

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«Крымский федеральный университет имени В.И. Вернадского»
Физико-технический институт
(наименование структурного подразделения (филиала))

ОТЧЕТ
по производственной, эксплуатационной практике

обучающегося Шенгелай Всеволода Михайловича

Физико-технический институт

Кафедра компьютерной инженерии и моделирования

Направление подготовки/специальность 09.03.04 «Программная инженерия»

Направленность (профиль)/специализация Высокоуровневое программирование

Курс 3

Группа ПИ-б-о-201(1)

Форма обучения очная

Руководитель практики от Университета _____,
(подпись) (должность, ФИО)

Отчет защищен с оценкой _____

ОГЛАВЛЕНИЕ

ОГЛАВЛЕНИЕ	2
ВВЕДЕНИЕ	3
ГЛАВА 1 Изучение синтаксиса SQL и работы с реляционными базами данных	4
1.1 Обзор выполнения Задания 1 «Основы языка SQL».	4
1.2 Обзор выполнения Задания 2 «Типы данных и операторы».....	5
1.3 Обзор выполнения Задания 3 «Агрегация данных и объединение таблиц».....	5
1.4 Обзор выполнения Задания 4 «Подзапросы в SQL».....	6
1.5. Обзор выполнения Задания 5 «Процедуры, триггеры, представления и оптимизация работы с данными».....	7
1.6. Обзор выполнения Задания 6 «Работа с MySQL».....	8
1.6. Практические задачи их решение.	9
ГЛАВА 2 Изучение синтаксиса языка PHP 7.1 и создание веб-приложения	14
2.1 Анализ инструментальных средств	14
2.1.1 Lavarel	14
2.1.2 Symfony	14
2.1.3 CodeIgniter	15
2.1.4 Yii	15
2.1.4 CakePHP	15
2.2. Составление ТЗ к веб-сайту.....	16
2.2.1. [BP] Описание основных сценариев работы.....	16
2.2.1.1 [BP01] Авторизация.....	16
2.2.1.2. [BP02] Аутентификация.....	17
2.2.1.3 [BP01] Регистрация	18
2.2.2 [FR] Моделирование структур хранения данных.....	19
2.3. Реализация веб-сайта.....	21
ЗАКЛЮЧЕНИЕ	27
ЛИТЕРАТУРА	28
ПРИЛОЖЕНИЕ 1	29

ВВЕДЕНИЕ

Целью данной производственной практики является приобретение знаний и навыков в области языка SQL, процедур, триггеров, представлений и оптимизации работы с данными. В рамках практики я также осваиваю работу с MySQL и разработку веб-приложений на языке PHP 7.1.

Для достижения поставленной цели, я выполняю ряд заданий, которые охватывают различные аспекты работы с данными и разработки веб-приложений. Задания включают изучение основ языка SQL, выполнение запросов, создание таблиц и определение связей, агрегацию данных, использование подзапросов, разработку процедур, триггеров и представлений, оптимизацию работы с данными, а также работу с MySQL чистым PHP.

В данном отчёте я представляю результаты выполнения каждого задания, описывая процесс и полученные результаты. Целью отчёта является документирование моего прогресса и оценка достигнутых результатов в каждой области. Важно отметить, что полученные знания и навыки будут иметь практическую применимость в моей будущей профессиональной деятельности.

ГЛАВА 1

Изучение синтаксиса SQL и работы с реляционными базами данных

1.1 Обзор выполнения Задания 1 «Основы языка SQL».

Изучение основных команд SQL:

- **SELECT:** изучение синтаксиса команды SELECT для выборки данных из таблицы, использование операторов выбора полей и таблиц, а также применение условий и операторов сравнения для фильтрации данных.
- **INSERT:** понимание синтаксиса команды INSERT для добавления новых записей в таблицу, указание значений для добавления.
- **UPDATE:** изучение синтаксиса команды UPDATE для обновления данных в таблице, применение условий для обновления определенных записей.
- **DELETE:** ознакомление с синтаксисом команды DELETE для удаления данных из таблицы, использование условий для удаления определенных записей.

Создание таблиц и определение связей:

- Изучение процесса создания таблиц в базе данных, указание имени таблицы, полей и их типов данных.
- Определение связей между таблицами с помощью первичных и внешних ключей.

Практическое применение команд SQL:

- Выполнение запросов SELECT для выборки данных из таблицы, использование условий, операторов сравнения и логических операторов для фильтрации результатов.
- Использование команды INSERT для добавления новых записей в таблицу, указание значений и соответствующих полей.
- Применение команды UPDATE для обновления данных в таблице, использование условий для обновления только определенных записей.
- Удаление данных с помощью команды DELETE, применение условий для удаления определенных записей.

1.2 Обзор выполнения Задания 2 «Типы данных и операторы».

Изучение различных типов данных в SQL:

- Освоение числовых типов данных, таких как целые числа, десятичные числа и числа с плавающей запятой.
- Понимание строковых типов данных, включая символьные строки и текст.
- Ознакомление с логическим типом данных и типом даты/времени.

Изучение операторов выбора данных:

- Понимание команды SELECT и ее использование для выборки данных из таблиц.
- Изучение операторов фильтрации, таких как WHERE, IN, BETWEEN, LIKE и других, для ограничения выборки по условиям.
- Освоение оператора ORDER BY для сортировки результатов выборки.

Практическое применение типов данных и операторов:

- Создание таблиц с различными типами данных и заполнение их данными.
- Выполнение запросов SELECT с использованием операторов фильтрации и сортировки.
- Анализ результатов выборки и сравнение с ожидаемыми выводами.

1.3 Обзор выполнения Задания 3 «Агрегация данных и объединение таблиц».

Изучение агрегатных функций:

- Понимание понятия агрегации данных и ее роли в SQL.
- Ознакомление с различными агрегатными функциями, такими как SUM, AVG, COUNT, MIN, MAX и др.
- Изучение синтаксиса и применение агрегатных функций для расчета сумм, средних значений, количества и других статистических данных.

Изучение операторов объединения таблиц:

- Понимание концепции объединения таблиц и его применения в SQL.
- Изучение операторов JOIN, INNER JOIN, LEFT JOIN, RIGHT JOIN и других для объединения данных из разных таблиц.
- Понимание использования условий объединения и выбора нужных столбцов при объединении таблиц.

Практическое применение агрегации данных и объединения таблиц:

- Выполнение запросов с использованием агрегатных функций для расчета статистических данных.
- Объединение таблиц для получения связанных данных из разных источников.
- Анализ результатов агрегации и объединения и сравнение с ожидаемыми выводами.

1.4 Обзор выполнения Задания 4 «Подзапросы в SQL».

Изучение понятия подзапросов:

- Понимание концепции подзапросов и их роли в SQL.
- Ознакомление с различными типами подзапросов, такими как скалярные, списковые и коррелированные подзапросы.
- Понимание использования подзапросов для получения сложных результатов и выполнения вложенных операций.

Изучение синтаксиса и применение подзапросов:

- Освоение синтаксиса создания подзапросов в SQL.
- Понимание использования операторов сравнения, логических операторов и агрегатных функций в подзапросах.
- Изучение примеров применения подзапросов для выполнения различных задач.

Практическое применение подзапросов:

- Создание запросов с использованием подзапросов для фильтрации данных.

- Использование подзапросов для создания временных таблиц и промежуточных результатов.
- Анализ результатов и сравнение с ожидаемыми выводами.

1.5. Обзор выполнения Задания 5 «Процедуры, триггеры, представления и оптимизация работы с данными».

