Санкт-Петербургское государственное бюджетное

профессиональное образовательное учреждение

«Ижорский колледж»

КУРСОВАЯ РАБОТА

ТЕМА: «Личный финансовый трекер – для учёта доходов и расходов»

по ПМ.11 МДК.11.01 Технология разработки и защиты баз данных

Выполнил

обучающийся группы 231c

Шестаков Иван Николаевич

Проверил

преподаватель спец. дисциплин

специальности 09.02.07

Информационные системы и программирование

Венедиктов Д.В.

Санкт-Петербург, 2025ЗАДАНИЕ ДЛЯ ВЫПОЛНЕНИЯ КУРСОВОЙ РАБОТЫ

1. Изучить теоретические основы технологии разработки баз данных, включая этапы проектирования, нормализации, реализации и тестирования.

2. Рассмотреть методы и средства защиты баз данных от несанкционированного доступа, атак и вредоносных программ.

3. Провести анализ существующих систем управления базами данных (СУБД) с точки зрения их функциональности, безопасности и удобства использования.

4. Разработать проект базы данных для конкретной предметной области, учитывая требования к безопасности и конфиденциальности данных.

5. Реализовать разработанный проект в выбранной СУБД.

6. Протестировать разработанную базу данных на предмет соответствия требованиям безопасности и надёжности.

7. Оценить эффективность разработанной системы защиты базы данных и предложить рекомендации по её улучшению.

8. Оформить результаты работы в виде курсовой работы, включающей введение, основную часть, заключение, список использованных источников и приложения.

В основной части курсовой работы необходимо подробно рассмотреть следующие вопросы:

* основные этапы разработки базы данных;
* методы обеспечения целостности и безопасности данных;
* механизмы аутентификации и авторизации пользователей;
* шифрование данных и защита от вредоносного ПО;
* особенности реализации проекта базы данных в выбранной СУБД;
* результаты тестирования и оценка эффективности системы защиты.

Для выполнения задания рекомендуется использовать современные научные источники, а также практический опыт разработки и защиты баз данных.

ПЛАН-ГРАФИК ВЫПОЛНЕНИЯ КУРСОВОЙ РАБОТЫ

|  |  |  |
| --- | --- | --- |
| Дата | Содержание работ | Отметка о выполнении |
| 10.09 | Выбор темы курсовой работы |  |
| 17.09 | Знакомство с методическими рекомендациями по выполнению курсовой работы |  |
| 24.09 | Знакомство с планом-графиком выполнения курсовой работы |  |
| 08.10 | Работа с нормативно-правовыми документами, учебной литературой |  |
| 22.10 | Анализ предметной области. Постановка задачи |  |
| 29.10 | Составление введения к курсовой работе |  |
| 05.11 | Описание предметной области и функции решаемых задач |  |
| 12.11 | Выбор средств для выполнения курсовой работы. Выбор среды разработки |  |
| 19.11 | Концептуально-логическое проектирование. Составление ER-диаграммы |  |
| 10.12 | Создание и заполнение базы данных |  |
| 24.12 | Представления в базе данных |  |
| 21.01 | Процедуры в базе данных |  |
| 04.02 | Создание ролей в базе данных |  |
| 18.02 | Аутентификация пользователей в базе данных |  |
| 25.02 | Импорт и экспорт базы данных |  |
| 04.03 | Разработка стратегии резервного копирования базы данных |  |
| 11.03 | Тестирование базы данных |  |
| 18.03 | Оптимизация базы данных |  |
| 25.03 | Составление заключения к курсовой работе |  |
| 01.04 | Составление списка источников информации, используемых при выполнении курсовой работы |  |
| 08.04 | Подготовка курсовой работы к защите |  |

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 7](#_Toc180594792)

[ГЛАВА 1. ФУНКЦИОНАЛЬНЫЕ ВОЗМОЖНОСТИ БАЗЫ ДАННЫХ 8](#_Toc180594793)

[1.1. Анализ предметной области 8](#_Toc180594794)

[1.2. Анализ технологий для разработки базы данных 8](#_Toc180594795)

[1.3. Постановка задачи курсовой работы 10](#_Toc180594796)

[1.3.1. Цели курсовой работы 10](#_Toc180594797)

[1.3.2. Задачи курсовой работы 11](#_Toc180594798)

[1.4.1. Функциональные требования к системе 11](#_Toc180594799)

[1.4.2. Нефункциональные требования к системе 11](#_Toc180594800)

[1.5. Выбор программных средств для разработки базы данных 11](#_Toc180594801)

[1.6. Выбор среды для разработки базы данных 14](#_Toc180594802)

[ГЛАВА 2. ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 17](#_Toc180594803)

[2.1. Концептуально-логическое моделирование 17](#_Toc180594804)

[2.1.1 Концептуальное моделирование 17](#_Toc180594805)

[2.1.2. Логическое моделирование 18](#_Toc180594806)

[2.2. Описание информационных объектов базы данных 20](#_Toc180594807)

[2.3.1. Создание базы данных 25](#_Toc180594808)

[2.3.2. Заполнение базы данных 25](#_Toc180594809)

[2.4. Представления в базе данных 27](#_Toc180594810)

[2.5. Процедуры в базе данных 30](#_Toc180594811)

[2.6. Создание ролей в базе данных 30](#_Toc180594812)

[2.7. Аутентификация пользователей в базе данных 30](#_Toc180594813)

[2.8. Импорт и экспорт базы данных 30](#_Toc180594814)

[2.9. Разработка стратегии резервного копирования базы данных 30](#_Toc180594815)

[2.10.1. Тестирование базы данных 31](#_Toc180594816)

[2.10.2. Оптимизация базы данных 31](#_Toc180594817)

[ЗАКЛЮЧЕНИЕ 32](#_Toc180594818)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 33](#_Toc180594819)

# ВВЕДЕНИЕ

В современном цифровом мире данные представляют собой один из ключевых ресурсов, влияющих на стратегические решения и операционные процессы организаций. Базы данных стали основным инструментом для хранения, обработки и управления информацией, обеспечивая эффективное взаимодействие с данными. Разработка баз данных – это сложный процесс, требующий комплексного подхода, включающего анализ требований, проектирование, реализацию и поддержку.

С учетом растущего объема информации и усложнения бизнес-процессов, значимость качественного проектирования баз данных трудно переоценить. Правильная структура базы данных обеспечивает не только ее функциональность и производительность, но и легкость в управлении данными, что в свою очередь влияет на оперативность принятия решений.

