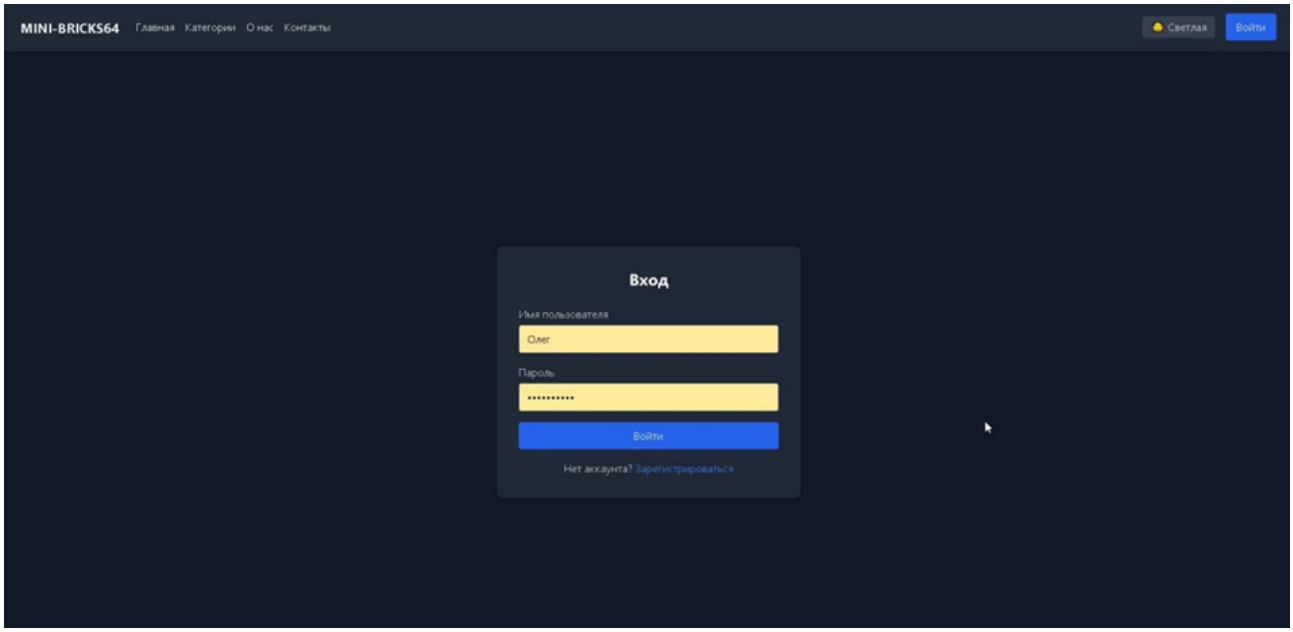

Рисунок 2 — Окно входа/регистрации.