Изучение процедур:

- Понимание понятия процедуры и ее роли в SQL.
- Ознакомление с созданием и вызовом процедур.
- Изучение передачи параметров в процедуры и обработки результата.

Изучение триггеров:

- Понимание понятия триггера и его применения в SQL.
- Ознакомление с созданием триггеров для автоматического выполнения определенных действий при изменении данных в базе данных.

Изучение представлений:

- Понимание понятия представления и его применения в SQL.
- Освоение создания и использования представлений для упрощения запросов к данным и создания виртуальных таблиц.

Практическое применение процедур, триггеров, представлений и оптимизации работы с данными:

- Разработка процедур для автоматизации повторяющихся задач.
- Создание триггеров для автоматической обработки данных при определенных событиях.
- Использование представлений для упрощения сложных запросов.
- Оптимизация работы с данными для повышения производительности и эффективности запросов.

1.6. Обзор выполнения Задания 6 «Работа с MySQL».

Изучение особенностей MySQL:

- Ознакомление с основными характеристиками и возможностями MySQL.
- Понимание различий между MySQL и другими системами управления базами данных.
- Изучение основных команд и операторов, уникальных для MySQL.

Использование MySQL в контексте SQL:

- Применение изученных ранее концепций и навыков SQL в работе с MySQL.
- Создание таблиц, выполнение запросов и модификация данных в MySQL.
- Ознакомление с функциями и процедурами, специфичными для MySQL.

Выполнение задания, связанного с MySQL:

- Постановка задачи или задания, связанного с использованием MySQL.
- Проектирование и создание структуры таблиц и связей для выполнения задания.
- Написание и выполнение запросов, процедур или других элементов, необходимых для решения задачи.

1.6. Практические задачи их решение.

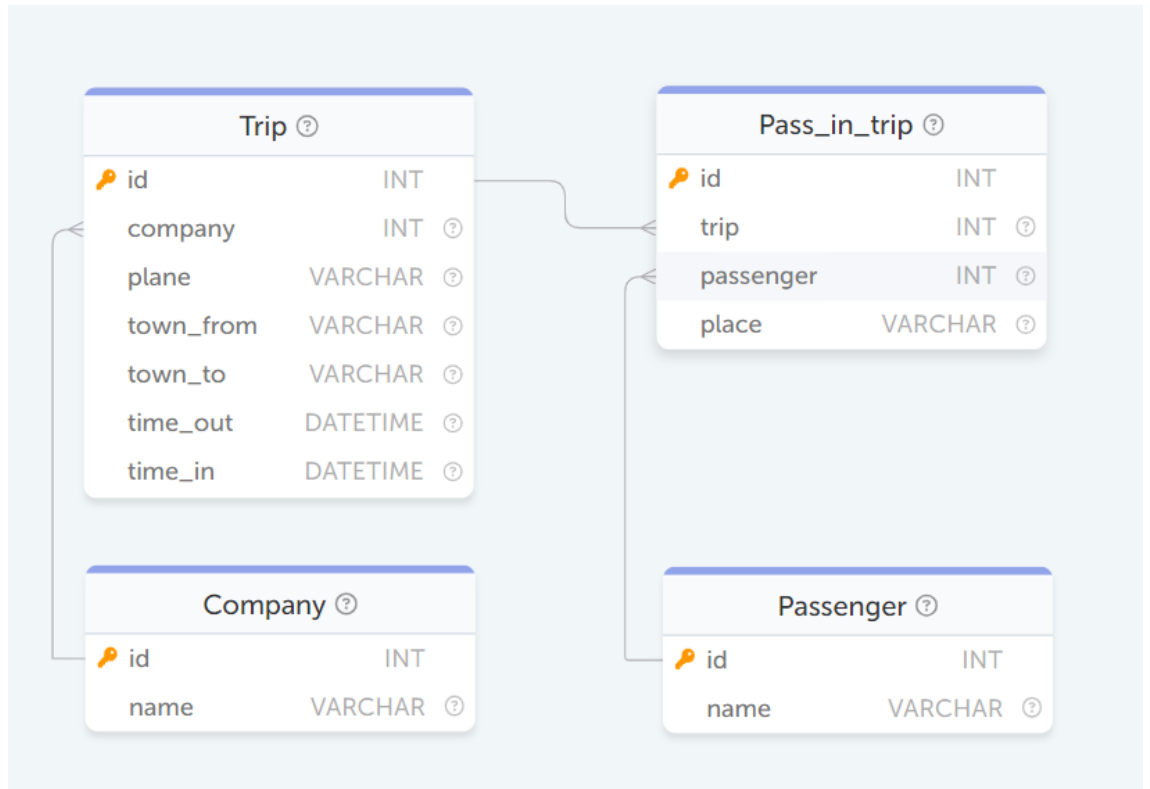


Рисунок 1. Структура базы данных, в которой будут выполняться все практические задачи.

Задача 1, средняя сложность. В какие города можно улететь из Парижа (Paris) и сколько времени это займёт?

Поля в результирующей таблице: town_to, flight_time. Используйте конструкцию "as flight_time" для вывода необходимого времени. Это необходимо для корректной проверки. Формат для вывода времени: HH:MM:SS

```

1 SELECT DISTINCT town_to,
2     SEC_TO_TIME(TIME_TO_SEC(time_in) - TIME_TO_SEC(time_out)) AS
3     flight_time
4 FROM Trip
5 WHERE town_from = 'Paris';
  
```

Рисунок 2. Код SQL-запроса

Результат запроса		Показать таблицу ▼
	town_to	flight_time
1	Rostov	03:33:00
2	London	01:00:00

Рисунок 3. Результат запроса

Задача 2, средняя сложность. Вывести вылеты, совершенные с 10 ч. по 14 ч. 1 января 1900 г.

Поля в результирующей таблице: *.

```
1 SELECT *
2 FROM Trip
3 WHERE time_out >= '1900-01-01 10:00:00' AND time_out <= '1900-01-01 14:00:00'
   ;
```

Рисунок 4. Код SQL-запроса

Результат запроса							Показать таблицу ▼
	id	company	plane	town_from	town_to	time_out	time_in
1	1182	1	TU-134	Moscow	Rostov	1900-01-01T12:35:00.000Z	1900-01-01T14:30:00.000Z
2	7772	5	Boeing	Singapore	London	1900-01-01T12:00:00.000Z	1900-01-02T02:00:00.000Z
3	7774	5	Boeing	Singapore	London	1900-01-01T14:00:00.000Z	1900-01-02T06:00:00.000Z

Рисунок 5. Результат запроса

Задача 3, средняя сложность. Вывести пассажиров с самым длинным именем.

Поля в результирующей таблице: name.

```
SELECT name
FROM Passenger
WHERE LENGTH(name) = (
    SELECT MAX(LENGTH(name))
    FROM Passenger
);
```

Рисунок 6. Код SQL-запроса

Результат запроса		Показать таблицу ▼
	name	
1	Catherine Zeta-Jones	

Рисунок 7. Результат запроса

Задача 4, средняя сложность. Вывести имена людей, у которых есть полный тёзка среди пассажиров.

Поля в результирующей таблице: name.

```
SELECT name
FROM Passenger
WHERE LENGTH(name) = (
    SELECT MAX(LENGTH(name))
    FROM Passenger
);
```

Рисунок 6. Код SQL-запроса

Результат запроса		Показать таблицу ▼	
	name		
1	Catherine Zeta-Jones		

Рисунок 7. Результат запроса

Задача 5, средняя сложность. Вывести имена людей, у которых есть полный тёзка среди пассажиров.