Однако с увеличением объема данных и многообразием технологий, связанных с их хранением, возникает необходимость уделять особое внимание защите данных. Утечки информации и кибератаки могут привести к значительным финансовым потерям и подрыву доверия со стороны клиентов. Поэтому технологии защиты баз данных становятся критически важными для бизнеса всех размеров.

Эта работа сосредоточится на ключевых аспектах разработки и защиты баз данных, включая методы проектирования и элементы безопасности, которые должны быть внедрены для обеспечения надежности и защиты данных в любой организации. Изучение этих тем поможет определить лучшие практики, которые должны применяться для достижения устойчивости и эффективности работы с информацией в условиях динамично меняющейся технологической среды.

# ГЛАВА 1. ФУНКЦИОНАЛЬНЫЕ ВОЗМОЖНОСТИ БАЗЫ ДАННЫХ

## 1.1. Анализ предметной области

Технология разработки и защиты баз данных охватывает широкий спектр процессов, методов и инструментов, необходимых для создания, управления и обеспечения безопасности баз данных. Это область, которая активно развивается в ответ на растущие требования к хранению и защите информации, что делает её актуальной для различных отраслей, таких как финансы, здравоохранение, образование и многие другие.

## 1.2. Анализ технологий для разработки базы данных

Разработка баз данных включает в себя использование различных технологий и инструментов, которые помогают осуществлять проектирование, реализацию, управление и оптимизацию баз данных. Рассмотрим основные технологии и подходы, которые широко применяются в этой сфере.

**Системы управления базами данных (СУБД)**

СУБД являются основными инструментами для создания и управления базами данных. Они могут быть разделены на несколько категорий:

* реляционные СУБД (RDBMS): MySQL, PostgreSQL, Oracle, Microsoft SQL Server;
* нереляционные СУБД (NoSQL): MongoDB, Cassandra, Redis;
* гибридные СУБД: Microsoft Azure Cosmos DB.

**Языки программирования**

* SQL (Structured Query Language): основной язык для работы с реляционными базами данных. Позволяет создавать, изменять, запрашивать и управлять данными;
* Python и JavaScript: часто используются для взаимодействия с базами данных в веб-приложениях. Библиотеки, такие как SQLAlchemy или Sequelize, упрощают работу с базами данных;
* Java и C#: используются в крупных корпоративных приложениях, особенно с реляционными базами данных.

**Инструменты для проектирования баз данных**

* ER-моделирование: Инструменты, такие как Lucidchart, Draw.io или ER/Studio, позволяют создавать диаграммы сущностей и взаимосвязей (ER-диаграммы), которые помогают визуализировать структуру базы данных;
* системы управления версиями: Git и другие системы позволяют контролировать изменения в схемах баз данных и процессах миграции.

**Инструменты для обеспечения безопасности баз данных**

* шифрование: использование технологий шифрования данных (например, AES) для защиты информации как «в покое», так и «в движении»;
* аутентификация и авторизация: применение OAuth, OpenID Connect и других протоколов для управления доступом к базе данных;
* мониторинг и аудит: инструменты, такие как Splunk, могут использоваться для отслеживания запросов и анализа событий безопасности.

**Методы оптимизации производительности**

* индексация: создание индексов для ускорения выполнения запросов, особенно в реляционных базах данных.
* кэширование: использование кэшей, таких как Redis или Memcached, для ускорения доступа к часто запрашиваемым данным.
* балансировка нагрузки: распределение запросов между несколькими серверами баз данных для повышения производительности и отказоустойчивости.

Выбор технологий для разработки базы данных зависит от специфики проекта, объема данных, ожидаемых нагрузок и требований к безопасности. Комплексный подход к выбору инструментов и методов способствует созданию эффективных и надежных систем управления данными, которые могут адаптироваться к изменяющимся условиям и требованиям.

## 1.3. Постановка задачи курсовой работы

С ростом объемов данных, а также увеличением угроз безопасности информации, эффективная разработка и защита баз данных становятся крайне важными задачами для организаций. Безопасность данных имеет критическое значение для решения вопросов конфиденциальности, целостности и доступности информации.

Успешное выполнение работы будет способствовать более глубокому пониманию актуальных проблем и новых подходов в области безопасного управления данными.

### 1.3.1. Цели курсовой работы

Основная цель заключается в создании эффективной и безопасной базы данных, способной обеспечить надежный доступ к информации и защиту от потери или несанкционированного доступа.

### 1.3.2. Задачи курсовой работы

* проектирование архитектуры баз данных;
* разработка схемы данных и моделей;
* обеспечение безопасности данных через шифрование, аутентификацию и авторизацию;
* реализация механизмов резервного копирования и восстановления;
* мониторинг и аудит доступа к данным.

## 1.4.1. Функциональные требования к системе

* возможность создания, обновления и удаления данных;
* поддержка сложных запросов и отчетов;
* обеспечение многопользовательского доступа.

## 1.4.2. Нефункциональные требования к системе

* высокая производительность и надежность;
* защита от несанкционированного доступа;
* удобный интерфейс для пользователей.

## 1.5. Выбор программных средств для разработки базы данных

Выбор программных средств для разработки баз данных является ключевым этапом, который влияет на функциональность, производительность и безопасность системы.

При выполнении данной курсовой работы я планирую использовать следующие программные средства:

* система управления базами данных – MySQL;
* языки программирования – SQL, JavaScript;
* средства проектирования базы данных – Lucidchart;
* инструменты для обеспечения безопасности данных – Vormetric, Splunk;
* средства тестирования и оптимизации приложений – JMeter, Database Engine Tuning Advisor.

Выбор программных средств для разработки базы данных зависит от специфических требований проекта, таких как объем данных, тип доступа, требования к безопасности и производительности. Важно учитывать будущую масштабируемость и поддерживаемость решений, чтобы создать надежную и эффективную систему.

## 1.6. Выбор среды для разработки базы данных

Выбор подходящей среды для разработки базы данных является критически важным этапом проекта, который может оказать значительное влияние на эффективность работы команды и качество конечного продукта.

При выполнении данной курсовой работы я планирую использовать следующие среды и инструменты:

* IDE и редактор кода – MySQL;
* система контейнеризации – Docker;
* система управления версиями – git;
* фреймворк и библиотека – Hibernate;

Выбор среды для разработки базы данных требует комплексного подхода с учетом специфики проекта, требований к производительности, безопасности и интеграции. Правильный выбор инструментария и технологии существенно влияет на успех проекта, поэтому следует тщательно проанализировать свои потребности и доступные решения.

# ГЛАВА 2. ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

## 2.1. Концептуально-логическое моделирование

Концептуально-логическое моделирование – это важный этап в проектировании базы данных, который позволяет структурировать данные и определить их взаимосвязи. Этот процесс разделяется на два ключевых этапа: концептуальное моделирование и логическое моделирование.