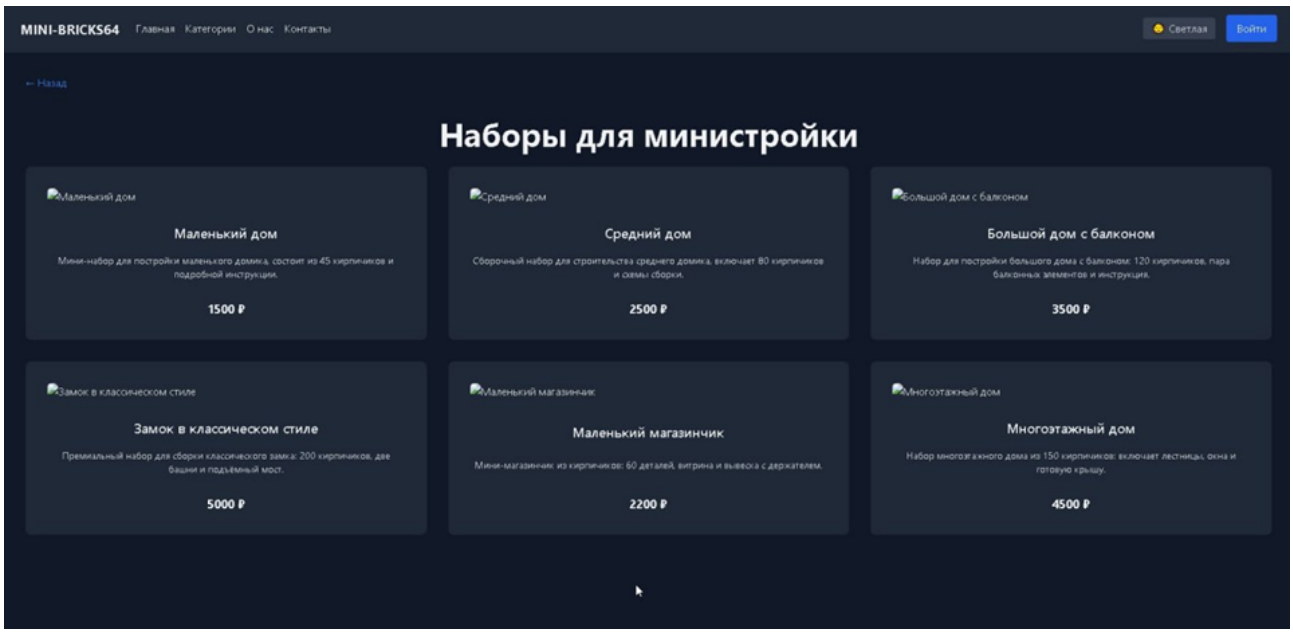

Рисунок 3 — Список товаров в одном из разделов.

Третий раздел посвящен проектированию и созданию базы данных.

Была выбрана монолитная архетиктура.

Причины выбора монолита для MiniBriks64:

1. Простота разработки и деплоя. Все контроллеры, сервисы и репозитории находятся в одном Spring Boot-приложении;

2. Небольшой объём функционала на первом этапе. Нужен единый API для категорий, товаров и авторизации;

3. Отсутствие высоких нагрузок. Проект рассчитан на десятки–сотни одновременных пользователей, с которыми монолит справляется без потерь в производительности;

4. Экономия ресурсов. Нет необходимости сразу настраивать систему оркестрации (Kubernetes, сервис-дискавери).

Перспектива: в будущем, при росте числа клиентов и расширении функционала (корзина, заказ, отзывы), архитектуру можно эволюционно перевести на микросервисы, вынеся, например, модуль заказов и оплату в отдельный сервис.

Для корректного функционирования веб-приложения была спроектирована база данных, включающая в себя три сущности:

-Пользователь (User): id — уникальный идентификатор пользователя (тип: Long), username — имя пользователя (уникальное, тип: String), email

— адрес электронной почты пользователя (уникальное, тип: String), password — зашифрованный пароль пользователя (тип: String), role — роль пользователя в системе .

-Категория товара (Category): id — уникальный идентификатор категории (тип: Long), name — наименование категории (уникальное, тип: String), категории (тип: String).

-Товар (Product): id — уникальный идентификатор товара (тип: Long), name — название товара (тип: String), description — подробное описание товара (тип: String), price — стоимость товара (тип: Double).

Взаимосвязи между сущностями:

-Один пользователь может иметь одну роль;

-Одна категория может содержать множество товаров;

-Каждый товар обязательно принадлежит к одной категории.

Визуальная схема:

В качестве системы управления базами данных выбрана PostgreSQL по следующим причинам:

-Надежность и отказоустойчивость;

-Поддержка сложных связей между таблицами;

-Расширенная работа с текстовыми и JSON-полями;

-Высокая производительность при обработке большого объема данных;

-Открытый исходный код и активное сообщество разработчиков.

Версия PostgreSQL, использованная в проекте: PostgreSQL. Подключение базы данных происходит через Spring Boot с использованием JDBC URL.

В четвертом разделе была выполнена реализация веб-приложения. Разработка веб-приложения для продажи наборов для мини-стройки выполнялась с применением архитектуры "клиент–сервер где:

-Клиентская часть (Frontend) реализована на JavaScript с использованием библиотеки React;

-Серверная часть (Backend) построена на Spring Boot — мощном фреймворке для создания REST API на Java;

-Взаимодействие между фронтендом и бэкендом осуществляется по HTTP через REST API;

-Для хранения данных используется СУБД PostgreSQL;

-Приложение разработано с учётом масштабируемости, расширяемости и модульности;

-Логика разделена по слоям: контроллеры, сервисы, репозитории, DTO и конфигурационные компоненты.Фронтенд разработан на базе React;

-Все компоненты разделены на логические части:

1)HomePage.js — отображение списка категорий

2)LoginPage.js — форма входа и регистрации;

3) CategoryProductsPage.js — отображение товаров из конкретной категории;

4)Header.js — навигационная панель;

5)ThemeContext.js и AuthContext.js — глобальные состояния темы и авторизации;

6)Для стилизации приложения используется Tailwind CSS, позволяющий писать гибкие и адаптивные стили прямо в JSX.

В пятом разделе было произведено тестирование клиентской части.

Проверены переходы между страницами:

1)С главной страницы на страницу логина;

2)С главной страницы на страницу конкретной категории;

3)Возврат на главную страницу с любой вложенной.

Все переходы работают с использованием react-router-dom.

Отображение интерфейса:

Проверено: Корректное отображение заголовков, кнопок, карточек; Смена темы (тёмная/светлая) и её сохранение между перезагрузками; Адаптивность (отображение на разных размерах экрана, включая мобильный).