Поля в результирующей таблице: name.

```
1 SELECT DISTINCT p1.name
2 FROM Passenger p1
3 INNER JOIN Passenger p2 ON SUBSTRING_INDEX(p1.name, ' ', -1)
   SUBSTRING_INDEX(p2.name, ' ', -1)
4 AND p1.id <> p2.id;
```

Рисунок 8. Код SQL-запроса

Результат запроса		Показать таблицу ▼	
	name		
1	Bruce Willis		

Рисунок 9. Результат запроса

Задача 6, средняя сложность. Вывести отсортированный по количеству перелетов (по убыванию) и имени (по возрастанию) список пассажиров,

совершивших хотя бы 1 полет.

Поля в результирующей таблице: name, count. Используйте конструкцию "as count" для агрегатной функции подсчета количества перелетов. Это необходимо для корректной проверки.

```
SELECT p.name, COUNT(*) as count
FROM Passenger p
JOIN Pass_in_trip pt ON p.id = pt.passenger
GROUP BY p.name
HAVING count > 0
ORDER BY count DESC, p.name ASC;
```

Рисунок 10. Код SQL-запроса

Результат запроса		Показать таблицу ▼
	name	count
1	Michael Caine	4
2	Mullah Omar	4
3	Bruce Willis	3
4	Harrison Ford	3
5	Jennifer Lopez	3
6	Kurt Russell	3
7	Nikole Kidman	3

Рисунок 11. Результат запроса

Задача 7, высокая сложность. Удалить компании, совершившие наименьшее количество рейсов.

```
SELECT p.name, COUNT(*) as count
FROM Passenger p
JOIN Pass_in_trip pt ON p.id = pt.passenger
GROUP BY p.name
HAVING count > 0
ORDER BY count DESC, p.name ASC;
```

Рисунок 12. Код SQL-запроса

Результат запроса		Показать таблицу ▼	
	id	name	
1	1	Don_avia	
2	5	British_AW	

Рисунок 13. Результат запроса

ГЛАВА 2

Изучение синтаксиса языка PHP 7.1 и создание веб-приложения

2.1 Анализ инструментальных средств

Проанализируем популярные PHP фреймворки и составим их описание, а также опишем задачи, для которых применяются эти фреймворки и также самые полезные инструменты, содержащиеся в них.

2.1.1 Laravel

Описание: это мощный фреймворк с простым синтаксисом и широким набором инструментов. Он подходит для разработки веб-приложений любого масштаба. Laravel предлагает удобную работу с маршрутизацией, базами данных, аутентификацией, шаблонизацией, валидацией и многими другими аспектами разработки.

Инструменты: миграции баз данных, встроенная система аутентификации и авторизации, eloquent ORM (Object-Relational Mapping), Blade шаблонизатор, маршрутизация, очереди задач и планировщик задач, кэширование, формы и валидация данных, генерация API, отладка и профилирование.

2.1.2 Symfony

Описание: это гибкий фреймворк, ориентированный на создание сложных веб-приложений. Он предлагает мощные инструменты для работы с маршрутизацией, формами, базами данных и другими важными компонентами. Symfony также обеспечивает высокую степень настраиваемости и поддержку стандартов разработки.

Инструменты: компоненты для обработки HTTP-запросов, маршрутизация, формы и валидация данных. Doctrine ORM (Object-Relational Mapping), шаблонизатор Twig, система событий и подписчиков, кэширование, консольные команды, безопасность и авторизация, тестирование.

2.1.3 CodeIgniter

Описание: это легковесный фреймворк, который идеально подходит для быстрой разработки веб-приложений. Он предоставляет простой и интуитивно понятный интерфейс, а также широкий спектр инструментов, включая работу с базами данных, формами, сессиями и другими основными компонентами. Инструменты: встроенные библиотеки для работы с базой данных, формы и валидация данных, сессии и куки, загрузка файлов, роутинг, отладка и профилирование, кэширование, шифрование и безопасность, шаблонизаторы.

2.1.4 Yii

Описание: это высокопроизводительный фреймворк с широким набором возможностей. Он предлагает инструменты для работы с базами данных, аутентификацией, кэшированием, миграциями и другими функциональными возможностями. Yii позволяет разрабатывать как веб-приложения, так и веб-сервисы.

Инструменты: миграции баз данных, Active Record для работы с базой данных, встроенная система аутентификации и авторизации, кэширование, шаблонизаторы, генерация кода (Gii), механизм событий, валидация данных, RESTful API, работа с веб-сервисами.

2.1.4 CakePHP

Описание: это простой и эффективный фреймворк, спроектированный для разработки веб-приложений. Он предоставляет инструменты для работы с базами данных, ORM, аутентификацией, маршрутизацией и другими ключевыми компонентами. CakePHP позволяет быстро создавать функциональные и масштабируемые веб-приложения.

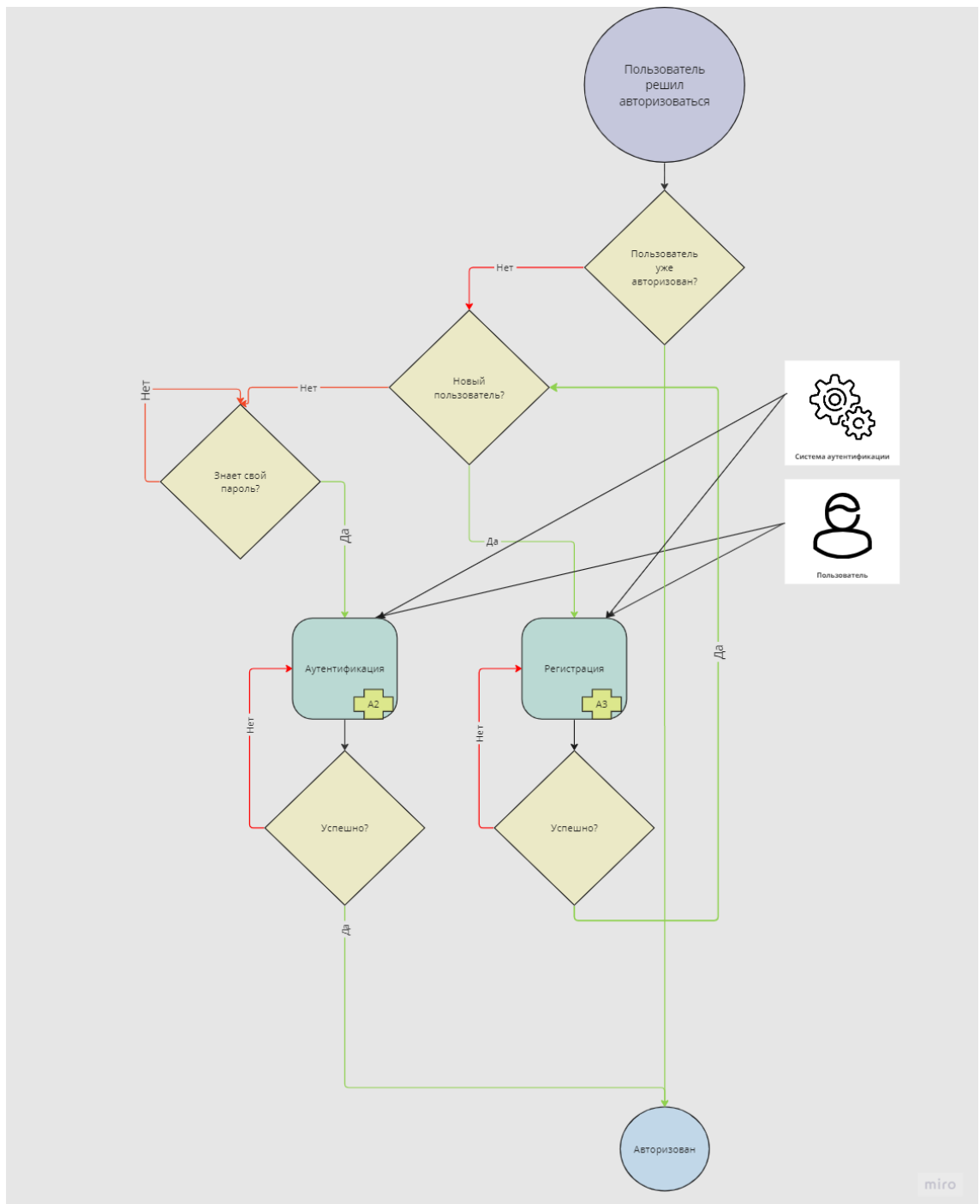
Инструменты: миграции баз данных, ORM (Object-Relational