### 2.1.1 Концептуальное моделирование

Целью концептуального моделирования является определение основных сущностей и их связей в предметной области, не вдаваясь в детали реализации.

**Идентификация сущностей**

* Users
* Account
* Transaction
* Category
* Budget
* Currency

**Определение атрибутов**

* Users: ID, Name, Email, Date, Password, Created\_at, Avatar, Is\_admin.
* Account: ID, Name, **Balance**, Currency, UserID.
* Transaction: ID, Type, Amount, Date, AccontID, CategoryID.
* Category: ID, Name.
* Budget: ID, Amount, Period, UserID, CategoryID.
* Currency: Code, Name.

**Установление связей**

* Пользователь и Счёт (1:N).
* Пользователь и Бюджет (1:N).
* Счёт и Транзакция (1:N).
* Категория и Транзакция (1:1).
* Валюта и Счёт (1:1).
* Бюджет и Категория (1:N).

**Создание ER-диаграммы**

В данной диаграмме прямоугольные блоки отвечают за сущности, овальные – атрибуты, ромбы - связи

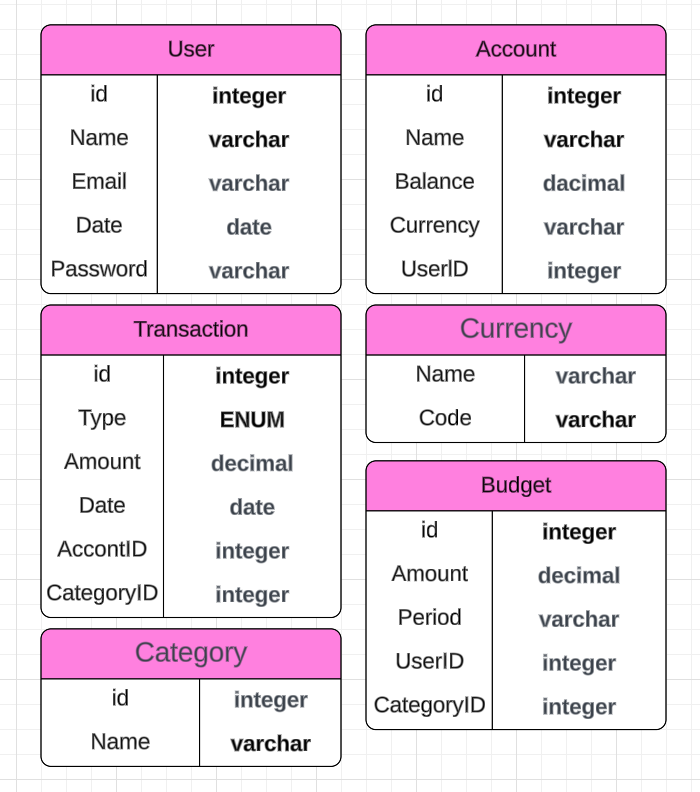


Рисунок 1 – ER-диаграмма разрабатываемой базы данных

### 2.1.2. Логическое моделирование

Целью логического моделирования является перевод концептуальной модели в логическую, с уточнением структуры базы данных и обязательств для хранения данных.

**Определение первичных ключей**

* Пользователь: ID.
* Счёт: ID.
* Транзакция: ID.
* Категория ID.
* Бюджет: ID.
* Валюта: Code.

**Формализация связей**

* Для связи между пользователем и счётом требуется: UserID.
* Для связи между счётом и транзакцией требуется: AccontID.
* Для связи между категорией и транзакцией требуется: CategoryID.
* Для связи между бюджетом и пользователем требуется: UserID.
* Для связи между бюджетом и категорией требуется: CategoryID.
* Для связи между счётом и валютой требуется: Code.

**Определение типов данных**

* ID: INTEGER
* Текст: VARCHAR
* Дата: DATETIME
* Имя: VARCHAR
* Пароль: VARCHAR
* Email: VARCHAR