Авторизация и регистрация:

Проверены сценарии: Успешная регистрация нового пользователя; Попытка регистрации с уже существующим логином/почтой; Успешный вход пользователя; Отображение и скрытие кнопки "Войти"/значка пользователя в шапке в зависимости от статуса; Сохранение пользователя в localStorage.

Работа с категориями и товарами:

При открытии страницы категорий данные приходят с сервера и отображаются корректно; Отображаются изображения категорий; При нажатии на

категорию открывается страница с товарами; Название и описание товара выводятся из базы данных.

База данных:

Проверено заполнение базы через DataInitializer; Убедились, что изображения категорий сохраняются как строки с путём /images/categories/...; Проверена работа ORM и соответствие сущностей таблицам; Проверена уникальность username и email на уровне базы данных.

Тестирование безопасности:

Проверена блокировка доступа к закрытым эндпоинтам без авторизации (если бы они были реализованы); Убедились, что пароли сохраняются в зашифрованном виде с использованием BCrypt; CSRF отключён, что подходит для REST API, взаимодействующего с SPA-приложением (например, React).

Обработка ошибок:

На фронтенде реализована логика отображения ошибок:

1)При вводе некорректных данных (например, неправильный логин/пароль) отображается сообщение;

2)При сбое подключения к серверу — информирование пользователя о проблеме; Все ошибки логируются в консоли и обрабатываются безопасно.

В результате тестирования установлено:

1)Веб-приложение соответствует заявленным требованиям;

2)функциональность реализована в полном объёме;

3)Интерфейс работает корректно во всех актуальных браузерах;

4)Все основные сценарии пользовательского взаимодействия отлажены;

5)Обеспечена стабильная работа с базой данных;

6)Реализована безопасность хранения паролей и защита от некорректного ввода.

Таким образом, веб-приложение MiniBriks64 готово к дальнейшему наполнению и использованию в демонстрационных целях.

Заключение. Разработка веб-приложения MiniBriks64 стала полноценным проектом, включающим в себя как техническую реализацию, так и логическое проектирование всех его компонентов. В процессе работы над ВКР была достигнута основная цель — создать удобную, современную и расши-

ряемую платформу для продажи наборов мини-стройки и сопутствующих товаров.

Подведение итогов

В ходе реализации проекта были выполнены следующие ключевые задачи:

- Проведён анализ требований, определена целевая аудитория и функциональные потребности, что позволило сформировать обоснованный план разработки;

- Разработана архитектура приложения, включающая серверную и клиентскую части, взаимодействие с базой данных и структурирование кода по слоям;

- Спроектирована и реализована база данных, в которую входят сущности «Пользователи», «Категории» и «Продукты». Использована СУБД PostgreSQL, обеспечивающая надёжное хранение и работу с данными;

- Реализована регистрация и авторизация пользователей, включая защиту паролей и работу с сессиями через фронтенд;

- На клиентской части реализован интерфейс с использованием фреймворка React: пользователь может просматривать категории товаров, переходить к товарам внутри категории, регистрироваться и входить в систему;

- Обеспечена поддержка светлой и тёмной тем UI;

- Осуществлено отображение изображений категорий, включая работу со статическими ресурсами и настройку публичной папки на сервере;

- Выполнено тестирование функциональности, включая ручную проверку навигации, загрузки данных, регистрации и авторизации, а также корректности API.

Все поставленные в рамках диплома цели были достигнуты, что позволяет говорить об успешной реализации первой версии приложения.

Перспективы развития проекта

Проект MiniBriks64 имеет большой потенциал для дальнейшего развития.

В ближайшие этапы можно включить:

- Реализация корзины покупок и оформление заказов;

- Подключение платёжной системы (например, Stripe или ЮKassa);

- Разделение на роли пользователей (администратор, клиент);

- Личный кабинет пользователя с историей заказов;

- Загрузка изображений товаров через форму (не только из static);

- Система отзывов и оценок; Поддержка многоязычности;

- Адаптация под мобильные устройства с улучшением UI/UX;

- Интеграция с системой уведомлений (email или Telegram-бот);

- Контроль складских остатков и панель управления контентом.

Таким образом, текущая версия — это надёжный фундамент, на котором можно построить полноценную e-commerce платформу с широким функционалом.

Личностные и профессиональные результаты

В процессе работы над дипломной работой были приобретены и усилены следующие компетенции:

- Навыки проектирования и структурирования кода в многослойной архитектуре;

- Знания по развёртыванию и конфигурации Spring Boot-приложений;

- Углублённые навыки разработки на React и Tailwind CSS;

- Опыт интеграции фронтенда и бэкенда через REST API;

- Умение находить и устранять ошибки, возникающие в процессе сборки и выполнения;

- Опыт командной работы и взаимодействия с системой контроля версий (Git).

Таким образом, выполненный проект не только стал реализацией практической задачи, но и послужил важным этапом профессионального становления разработчика, способного создавать полноценные веб-приложения с нуля.