МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информатики и вычислительной техники

Кафедра информационной безопасности

Отчёт к курсовому проекту

по дисциплине “Безопасность систем баз данных”

**Разработка базы данных для строительной фирмы**

Выполнили: студенты группы БИ-31

Воротилов Т.Д., Иванов П.А., Лукьянов Д.А.

Проверил: доцент кафедры ИБ

Сучков Д.С.

Йошкар-Ола

2021 г.

**СОДЕРЖАНИЕ**

Оглавление

[Введение 3](#_Toc61462463)

[Техническое задание 4](#_Toc61462464)

[1.1 Требования к курсовой работе: 4](#_Toc61462465)

[1.2 Требования к базе данных 4](#_Toc61462466)

[1.3 Требование к API (минимальное количество реализованных методов) 4](#_Toc61462467)

[Порядок выполнения работы 5](#_Toc61462468)

[2.1 Этапы разработки базы данных 5](#_Toc61462469)

[2.2 Этапы разработки API 9](#_Toc61462470)

[2.3 Выбор хеш-функции 12](#_Toc61462471)

[Приложения 13](#_Toc61462472)

[Вывод 13](#_Toc61462473)

# Введение

В курсовой работе рассматривается создание базы данных, предназначенной для автоматизации работы строительной компании. База данных позволяет клиентам просматривать ассортимент продукции строительной компании. Реализована автоматизация оформления заказа, которая позволяет клиентам формировать заказ.

# Техническое задание

## 1.1 Требования к курсовой работе:

* Получить структуру данных из файла, согласно варианта. Привести к 3й нормальной форме. Добавить недостающие таблицы.
* Составить ER-диаграмму
* Разработать API для базы данных на любом языке, выполняющемся на стороне сервера (php, ASP.NET, Java, python, node.js, etc)
* Взаимодействие должно осуществляться по клиент-серверной архитектуре, подключение с клиентской программы недопустимо
* Провести настройку пользователей базы данных для разграничения прав доступа, привести пример конфигурации
* Все документы и исходные коды для курсовой работы должны храниться под контролем системы контроля версий — git или mercurial (https://github.com/, https://bitbucket.org/)
* Во время сдачи курсового проекта необходимо предоставить отчет о проделанной работе в печатном виде (отчет)

## 1.2 Требования к базе данных

* Наличие не менее 7 таблиц, в том числе таблицы сессий и пользователей
* Структура таблицы должна содержать не менее 3-х полей, одно из которых ключевое
* Правомерное использование типов данных
* Обязательно использование триггеров и/или хранимых процедур
* Форма нормализации не менее 3NF
* Индексирование по полям поиска

## 1.3 Требование к API (минимальное количество реализованных методов)

* аутентификация пользователя (создание сессии);
* добавление/удаление/изменение данных в таблицах;
* выборка данных их ключевых таблиц по запросам;
* выборка данных из таблиц с объединением результатов.

# Порядок выполнения работы

## 2.1 Этапы разработки базы данных

Разработана база данных, содержащая 8 таблиц. В том числе таблицы сессий (*usadba\_session*) и пользователей (*usadba\_users*). Для авторизации пользователя используется таблица - *usadba\_accounts,* содержащая информацию об аккаунте пользователя. Для хранения информации об ассортименте используются следующие таблицы:

*usadba\_action* – название продукции, описание и цена;