**Создание логической схемы базы данных**

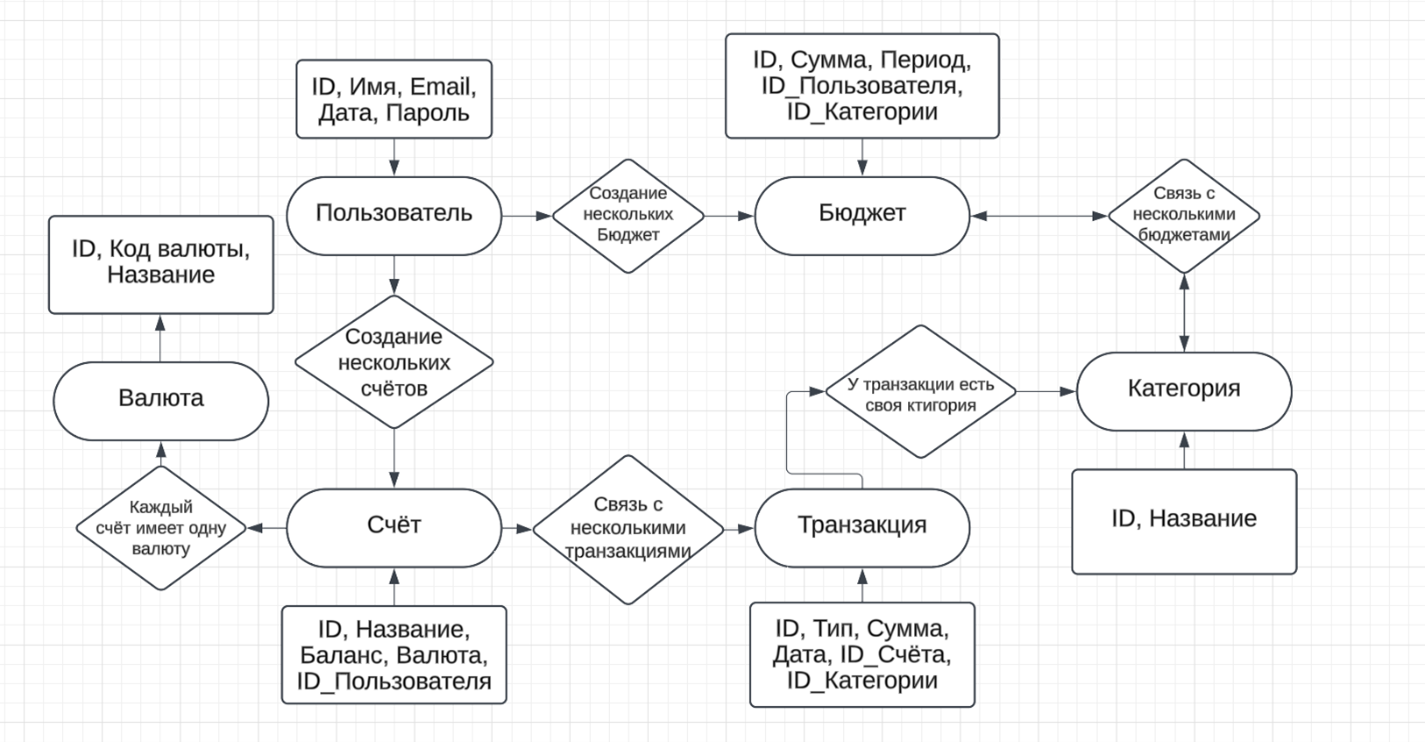


Рисунок 2 – Логическая схема базы данных

Концептуально-логическое моделирование является важным этапом разработки базы данных, который позволяет углубиться в детализацию структуры данных и обеспечить целостность и согласованность информации. Правильное моделирование позволяет избежать проблем на следующих этапах разработки и эксплуатации базы данных.

## 2.2. Описание информационных объектов базы данных

Информационные объекты базы данных представляют собой основные строительные блоки, на которых базируется структура любой базы данных. Они включают в себя различные сущности и их атрибуты, которые позволяют организовать, хранить и управлять данными.

Таблица 1 – информационные объекты разрабатываемой базы данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Объект | Атрибуты | Связи | Ограничения | Тип данных |
| User | ID, Name, Email, Date, Password | Один пользователь может создать Счёт,  Один пользователь может создать Бюджет | Email и ID должны быть уникальными,  Пароль должен быть обязательно | INTEGER, VARCHAR,  DATE |
| Account | ID, Name, Balance, Currency, UserID | Счёт пользователя связан с Транзакцией,  Cчёт пользователя связан с Валютой. | ID уникален, UserID - внешний ключ | INTEGER, VARCHAR, DECIMAL |
| Transaction | ID, Type, Amount, Date, AccountID, CategoryID | Транзакция связана с Счётом, Транзакция связана с Категорией. | ID уникален, AccountID и CategoryID - внешний ключ | INTEGER, VARCHAR, DECIMAL, DATE, ENUM |
| Category | ID, Name | Категория связана с Транзакцией, Категория связана с Бюджетом. | ID уникален, Name уникально | INTEGER, VARCHAR |
| Currency | Code, Name | Валюта связана с Счётом | Code уникален | VARCHAR |
| Budget | ID, Amount, Period, UserID, CategoryID | Бюджет связан с Пользователем,  Бюджет связан с Категорией | ID уникален, UserID и CategoryID - внешние ключи | INTEGER, VARCHAR, DECIMAL |

Описание информационных объектов базы данных позволяет структурировать и организовать информацию, обеспечивая эффективное хранение и доступ к данным. Правильное определение сущностей, их атрибутов и связей является ключом к успешному проектированию базы данных и ее функциональной полноте.

## 2.3.1. Создание базы данных

Для того чтобы создать базу данных можно использовать консольные SQL-запросы или взаимодействие с интерфейсом phpMyAdmin, здесь будет использован вариант с отправлением SQL-запросов. Создаем базу данных с стандартным именем.

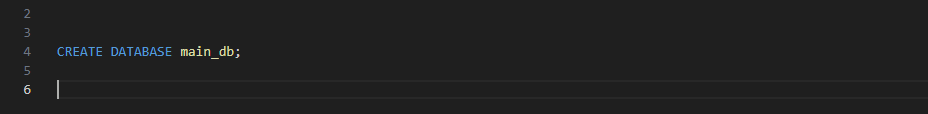


Рисунок 3 – созданная база данных

## 2.3.2. Заполнение базы данных

Следующим шагом является создание таблиц для хранения данных. Каждая таблица создаётся с учётом структуры, описанной на этапе концептуального моделирования. Для начала будет создана таблица «Пользователи».

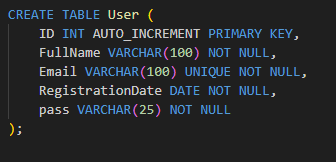


Рисунок 4 – заполнение базы данных

После успешного ответа от SQL-запроса можно увидеть таблицу в нашей базе данных.

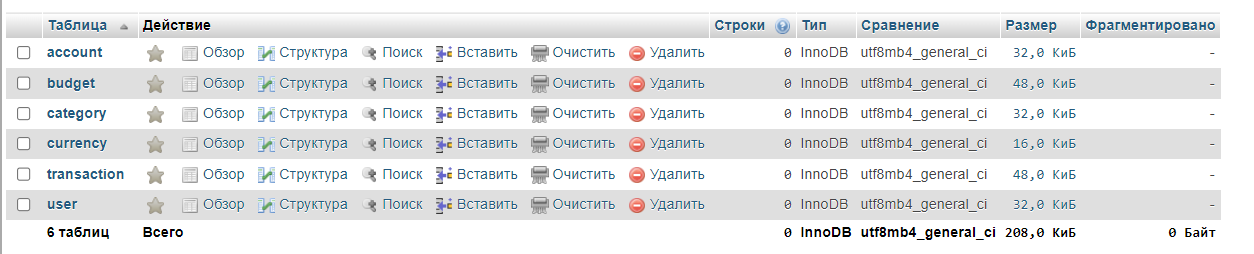


Рисунок 5 – добавление объекта в базу данных

По такой же аналогии были созданы еще три таблицы: «Счёт», «Транзакция», «Категория», «Бюджет», «Валюта» - которые представляют из себя такие же объекты.

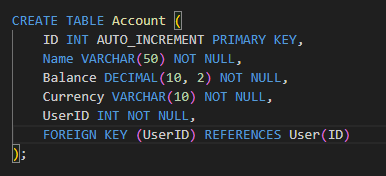


Рисунок 6 – созданная таблицы «Счёт»

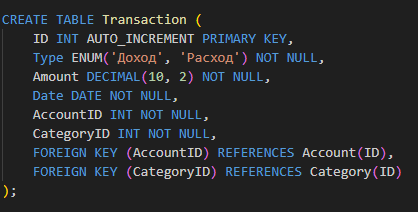


Рисунок 7 – созданная таблицы «Транзакция»

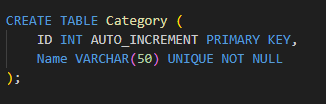


Рисунок 8 – созданная таблицы «Категория»

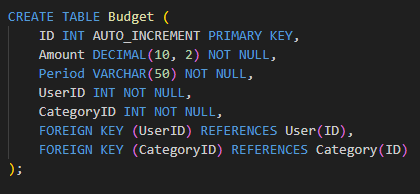


Рисунок 9 – созданная таблицы «Бюджет»

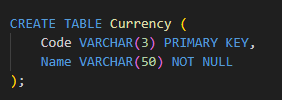


Рисунок 10 – созданная таблицы «Валюта»

После успешного выполнения SQL-запросов структура базы данных готова, на следующем этапе можно будет установить связи между объектами в базе данных.