Mapping), встроенная система аутентификации и авторизации, маршрутизация, генерация кода (Bake), валидация данных, кэширование, шаблонизаторы, работа с событиями, тестирование.

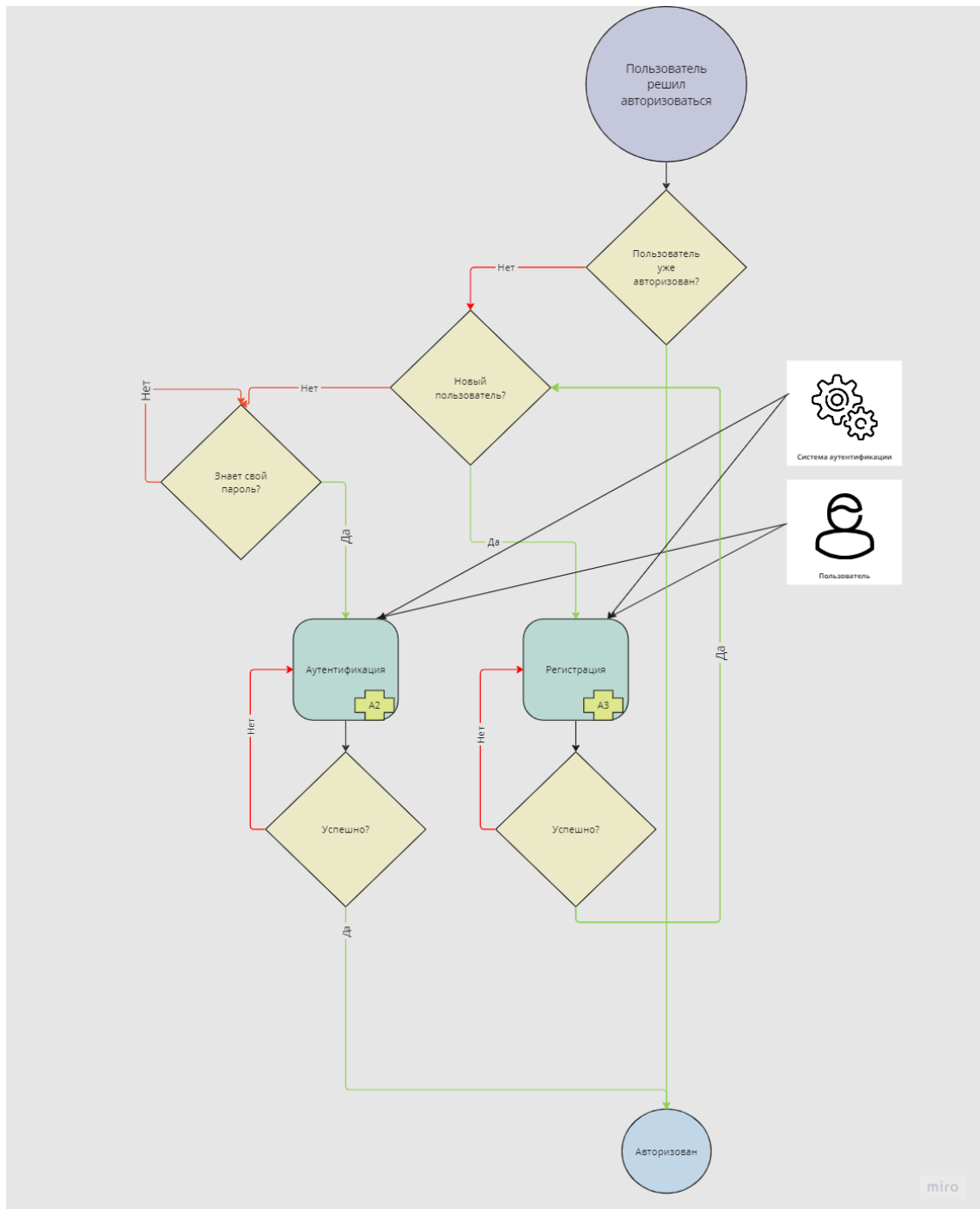
2.2. Составление ТЗ к веб-сайту

2.2.1. [BP] Описание основных сценариев работы

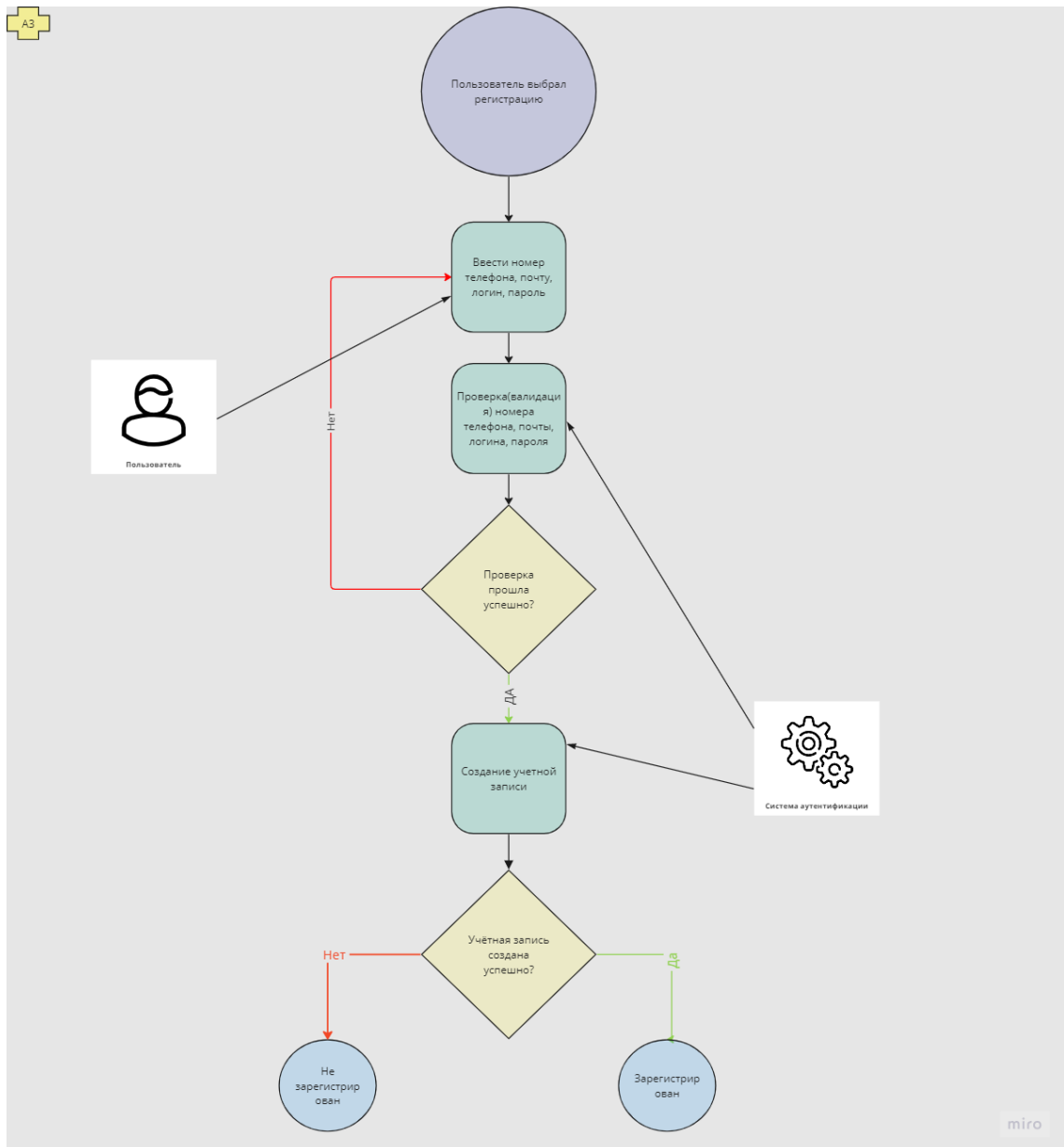
2.2.1.1 [BP01] Авторизация



2.2.1.2. [BP02] Аутентификация



2.2.1.3 [BP01] Регистрация



Сценарии работы в текстовом виде:

1. В форме регистрации пользователь должен указать имя(логин), телефон, почту, пароль и повторить ввод пароль.
2. Почта, логин и телефон должны быть уникальны, и, если такие в базе уже есть следует уведомить пользователя об этом.
3. Пароли в обоих полях должны совпадать, иначе уведомить пользователя об этом.
4. Авторизация быть возможно возможна по телефону или email (в

одном поле) и паролю, необходимо добавить Google reCAPTCHA при авторизации.

5. Сделать страницу, к которой только авторизованные пользователи имеют доступ. На этой странице пользователи могут менять свою личную информацию (имя, телефон, почта, пароль). Неавторизованные пользователи должны перенаправляться на главную страницу.

2.2.2 [FR] Моделирование структур хранения данных

У нас будет две таблицы: **"users"** и **"old_users"**.

Таблица **"users"** предназначена для хранения текущей информации о пользователях, таких как их логины, электронные адреса, номера телефонов и захешированные пароли. Эта таблица служит основным хранилищем данных о пользователях и используется для аутентификации, авторизации и управления доступом на сайте.

Таблица **"old_users"** используется для сохранения старых значений полей из таблицы **"users"** при их изменении. Триггер, связанный с таблицей **"users"**, отслеживает изменения и записывает старые значения полей в таблицу **"old_users"**. Это позволяет сохранить историю изменений данных пользователей и может быть полезно для аудита, отладки и восстановления данных.

Таблицы **"users"** и **"old_users"** могут быть связаны по значению поля **"id"**. Предполагается, что поле **"id"** в таблице **"users"** является первичным ключом, уникально идентифицирующим каждую запись о пользователе. В свою очередь, поле **"record_id"** может быть использовано в таблице **"old_users"** как внешний ключ для связи с соответствующим пользователем из таблицы **"users"**.

Также связь между таблицами **"users"** и **"old_users"** устанавливается через триггер, который реагирует на изменения в таблице **"users"**.

Уникального идентификатора записи в виде автоинкрементируемого

поля в таблице "old_users" не будет. У нас есть поле timestamp, которое будет делать каждую запись уникальным.

Таблица 1. Описание полей таблицы "users" и содержащейся в них информации

Поле	Описание
Id	уникальный идентификатор пользователя.
Login	логин пользователя.
Email	электронная почта пользователя.
telephone	номер телефона пользователя.