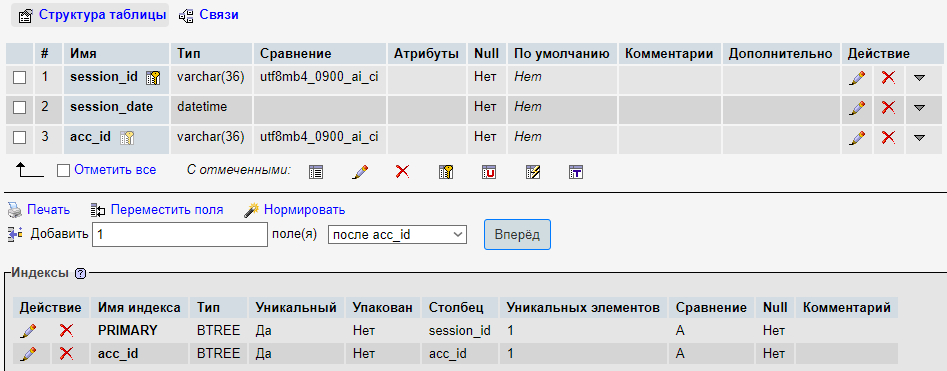
Для автоматизации оформления заказа, были созданы следующие таблицы:

*usadba\_availability –* баланс и платежные данные покупателя;

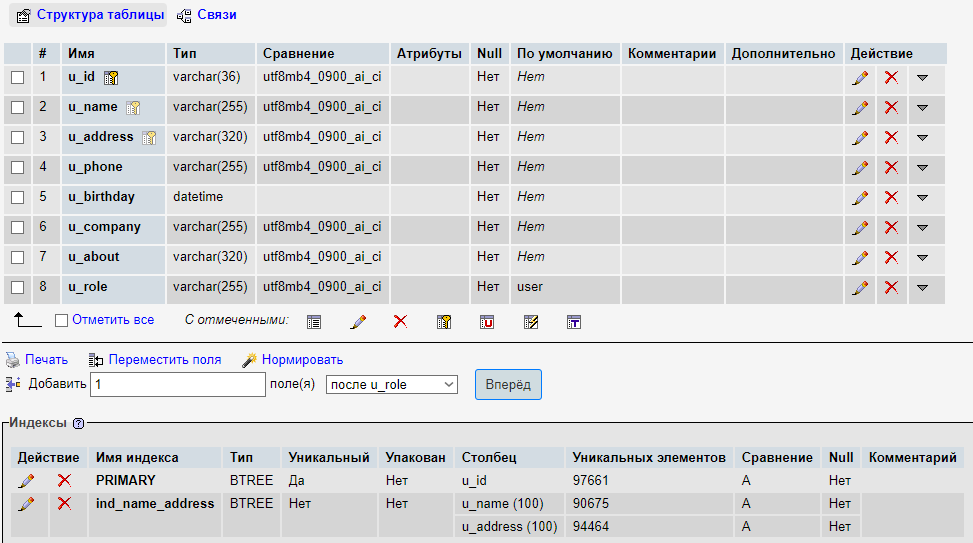
*usadba\_cart* – информация о роллах, добавленных в корзину;

*usadba\_orders* – оформленные заказы на доставку ролл;

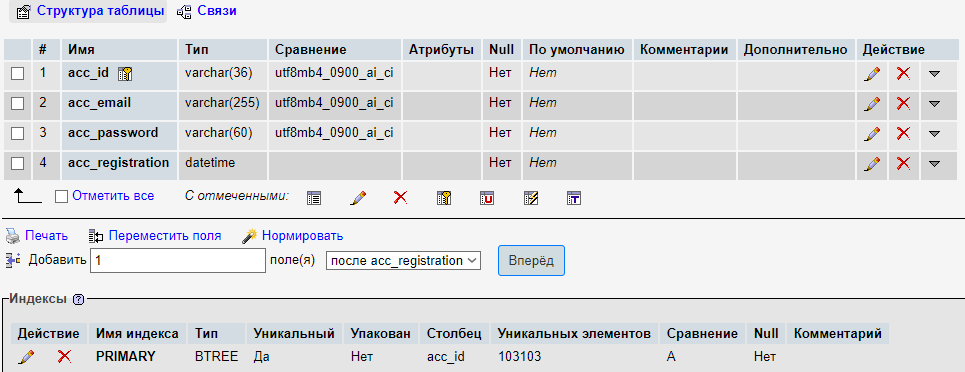
Структуры, реализованных таблиц:



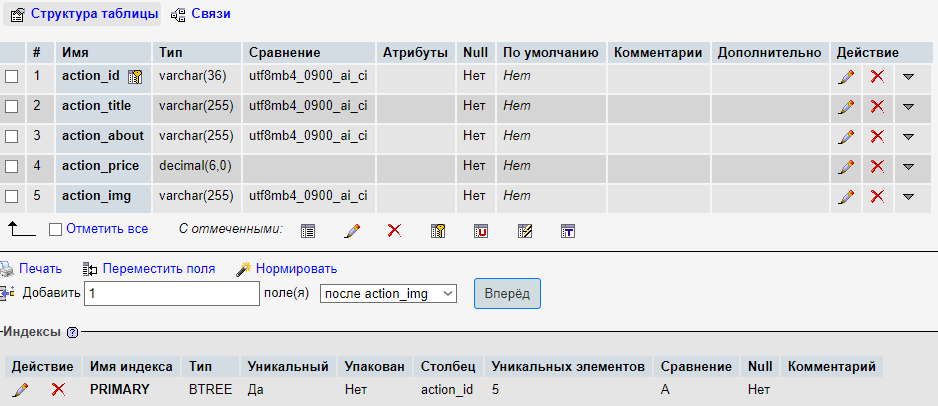
*Рис 1. usadba\_session*



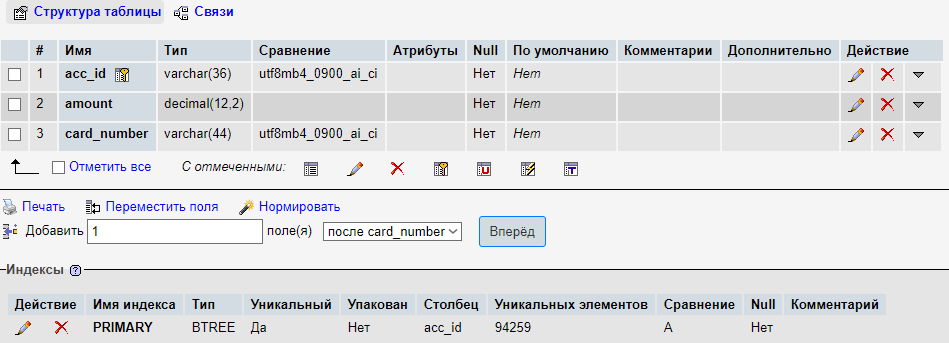
*Рис 2. usadba\_users*



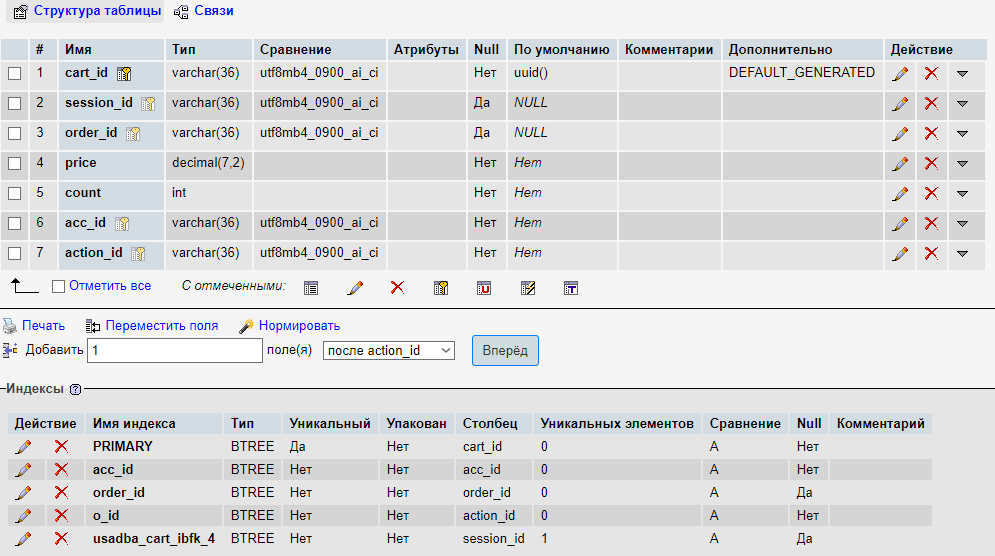
*Рис 3. usadba\_accounts*



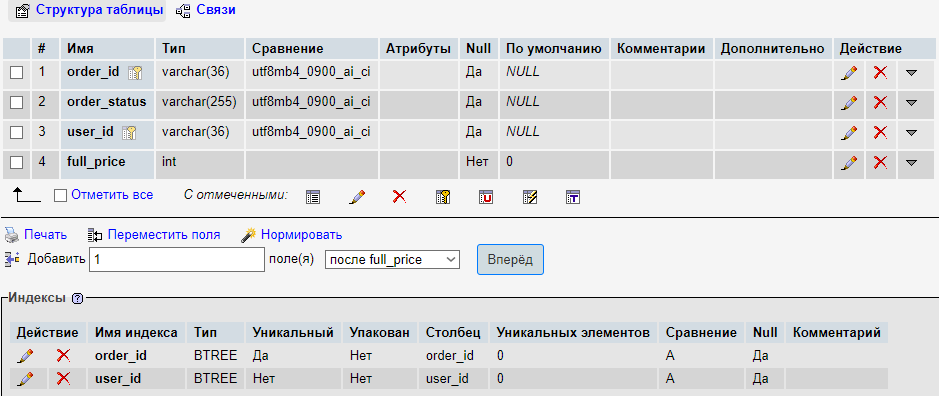
*Рис 4. usadba\_action*



*Рис 5. usada\_availability*



*Рис 6. usadba\_cart*



*Рис 7. usadba\_orders*

Используемые триггеры:

* *add\_payment –* триггер, отвечающий за добавление информации и платеже пользователя;