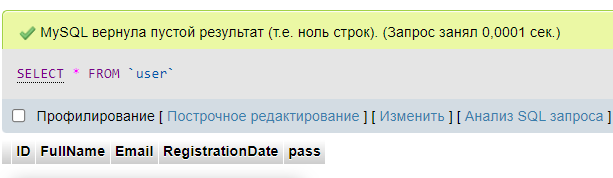


Рисунок 11 – добавление атрибутов объекту в базе данных

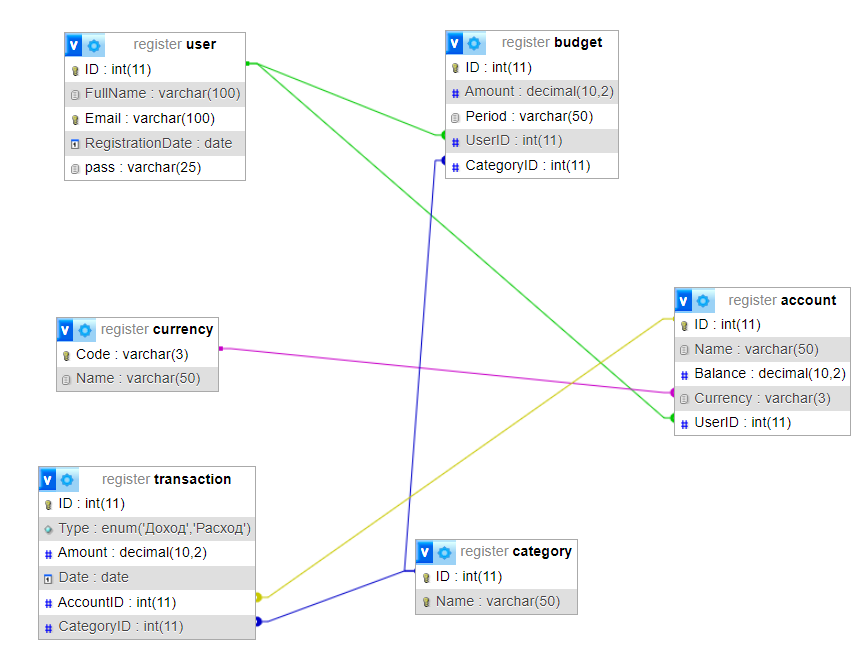


Рисунок 12 – установление связей между объектами в базе данных

Выбор типа данных для объекта в базе данных:

|  |  |
| --- | --- |
| ID | int |
| Name | varchar |
| Balance | decimal |
| Currency | varchar |
| UserID | int |
| Amount | decimal |
| Period | varchar |
| CategoryID | int |
| Code | varchar |
| Type | enum |
| Date | date |
| AccountID | int |
| FullName | varchar |
| Email | varchar |
| RegistrationDate | date |
| pass | varchar |

## 2.4. Представления в базе данных

Представление в базе данных – это виртуальная таблица, которая формируется на основе результата выполнения SQL-запроса. Она не содержит данных, а лишь определяет, как данные будут отображаться пользователям.

Функции представлений:

* упрощение сложных запросов – позволяют объединять данные из нескольких таблиц и показывать только нужные поля, что делает работу с данными более удобной;
* безопасность – можно ограничить доступ пользователей к определённым столбцам или строкам баз данных, предоставляя доступ только к представлениям;
* логическая независимость – изменения в базах данных не влияют на внешний интерфейс, так как представления могут скрывать изменения структуры таблиц;
* кодовая реиспользуемость – позволяют сократить код, избавляя от необходимости повторно писать одни и те же запросы.

Представление создается с помощью команды CREATE VIEW.

Представление создается с помощью команды CREATE VIEW. Пример представления, которое отображает все транзакции, включая их суммы, даты, категории и счета, но только для активных пользователей:

|  |
| --- |
| sql  CREATE VIEW active\_transactions AS  SELECT  t.ID AS TransactionID,  t.Amount,  t.Date,  c.Name AS Category,  a.Name AS Account,  u.FullName AS UserName  FROM  Transaction t  JOIN  Account a ON t.AccountID = a.ID  JOIN  Category c ON t.CategoryID = c.ID  JOIN  User u ON a.UserID = u.ID  WHERE  u.ID IS NOT NULL; |

После создания представления его можно использовать так же, как и таблицу. Данный запрос, возвращающий все транзакции, относящиеся к категории "Продукты".

|  |
| --- |
| sql  SELECT \* FROM transaction\_summary WHERE Category = 'Продукты'; |

Представления могут быть обновляемыми или не обновляемыми. Обновляемые представления позволяют выполнять операции INSERT, UPDATE и DELETE, при этом изменения будут применяться к базовой таблице.

Допустим, представление transaction\_summary отображает данные из таблицы Transaction, и вы хотите обновить поле Amount для транзакции с ID = 5:

|  |
| --- |
| UPDATE transaction\_summary  SET Amount = 1000.00  WHERE TransactionID = 5; |

Представления можно изменять с помощью команды CREATE OR REPLACE VIEW или удалять с помощью DROP VIEW

Допустим, необходимо изменить представление transaction\_summary, чтобы добавить информацию о пользователе и фильтровать только транзакции, выполненные после определённой даты.

|  |
| --- |
| CREATE OR REPLACE VIEW transaction\_summary AS  SELECT  t.ID AS TransactionID,  t.Amount,  t.Date,  c.Name AS Category,  a.Name AS Account,  u.FullName AS UserName  FROM  Transaction t  JOIN  Account a ON t.AccountID = a.ID  JOIN  Category c ON t.CategoryID = c.ID  JOIN  User u ON a.UserID = u.ID  WHERE  t.Date >= '2024-01-01'; |

Если представление больше не требуется, его можно удалить:

|  |
| --- |
| sql  DROP VIEW transaction\_summary; |

Представления в базах данных являются мощным инструментом для работы с данными. Они обеспечивают большую гибкость, безопасность и легкость в использовании, способствуя более эффективной организации и обработке информации. Понимание работы с представлениями – важный аспект для разработчиков и администраторов баз данных.