password	захешированный пароль пользователя.

Таблица 2. Описание полей таблицы " old_users" и содержащейся в них информации

Поле	Описание
record_id	ключ, связывающий измененные данные в таблице "users" с соответствующими записями в таблице "old_users"
Login	сохраненное старое значение поля "login" из таблицы "users".
Email	сохраненное старое значение поля "email" из таблицы "users".
telephone	сохраненное старое значение поля "telephone" из таблицы "users".
password:	сохраненное старое значение поля "password" из таблицы "users".
timestamp:	временная метка, указывающая на момент изменения в таблице "users".

2.3. Реализация веб-сайта

Реализацию сайта начнём с создания таблиц в базе данных и связей между ними. Таблицы создаём в MySQL через графический интерфейс.

Выполним SQL-запрос к базе данных, где создадим триггер `save_old_data`.

Этот триггер предназначен для сохранения старой версии данных пользователей перед их обновлением в таблице `"users"`. Когда происходит обновление записи в этой таблице, триггер срабатывает автоматически перед выполнением обновления и выполняет следующие действия.

Он создает новую запись в таблице `"old_users"`, куда сохраняются старые данные пользователя. В эту новую запись копируются значения из полей `"id"`, `"login"`, `"email"`, `"telephone"` и `"password"` старой версии данных, которые сейчас будут обновлены. Поле `"timestamp"` заполняется текущим временем, чтобы отразить момент сохранения старых данных.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
<input type="checkbox"/> 1	id	int			Нет	<i>Нет</i>		AUTO_INCREMENT	Ещё ▾
<input type="checkbox"/> 2	login	varchar(64)	<i>utf8mb4_0900_ai_ci</i>		Да	NULL			Ещё ▾
<input type="checkbox"/> 3	email	varchar(48)	<i>utf8mb4_0900_ai_ci</i>		Да	NULL			Ещё ▾
<input type="checkbox"/> 4	telephone	varchar(48)	<i>utf8mb4_0900_ai_ci</i>		Да	NULL			Ещё ▾
<input type="checkbox"/> 5	password	varchar(255)	<i>utf8mb4_0900_ai_ci</i>		Да	NULL			Ещё ▾

Рисунок 14. Общая информация о таблице `"users"`

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
<input type="checkbox"/> 1	record_id	int			Да	NULL			Ещё ▾
<input type="checkbox"/> 2	login	varchar(255)	<i>utf8mb4_0900_ai_ci</i>		Да	NULL			Ещё ▾
<input type="checkbox"/> 3	email	varchar(255)	<i>utf8mb4_0900_ai_ci</i>		Да	NULL			Ещё ▾
<input type="checkbox"/> 4	telephone	varchar(255)	<i>utf8mb4_0900_ai_ci</i>		Да	NULL			Ещё ▾
<input type="checkbox"/> 5	password	varchar(255)	<i>utf8mb4_0900_ai_ci</i>		Да	NULL			Ещё ▾
<input type="checkbox"/> 6	timestamp	timestamp			Да	NULL			Ещё ▾

Рисунок 15. Общая информация о таблице "old_users"

Детали

Название триггера:

Таблица:

Время:

Событие:

Определение

```

1 BEGIN
2   INSERT INTO old_users (record_id, login,
3     email, telephone, password, timestamp)
4     VALUES (OLD.id, OLD.login, OLD.email,
5       OLD.telephone, OLD.password,
6       CURRENT_TIMESTAMP);
7 END