*DELIMITER $$*

*CREATE TRIGGER `add\_payment` BEFORE INSERT ON `usadba\_orders` FOR EACH ROW BEGIN*

*INSERT INTO usadba\_payment(payment\_buyer, amount)*

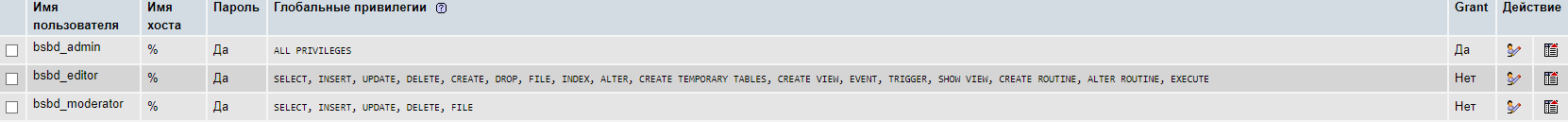
*VALUES(NEW.user\_id, NEW.full\_price);*

*END*

*$$*

*DELIMITER ;*

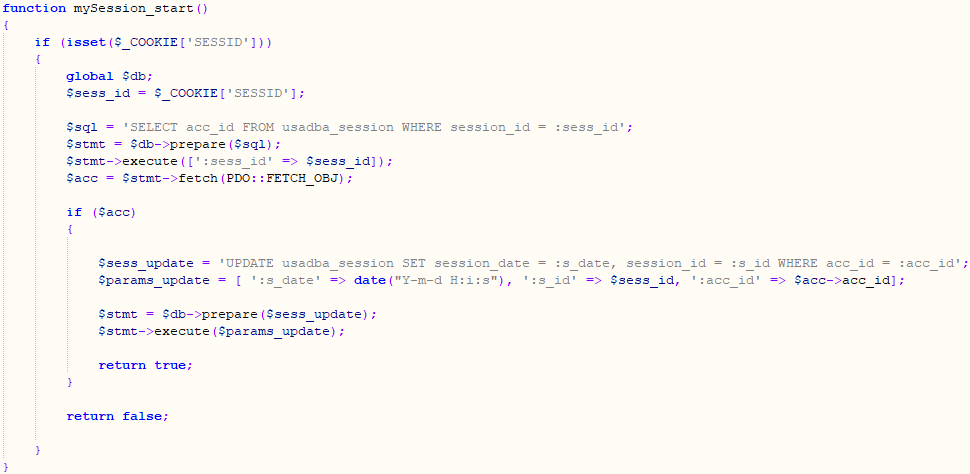
Проведена настройка пользователей базы данных для разграничения прав доступа:



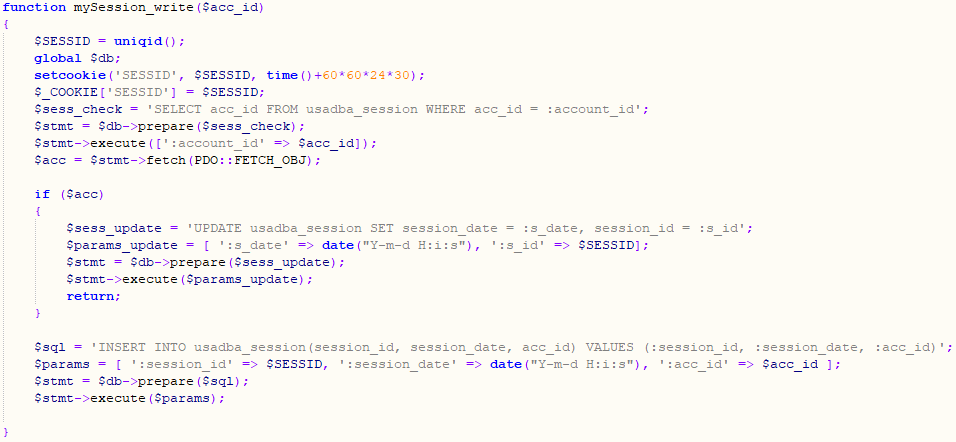
*Рис 8. пользователи базы данных*

## 2.2 Этапы разработки API

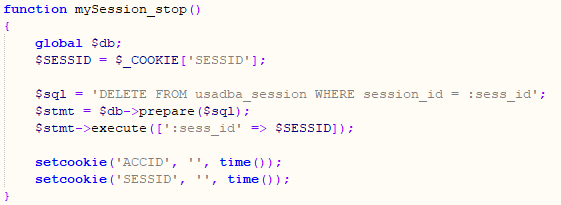
Было разработано API для аутентификации пользователя, состоящий из 3 функций, написанных на языке PHP.