## 2.5. Процедуры в базе данных

(2 курс 2 семестр)

## 2.6. Создание ролей в базе данных

(2 курс 2 семестр)

## 

## 2.7. Аутентификация пользователей в базе данных

Аутентификация пользователей — это процесс проверки их учетных данных перед предоставлением доступа к системе. В данной курсовой работе реализована система регистрации и входа пользователей, использующая безопасное хранение паролей с хешированием.

**Реализация регистрации пользователя**

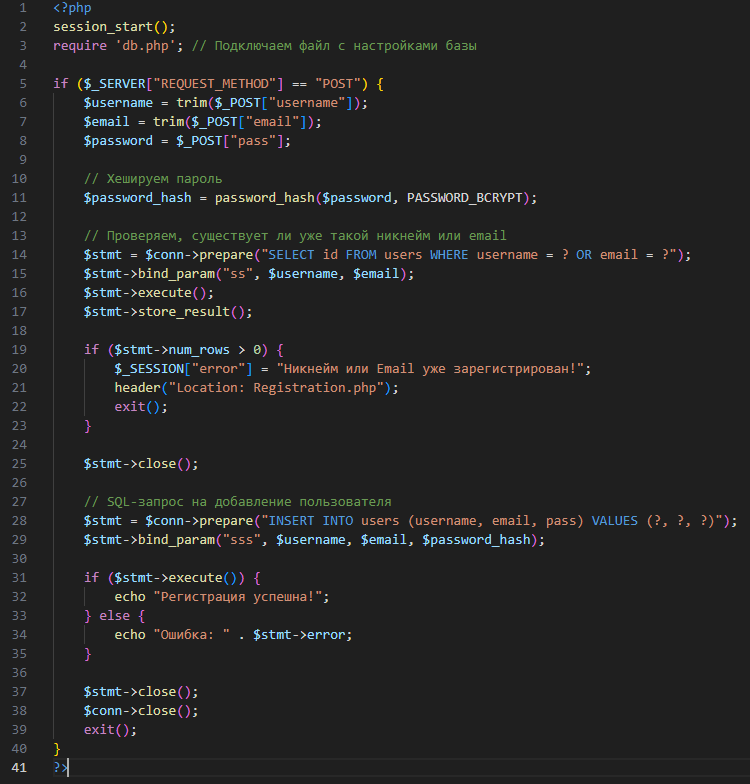
****

Рисунок 13 – реализация регистрации пользователя

Обоснование кода:

* Полученные данные обрабатываются и очищаются (`trim()`).
* Пароль хешируется с помощью `password\_hash()` (алгоритм `BCRYPT`).
* Проверяется уникальность `username` и `email`.
* Если таких записей нет, выполняется SQL-запрос на добавление пользователя.

**Реализация входа пользователя**

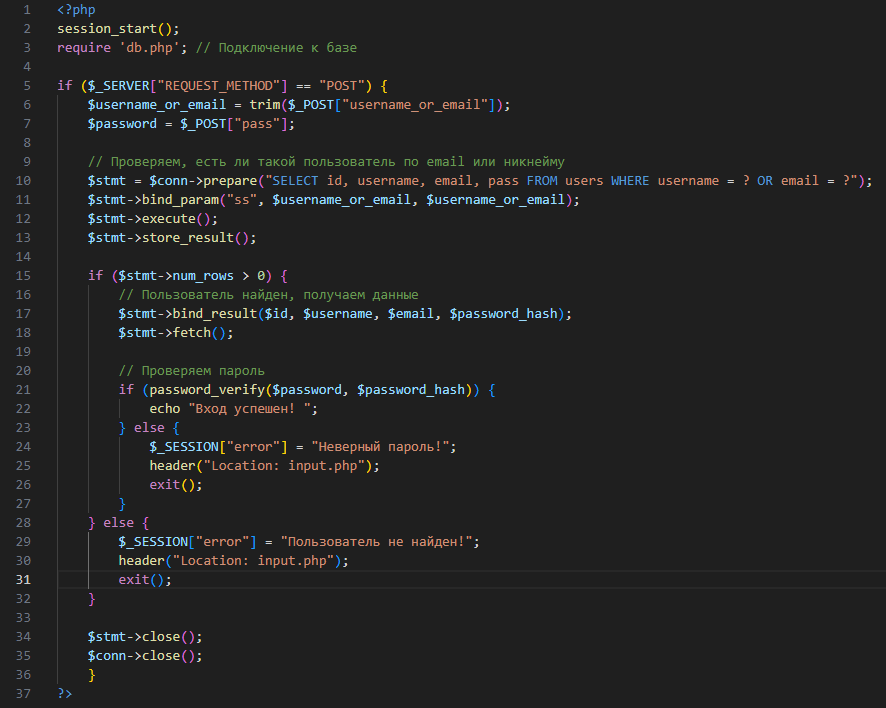
****

Рисунок 14 – реализация входа пользователя

Обоснование кода:

* Запрос ищет пользователя по `username` или `email`.
* Если пользователь найден, `password\_verify()` сверяет введённый пароль с хешем.
* В случае успешного входа создаётся сессия (`$\_SESSION["user\_id"]`).

**Подключение к базе данных**

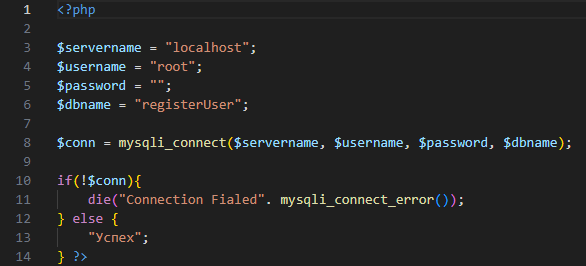
****

Рисунок 15 – подключение к базе данных

Обоснование кода**:**

* Используется ` mysqli\_connect` для соединения с базой данных.
* В случае ошибки соединения выводится сообщение `die()`

**Итоговый процесс работы системы аутентификации**

Регистрация:

* Пользователь вводит `username`, `email`, `password`.
* Система проверяет уникальность данных.
* Пароль хешируется и сохраняется в базе.
* Пользователь получает сообщение об успешной регистрации.

Вход в систему:

* Пользователь вводит `username` или `email` и `password`.
* Система проверяет существование пользователя.
* Пароль сверяется с хешем.
* При успешном входе создаётся сессия.

## 2.8. Импорт и экспорт базы данных

Импорт и экспорт базы данных – это важные процессы для сохранения резервных копий, переноса базы данных на другой сервер и восстановления информации. В данном проекте для работы с базой данных используется phpMyAdmin, что позволяет выполнять эти операции без использования командной строки.