```

Определитель:

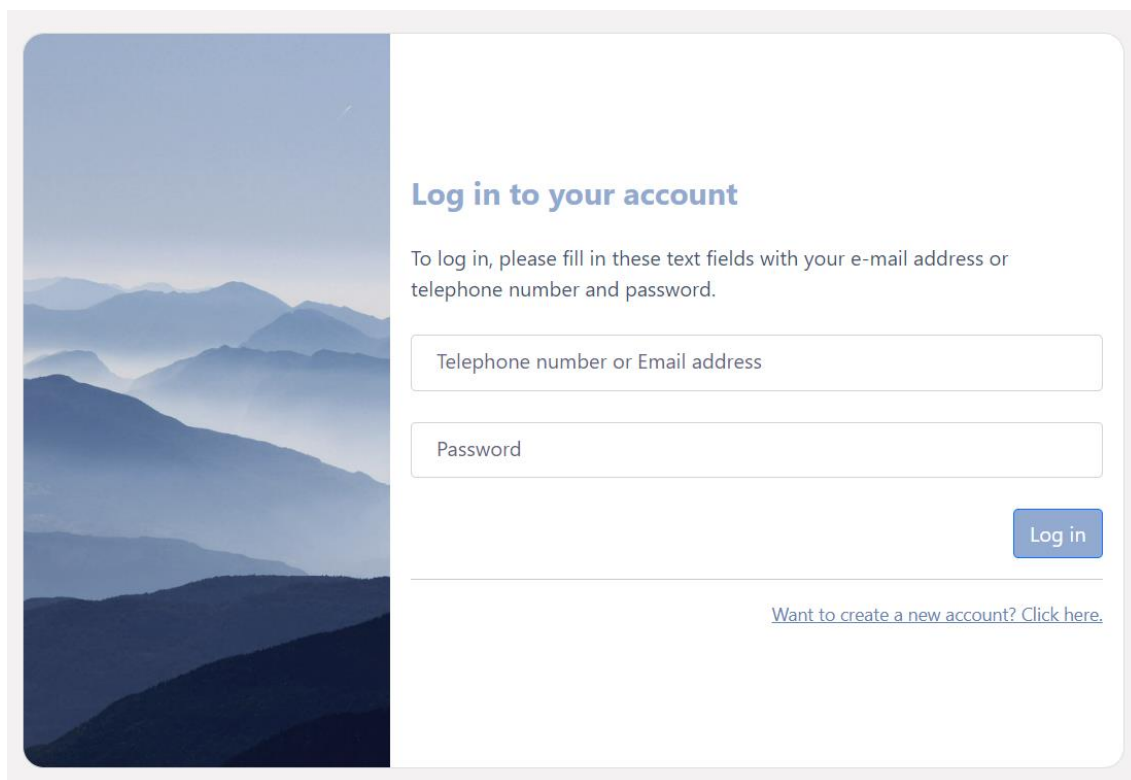
Вперёд Заккрыть

Рисунок 16. Код триггера "save_old_data"

Далее приступим к вёрстке страниц. Для достижения гармоничного и отзывчивого дизайна я решил использовать Bootstrap, так как он предоставляет широкий набор готовых компонентов и стилей, а также обеспечивает адаптивность для различных устройств и размеров экранов.

Я создал все страницы, не пользуясь специализированным ПО для дизайна и прототипирования веб-сайтов вроде Bootstrap Studio. Я использовал расширение “LiveServer” в VSCode, чтобы быстро просматривать изменения вёрстки.

Также я настроил стили Bootstrap, чтобы они соответствовали общему визуальному стилю и брендингу моего сайта. Я изменил цвета, шрифты и другие элементы дизайна, чтобы создать согласованный пользовательский интерфейс.

The image shows a login form on a webpage. On the left is a vertical image of misty mountains. The form is on the right, titled "Log in to your account". It includes instructions to fill in email or phone number and password. There are two input fields: "Telephone number or Email address" and "Password". A blue "Log in" button is at the bottom right. A link "Want to create a new account? Click here." is at the bottom.

Log in to your account

To log in, please fill in these text fields with your e-mail address or telephone number and password.

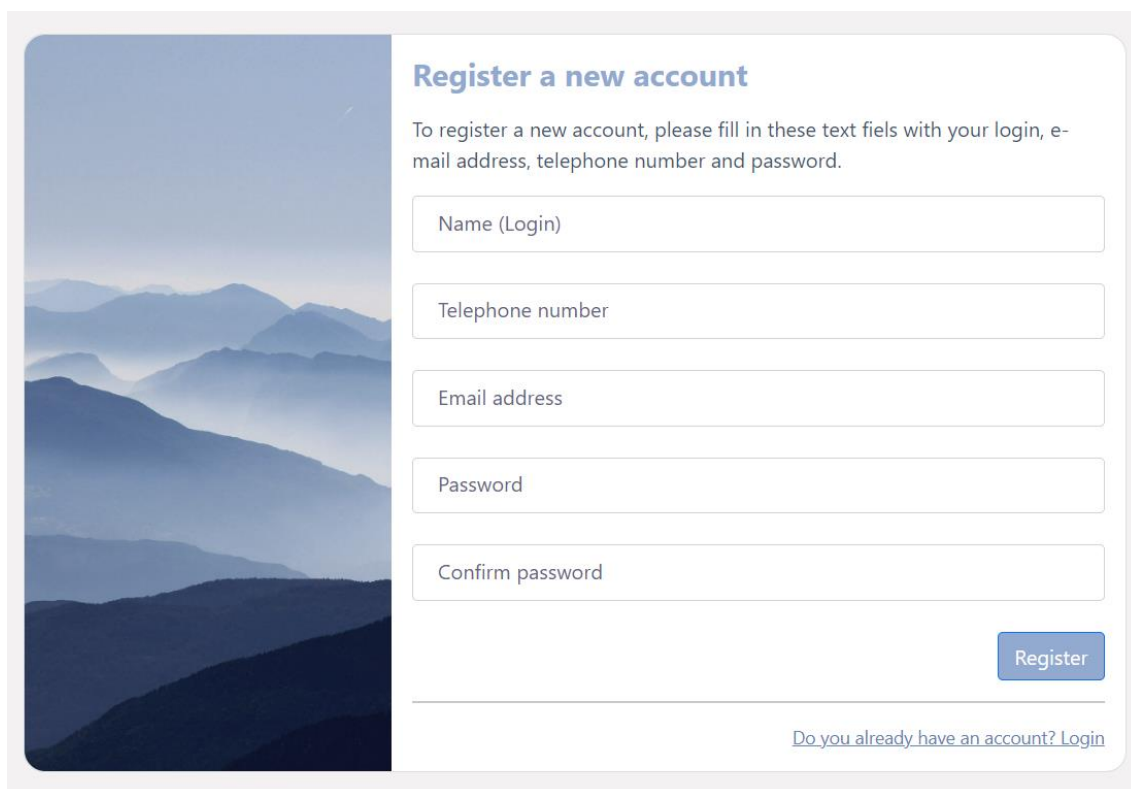
Telephone number or Email address

Password

Log in

[Want to create a new account? Click here.](#)

Рисунок 17. Страница index.php. Форма авторизации

The image shows a registration form on a webpage. On the left is a vertical image of misty mountains. The form is on the right, titled "Register a new account". It includes instructions to fill in login, email, phone number, and password. There are five input fields: "Name (Login)", "Telephone number", "Email address", "Password", and "Confirm password". A blue "Register" button is at the bottom right. A link "Do you already have an account? Login" is at the bottom.