*Рис 9. Создание сессии*



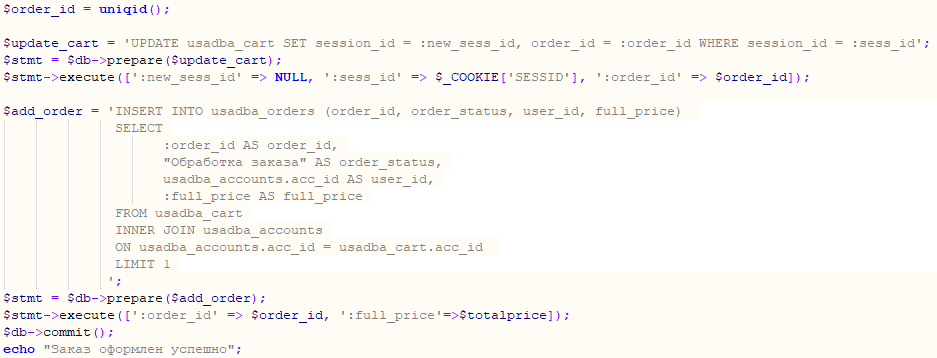
*Рис 10. Запись сессии*



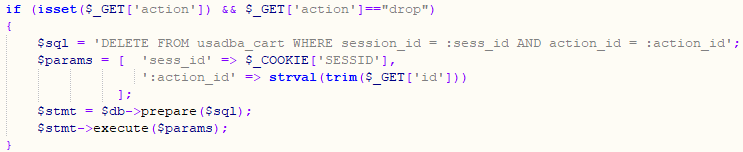
*Рис 11. Остановка сессии*

Разработана функция добавления, удаления и изменения данных.

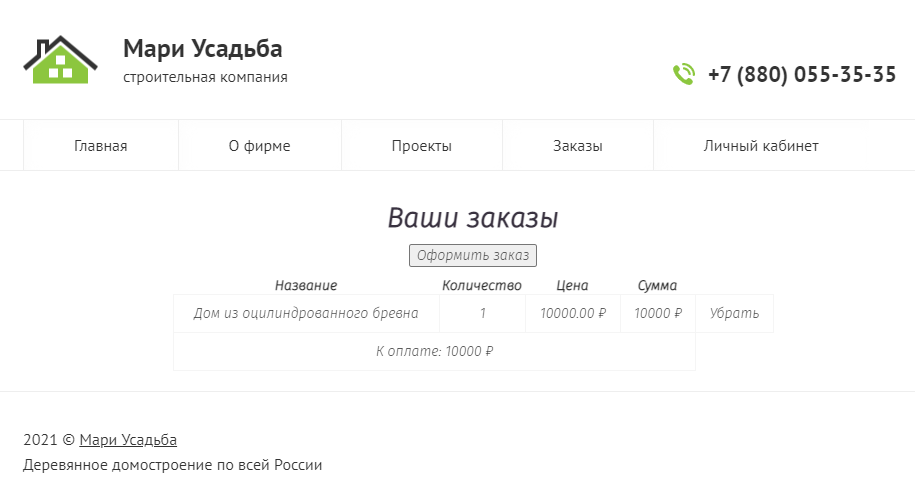
Добавлена функция вывода и объединения данных из ключевых таблиц.