**Экспорт базы данных**

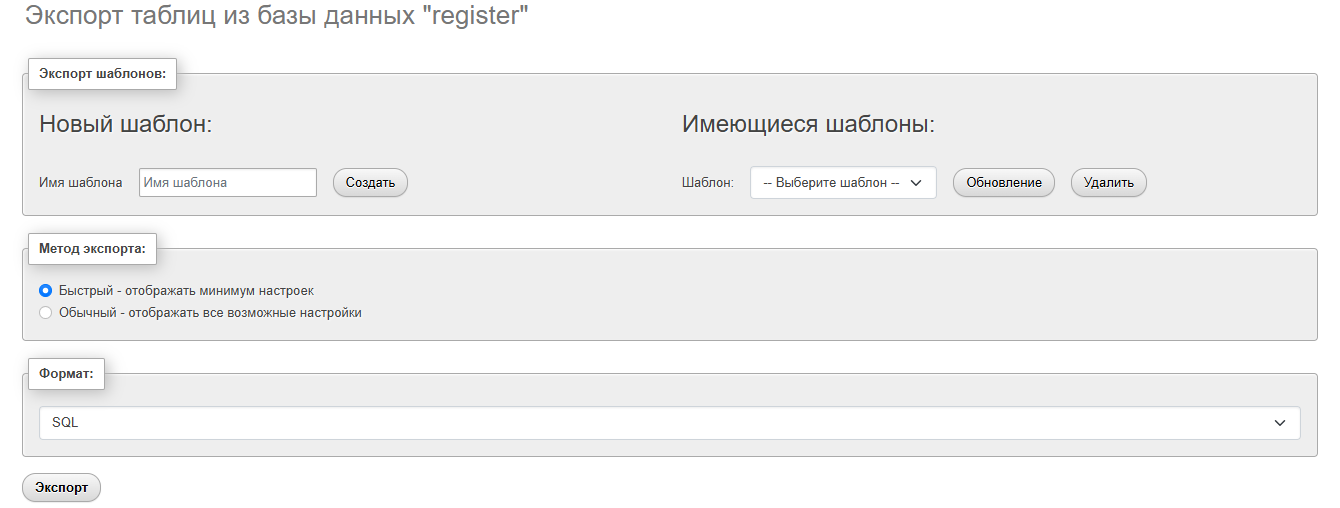
****

Рисунок 16 – экспорт базы данных

Для создания резервной копии базы данных выполните следующие шаги:

1. Выбрать базу данных в phpMyAdmin.
2. Перейти во вкладку "Экспорт".
3. Выбрать "Настроенный" метод.
4. В разделе "Таблицы" отметить только нужные таблицы.
5. Нажать "ОК" – будет загружен файл `.sql`, содержащий только выбранные таблицы.

**Импорт базы данных**

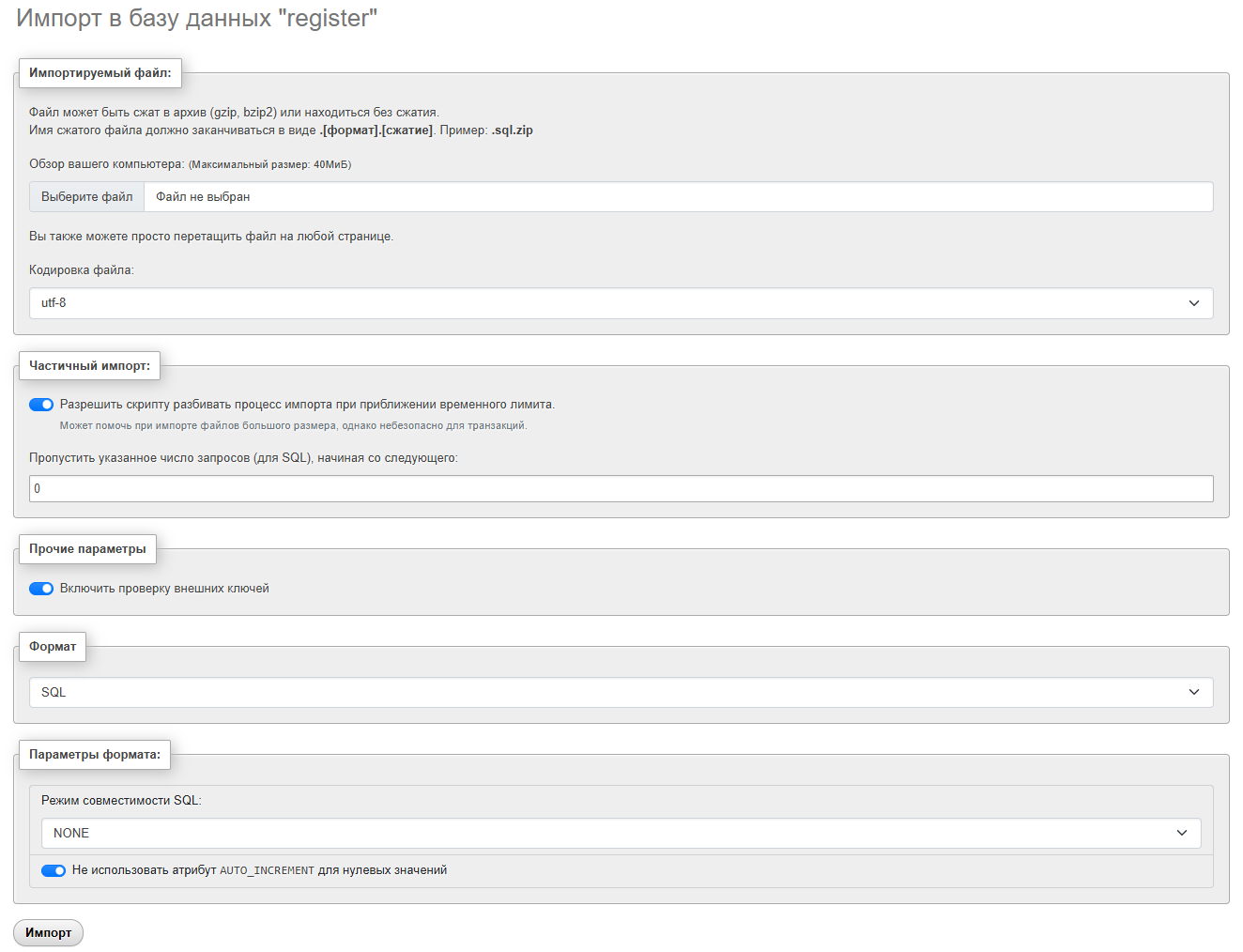
****

Рисунок 17 – импорт базы данных

Чтобы восстановить базу данных из резервной копии:

1. Открыть phpMyAdmin.
2. Если база данных отсутствует, создать её:

* Перейти во вкладку "Базы данных".
* Ввести имя базы и нажать "Создать".