Register a new account

To register a new account, please fill in these text fiels with your login, e-mail address, telephone number and password.

Name (Login)

Telephone number

Email address

Password

Confirm password

Register

[Do you already have an account? Login](#)

Рисунок 18. Страница registration.php. Форма регистрации

Profile Settings

Name(Login)
Всеволод Ростовский

Telephone number
79787988045

Email address
vsevolod.sheng@mail.ru

Old password

New password

Confirm new password

[Посмотреть историю изменений](#)

[Save Profile](#) [Exit](#)

Рисунок 19. Страница profile.php. Профиль пользователя

Data List

Record_ID: 15
Login: Всеволод Ростовский
Email: vsevolod.sheng@mail.ru
Telephone: +7978798802
Timestamp: 2023-06-15 10:21:48

Record_ID: 15
Login: Всеволод Ростовский
Email: vsevolod.sheng@mail.ru
Telephone: +79787988044
Timestamp: 2023-06-15 10:22:02

Record_ID: 15
Login: Всеволод Ростовский
Email: vsevolod.sheng@mail.ru
Telephone: +79787988045
Timestamp: 2023-06-15 11:37:53

[Назад к профилю](#)

[Take data changes](#)

Рисунок 20. Страница data.php. Страница со списком изменений данных аккаунта

Опишем структуру нашего сайта, где файлы в папке "server" отвечают за обработку данных и бизнес-логику, а файлы в корне проекта отвечают за отображение веб-страниц пользователю.

1 Папка "assets":

- Изображения: здесь хранятся все изображения, которые используются на вашем сайте (у нас это аватары пользователей и фоновые изображения).
- CSS стили: В этой папке находятся файлы со стилями CSS, которые определяют внешний вид и макет наших веб-страниц.
- Код JavaScript: содержит файлы JavaScript, которые могут содержать клиентскую логику и добавлять интерактивность на наш сайт (эта папка пуста, создана для будущего AJAX)

2 Папка "server":

- "connect.php": файл отвечает за установку соединения с базой данных. В нем мы устанавливаем параметры подключения, такие как хост, имя пользователя, пароль и имя базы данных. Затем он создает переменную "link", через которую можно обратиться к базе данных, и она может быть использована в других файлах, где требуется работа с базой данных. Этот файл подключается (require_once) в других файлах, где требуется работа с базой данных.
- "data_handler.php": файл, который обрабатывает страницу "data.php". Он проверяет базу данных на наличие истории изменений полей аккаунта пользователя и передает эту информацию на страницу "data.php".
- "logout.php": в этом файле реализована логика выхода пользователя из аккаунта. Он удаляет глобальную переменную \$_SESSION, которая содержит информацию о текущей сессии пользователя, и тем самым завершает сеанс авторизации.
- "functions.php": файл, который содержит функции для проверки уникальности логина, почты и номера телефона при регистрации пользователей. Также он содержит функции для валидации логина, почты и телефона с использованием регулярных выражений.
- "profile_handler.php": файл, который содержит логику изменения

личной информации пользователей, таких как имя, пароль, почта и телефон. Он обрабатывает данные, полученные из формы профиля пользователя, выполняет необходимые проверки и обновляет соответствующие поля в базе данных.

- "signin.php": файл, который содержит логику авторизации пользователя. Здесь также происходит подключение к сервису Google reCAPTCHA для проверки пользователя.
- "signup.php": файл обрабатывает данные, полученные из формы регистрации, выполняет необходимые проверки (на уникальность логина, электронной почты, номера телефона) и создает новую запись пользователя в базе данных.

3 В корне проекта:

- "data.php": страница, которая отображает историю изменений личной информации аккаунта. Использует данные, полученные из "data_handler.php".
- "profile.php": страница профиля пользователя. Здесь отображается информация о пользователе: имя(логин), фотография профиля, номер телефона и email. Пользователь также может взаимодействовать с этой страницей для изменения своих данных.
- "registration.php": страница с формой регистрации пользователя. Здесь пользователь вводит необходимые данные для создания учетной записи, такие как логин, пароль, адрес электронной почты и телефон. Форма отправляет данные на "signup.php" для обработки.
- "index.php": главная страница, которая содержит форму входа пользователя. Данные отправляются на "signin.php" для проверки и выполнения авторизации.

Все файлы с кодом находятся в Приложении1

ЗАКЛЮЧЕНИЕ

В ходе производственной практики я успешно освоил основы языка SQL и работу с реляционными базами данных. Выполнил ряд заданий, включающих выполнение запросов, создание таблиц, агрегацию данных, использование подзапросов, разработку процедур, триггеров и представлений, а также оптимизацию работы с данными. Благодаря этому, я приобрел необходимые навыки для эффективной работы с базами данных и выполнения разнообразных задач, связанных с их обработкой.

Кроме того, я изучил синтаксис языка PHP 7.1 и освоил создание веб-приложений. Анализировал различные инструменты разработки, такие как Laravel, Symfony, CodeIgniter, Yii и CakePHP, их особенности и возможности. Составил техническое задание для создания веб-сайта, описал основные сценарии работы и моделировал структуры хранения данных.

В результате практики я успешно реализовал веб-сайт, соответствующий поставленным требованиям. Полученный опыт и навыки в разработке веб-приложений на языке PHP будут полезны в моей будущей профессиональной деятельности.

Прохождение данной практики позволило мне углубить знания в области работы с базами данных и веб-разработки, а также развить навыки анализа, проектирования и реализации программных решений. Я уверен, что полученные знания и опыт я смогу применить на практике и достичь успеха в своей карьере в области разработки программного обеспечения.

ЛИТЕРАТУРА

1. Иванов, Алексей. Основы SQL: Подробное руководство. Москва: Издательский дом "Эксмо", 2022.
2. Петрова, Елена. "Продвинутые техники SQL". Журнал "Управление базами данных", 2021, Т. 32, № 3, С. 45-62.
3. Смирнов, Дмитрий. "Введение в реляционные базы данных". В Материалы международной конференции по системам управления базами данных, Москва: Издательство "НТЦ "Инфра-М", 2023, С. 78-92.
4. PHP Documentation. Доступно по: <https://www.php.net/docs.php>. Дата обращения: 10 мая 2023.