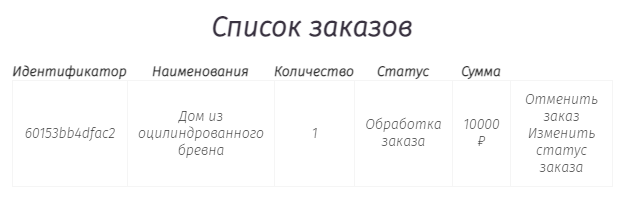
*Рис 12. Добавление и изменение данных*



*Рис 13. Удаление данных*



*Рис 14. Заказы покупателя*



*Рис 15. Список заказов (доступен только администратору)*

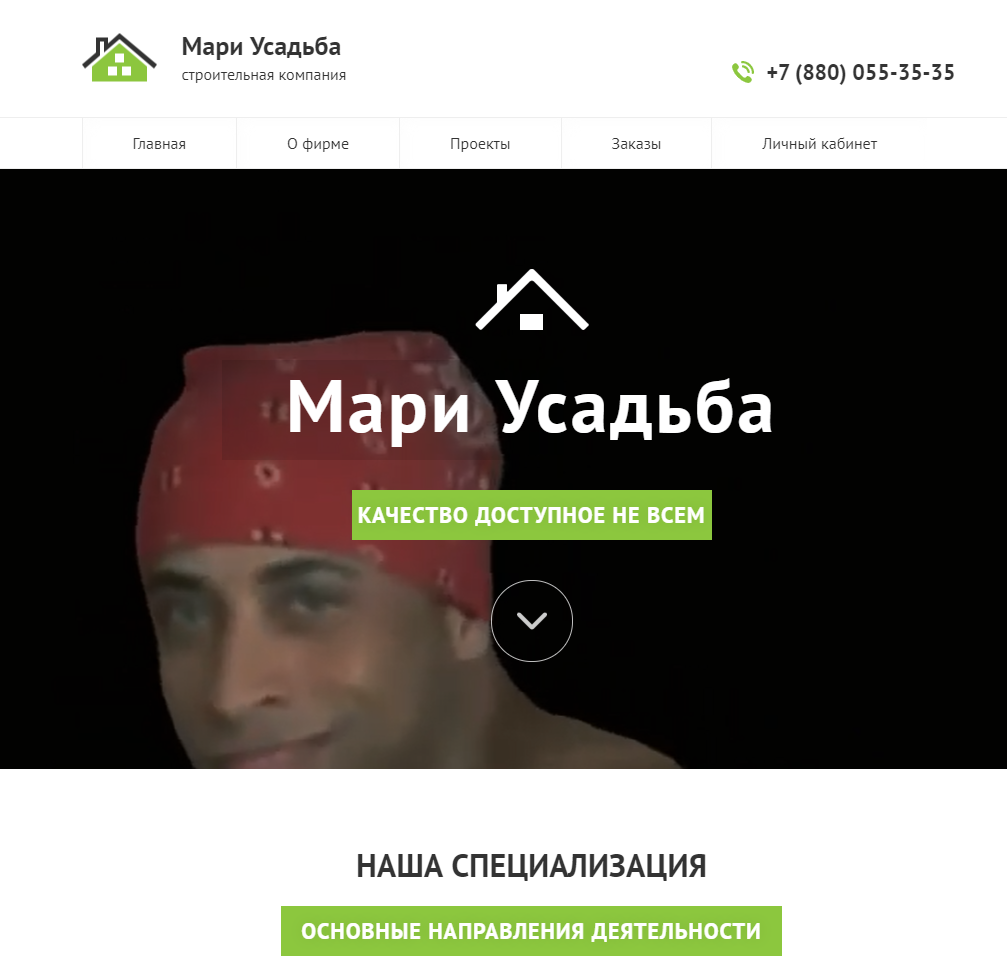


Рис.16 Главная страница

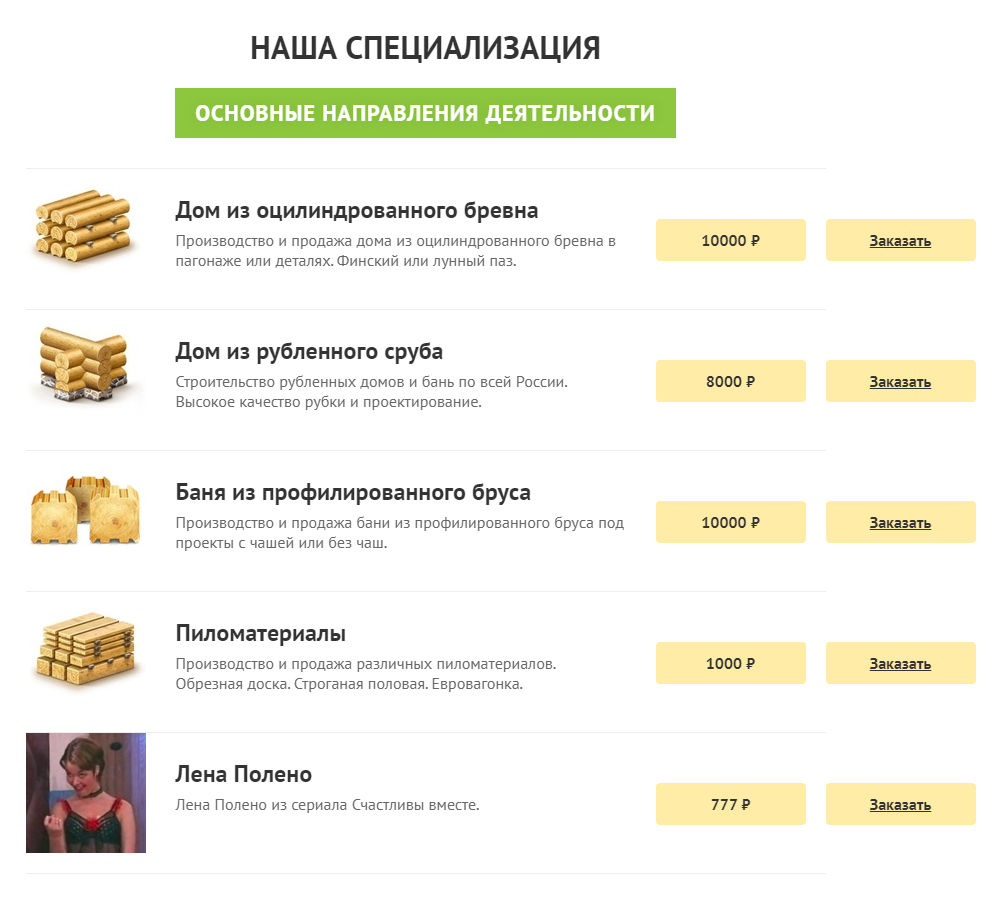


Рис.17 Главная страница (заказы)

## 2.3 Выбор хеш-функции

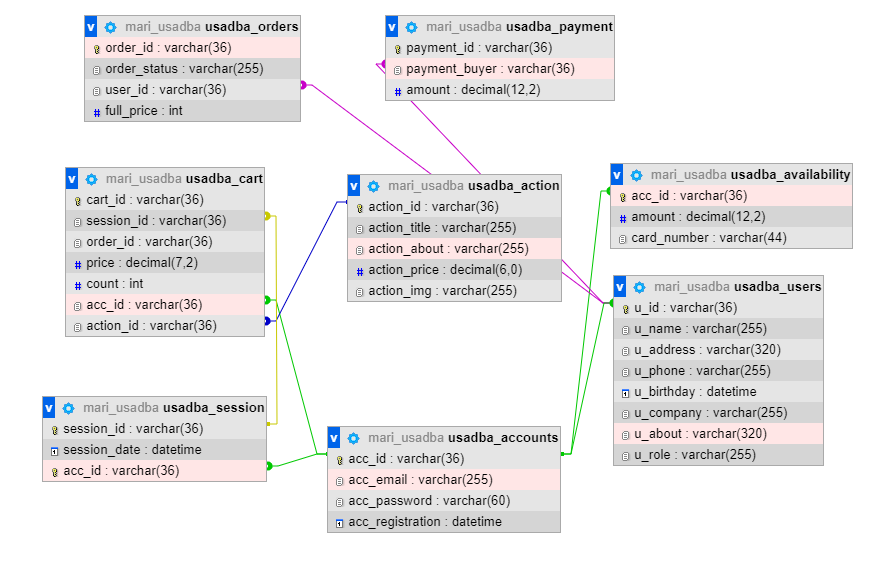
Для хеширования паролей пользователей используется хеш-функция bcrypt

bcrypt — адаптивная криптографическая хеш-функция формирования ключа, используемая для защищенного хранения паролей. Разработчики: Нильс Провос и David Mazières. Функция основана на шифре Blowfish, впервые представлена на USENIX в 1999 году.

BCrypt работает гораздо медленнее других методов хэширования, но его результат защищён гораздо сильнее. Ведь когда дело касается хэширования и шифрования, быстрее не означает лучше. Чем больше времени занимает шифрование, тем дольше будет длиться расшифровка.

# Приложения

1. ER-диаграмма



1. Исходные коды и документы: https://github.com/13adams37/bsbd\_2021\_construction\_company

# **Вывод**

Во время выполнения курсового проекта были изучены методы работы с базами данных, способы управления. Таким образом, в ходе работы была разработана база данных, а также API, для автоматизации работы строительной компании.