1. Выбрать созданную базу.
2. Перейти во вкладку "Импорт".
3. Нажать "Выберите файл" и загрузить `.sql` файл.
4. Нажать "ОК", и база данных будет восстановлена.

## 2.9. Разработка стратегии резервного копирования базы данных

(2 курс 2 семестр)

## 2.10.1. Тестирование базы данных

Тестирование базы данных — это процесс проверки её корректной работы, целостности данных, производительности и безопасности.

**Регистрация**

**Проверка никнейма**

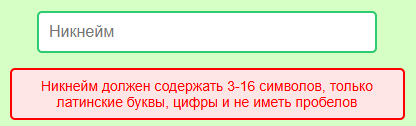


Рисунок 18 – проверка никнейма

Проверка никнейма нужна для обеспечения уникальности, безопасности и соответствия правилам платформы. Она предотвращает дублирование имён пользователей, фильтрует запрещённые или оскорбительные слова, соблюдает технические ограничения (например, длину и допустимые символы), а также защищает от мошенничества и фишинга. Это помогает создать удобную и безопасную среду для всех пользователей.

**Проверка почты**

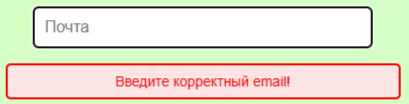
****

Рисунок 19 – проверка почты

Проверка почты необходима для подтверждения её работоспособности, предотвращения ошибок при вводе, защиты от спама и мошенничества. Она включает проверку формата email, существования домена, наличия почтового ящика и, при необходимости. Это помогает обеспечить безопасность аккаунтов и корректную работу системы.

**Проверка пароля**

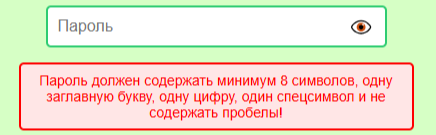
****

Рисунок 20 – проверка пароля

Проверка пароля необходима для обеспечения безопасности аккаунта. Она позволяет убедиться, что пароль соответствует требованиям платформы (например, минимальная длина, наличие цифр, букв и спецсимволов). Это защищает аккаунт от взлома, подборов пароля и утечек данных.

**Подтверждение пароля**

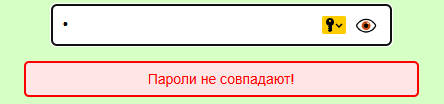
****

Рисунок 21 – подтверждение пароля

Подтверждение пароля необходимо для предотвращения ошибок при вводе и обеспечения безопасности аккаунта. Пользователь вводит пароль дважды, и система проверяет, совпадают ли оба варианта. Это снижает риск опечаток и позволяет избежать проблем с входом в аккаунт в будущем.

**Я принимаю все условия**

****

Рисунок 22 – я принимаю все условия

Фраза «Я принимаю все условия» обычно используется для подтверждения согласия с правилами и политиками сайта, сервиса или приложения. Это означает, что пользователь ознакомился с условиями использования, политикой конфиденциальности и другими положениями, и соглашается их соблюдать.

**Никнейм или Email уже зарегистрированы**

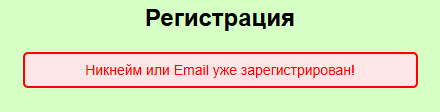
****

Рисунок 23 – никнейм или Email уже зарегистрированы

Сообщение «Никнейм или Email уже зарегистрированы» означает, что введённые данные уже используются в системе. Это может означать, что у вас уже есть аккаунт.

**Вход**

**Пользователь не найден**

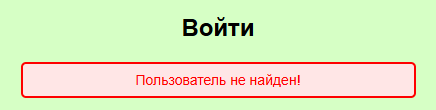
****

Рисунок 24 – пользователь не найден

Сообщение «Пользователь не найден» означает, что введённый никнейм или email отсутствует в базе данных.

**Неверный никнейм, почта или пароль**

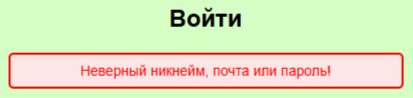
****

Рисунок 25 – неверный никнейм, почта или пароль

Сообщение «Неверный никнейм, почта или пароль» означает, что введённые данные не совпадают с зарегистрированными в системе.

## 2.10.2. Оптимизация базы данных

(2 курс 2 семестр)

# ЗАКЛЮЧЕНИЕ

В ходе изучения технологии разработки и защиты баз данных было выявлено, что успешная работа с данными в современных условиях требует комплексного подхода, включающего как аспекты проектирования, так и обеспечения безопасности. Принимая во внимание стремительное развитие информационных технологий, применение современных методов и инструментов становится неизбежным. Это позволяет не только оптимизировать процессы хранения и обработки данных, но и защитить их от возможных угроз.

Особое внимание следует уделить вопросам шифрования данных, аутентификации пользователей и резервного копирования. Эффективная защита баз данных требует активного мониторинга и регулярного обновления систем безопасности.

Таким образом, подходы к разработке и защите баз данных должны быть гибкими и адаптивными, чтобы соответствовать быстро меняющимся требованиям и угрозам. В будущем важно продолжать исследовать и внедрять новые технологии, что позволит обеспечить надежность и безопасность информации на всех уровнях.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Волик, М.В. Разработка базы данных в Access / М.В. Волик. – Издательство «Прометей», 2021. – 88 с.
2. Даккет, Дж. PHP и MySQL. Серверная веб-разработка / Дж. Даккет. – Издательство «Эксмо», 2023. – 688 с.
3. Дьяков, И.А. Базы данных. Язык SQL : учебное пособие / И.А. Дьяков. – Тамбов : Тамбовский государственный технический университет (ТГТУ), 2022. – 82 с.
4. Ла Рокка, М. Продвинутые и структуры данных. / М. ла Рокка. – Издательство «Питер», 2024. – 848 с.
5. Никсон, Р. Создаём динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5. 6-е изд. / Р. Никсон. – Издательство «Питер»: Санкт-Петербург, 2023. – 832 с.
6. https://www.figma.com/ (дата обращения: 15.03.2025).
7. https://www.github.com/ (дата обращения: 16.03.2025).

ПРИЛОЖЕНИЯ

(Скриншоты разработанной базы данных)