ПРИЛОЖЕНИЕ 1

Код файла functions.php

```
1.<?php
2.
3.require_once 'connect.php';
4.
5.function clear_data($val){
6.    //Удаляет пробелы (или другие символы) из начала и конца строки
7.    $val = trim($val);
8.    //Удаляет экранирование символов
9.    $val = stripslashes($val);
10.    //Удаляет теги HTML и PHP из строки
11.    $val = strip_tags($val);
12.    //Преобразует специальные символы в HTML-сущности
13.    $val = stripslashes($val);
14.    return $val;
15.}
16.
17./*
18.Набор запросов к БД для дальнейшей проверки уникальности логина, почты,
    телефона
19.*/
20.// Возвращает false - не уникален, возвращает true - уникален
21.function is_login_unique($login, $link){
22.    $query_login = "SELECT COUNT(*) FROM `users` WHERE `login` =
        '$login'";
23.    $result_login = mysqli_query($link, $query_login);
24.    $count_login = mysqli_fetch_array($result_login)[0];
25.    if ($count_login > 0){
26.        return false;
27.    }
28.    else{
29.        return true;
30.    }
31.}
32.function is_telephone_unique($telephone, $link){
33.    $query_telephone = "SELECT COUNT(*) FROM `users` WHERE `telephone` =
        '$telephone'";
34.    $result_telephone = mysqli_query($link, $query_telephone);
35.    $count_telephone = mysqli_fetch_array($result_telephone)[0];
36.    if ($count_telephone > 0){
37.        return false;
38.    }
39.    else{
40.        return true;
41.    }
42.}
43.function is_email_unique($email, $link){
44.    $query_email = "SELECT COUNT(*) FROM `users` WHERE `email` =
        '$email'";
```

```

45.     $result_email = mysqli_query($link, $query_email);
46.     $count_email = mysqli_fetch_array($result_email)[0];
47.     if ($count_email > 0){
48.         return false;
49.     }
50.     else{
51.         return true;
52.     }
53.}
54.
55.function is_valid_login($login){
56.    /*
57.    Формат логина
58.    1) Строка начинается с любой буквы в любом алфавите (Unicode)
59.    2) Затем содержит от 4 до 31 символов любого типа (включая буквы,
        цифры, пробелы и специальные символы)
60.    */
61.    $pattern_login = '/^[^W\d_][\p{L}\d\s\p{P}]{2,29}\p{L}$/u';
62.    return preg_match($pattern_login, $login);
63.}
64.function is_valid_telephone($telephone){
65.    /*
66.    Международный формат номера
67.    1) В начале могут быть пробелы, после них может быть "+" (а может и
        не быть)
68.    2) Далее должна идти группа цифр в количестве от 10 до 14.
69.    3) До и после каждой цифры может быть один из 8 знаков ("-", " ",
        "_", "(", ")", ":", "=", "+")
70.    */
71.    $pattern_phone = '/^(\\s*)?(\\+)?([-_():=+]?\\d[-_():=+]?)\\{10,14\\}(\\s*)?$/';
72.    return preg_match($pattern_phone, $telephone);
73.}
74.function is_valid_email($email){
75.    /*
76.    Формат e-mail адреса
77.    */
78.    $pattern_email = '/^[A-Z0-9._%+-]+@[A-Z0-9-]+.+. [A-Z]{2,4}$/i';
79.    return preg_match($pattern_email, $email);
80.}

```

Код файла signin.php

```

1.<?php
2.    session_start();
3.    require_once 'connect.php';
4.
5.    $captcha = $_POST['token'];
6.    $secretKey = "6Lf-rikmAAAAALnc4MzeL6yifWVGUSgfbZFGJx";
7.
8.    $url = 'URL: https://www.google.com/recaptcha/api/siteverify?secret='
        . \$secretKey . '&response=' . \$captcha;
9.

```

```

10.     $response = file_get_contents($url);
11.     $responseKeys = json_decode($response, true);
12.     header('Content-type: application/json');
13.
14.     if(!($responseKeys["success"] && $responseKeys["score"] >= 0.5)){
15.         //die('Вы робот');
16.     }
17.
18.     $telephone_or_email = $_POST['telephone_or_email'];
19.     $password = $_POST['password'];
20.     $hash = password_hash($password, PASSWORD_DEFAULT);
21.
22.     //password_verify(string $password, string $hash);
23.
24.     //Найдём запись о пользователе по телефону или почте
25.     //Запись должна быть одна
26.     $result= mysqli_query($link, "SELECT * FROM `users`
27.     WHERE (`telephone` = '$telephone_or_email' OR `email` =
28.     '$telephone_or_email')");
29.     //Преобразуем результат запроса в ассоциативный массив (словарь)
30.     $user = mysqli_fetch_assoc($result);
31.
32.     //Получим захешированный пароль, хранящийся в БД
33.     $userpassword = $user['password'];
34.
35.     //Если пароль правильный
36.     if(password_verify($password, $userpassword) == 1){
37.         print('Всё гуд');
38.
39.         $_SESSION['user'] = [
40.             'id' => $user['id'],
41.             'name' => $user['login'],
42.             'email' => $user['email'],
43.             'telephone' => $user['telephone'],
44.         ];
45.
46.         header('Location: ../profile.php');
47.     }
48.     else{
49.         $_SESSION['message'] = 'Не верный логин или пароль';
50.         header('Location: ../index.php');
51.     }
52.
53.     //print($userpassword);
54.
55.     //echo mysqli_num_rows($check_user);
56. ?>

```

Код файла signup.php

```

1.<?php
2.     session_start();

```

```

3.     require_once 'connect.php';
4.     require_once 'functions.php';
5.
6.     $name = clear_data($_POST['name']);
7.     $telephone = clear_data($_POST['telephone']);
8.     $email = clear_data($_POST['email']);
9.     $password = clear_data($_POST['password']);
10.    $confirm_password = clear_data($_POST['confirm_password']);
11.
12.
13.    $pattern_email = '/^[A-Z0-9._%+-]+@[A-Z0-9-]+.+. [A-Z]{2,4}$/i';
14.
15.    /*
16.    На будущее: добавить больше условий к паролю
17.    Сейчас ограничение только по длине
18.    */
19.
20.    $errors = [];
21.    $error_flag = 0;
22.
23.    if (!is_valid_login($name)){
24.        $errors['name'] = 'Строка должна начинаться с буквы и
        заканчиваться ею. Длина строки - 4 до 31 символов';
25.        $error_flag = 1;
26.    } elseif (!is_login_unique($name, $link)){
27.        $errors['name'] = 'Этот логин уже занят';
28.        $error_flag = 1;
29.    }
30.    if (!is_valid_telephone($telephone)){
31.        $errors['telephone'] = 'Формат телефона не верный';
32.        $error_flag = 1;
33.    } elseif (!is_telephone_unique($telephone, $link)){
34.        $errors['telephone'] = 'Такой телефон уже есть в базе данных';
35.        $error_flag = 1;
36.    }
37.    if (!is_valid_email($email)){
38.        $errors['email'] = 'Формат email-адреса не верный';
39.        $error_flag = 1;
40.    } elseif (!is_email_unique($email, $link)){
41.        $errors['email'] = 'Такой email уже есть в базе данных';
42.        $error_flag = 1;
43.    }
44.    if (strlen($password) < 4){
45.        $errors['password'] = 'Слишком короткий пароль';
46.        $error_flag = 1;
47.    }
48.    if (strlen($confirm_password) < 4){
49.        $errors['confirm_password'] = 'Слишком короткий пароль';
50.        $error_flag = 1;
51.    }
52.
53.
54.    if ($password === $confirm_password and !$error_flag){
55.        //print("Пароли совпадают");

```



```
56.
57.      //Хешируем пароль. Алгоритмы MD5 и SHA1 устарели. Используем
      функцию password_hash()
58.      $hash = password_hash($password, PASSWORD_DEFAULT);
59.
60.      mysqli_query($link, "INSERT INTO `users`
61.      (`id`, `login`, `email`, `telephone`, `password`) VALUES
62.      (NULL, '$name', '$email', '$telephone', '$hash')");
63.
64.      $_SESSION['message'] = 'Регистрация прошла успешно';
65.      header('Location: ../index.php');
66.  }
67.  else{
68.      if ($password !== $confirm_password){
69.          $errors['password'] = 'Пароли не совпадают';
70.          $errors['confirm_password'] = 'Пароли не совпадают';
71.      }
72.      $_SESSION['errors'] = $errors;
73.      $_SESSION['POST'] = $_POST;
74.      //die($errors['name']);
75.      //print("Пароли НЕ совпадают");
76.      //die('пароли не совпадают'); //Передать это сообщение на клиент
77.      header('Location: ../registration.php');
78.  }
```