МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ПЕРМСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»

*КОЛЛЕДЖ ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ*

**РАЗРАБОТКА БАЗЫ ДАННЫХ СИСТЕМЫ РАСЧЕТА С КЛИЕНТАМИ В БАНКЕ**

|  |  |
| --- | --- |
|  | Курсовая работа  студента 2 курса  группы ИСП-1,2 2019 БС  специальности «Информационные системы и программирование»  ФИО Нефедов Г.А./ \_\_\_\_\_\_\_\_\_\_\_  студент подпись  ФИО Серебрякова Н.А./ \_\_\_\_\_\_\_\_\_\_  руководитель подпись |

Пермь 2021

Оглавление

[Введение 2](#_Toc73228288)

[Основная часть 4](#_Toc73228289)

[Раздел 1. Анализ бизнес-процессов предметной области 4](#_Toc73228290)

[1.1 Описание предметной области 4](#_Toc73228291)

[1.1.1 Общее описание предметной области, функционал БД 4](#_Toc73228292)

[1.1.2 Описание входных документов 5](#_Toc73228293)

[1.1.3 Описание выходных документов и сообщений 5](#_Toc73228294)

[1.1.4 Описание запросов к базе данных 6](#_Toc73228295)

[1.1.5 Список ограничений 9](#_Toc73228296)

[1.2 Выбор и описание используемой среды разработки 10](#_Toc73228297)

[1.2.1 Описание используемой среды разработки базы данных 10](#_Toc73228298)

[1.2.2 Описание используемой среды разработки приложения 11](#_Toc73228299)

[1.3 Инфологическая модель (ИЛМ) 14](#_Toc73228300)

[1.4 Даталогическая модель 15](#_Toc73228301)

[1.4.1 Нормализованная реляционная модель 15](#_Toc73228302)

[1.4.2 Состав таблиц базы данных 15](#_Toc73228303)

[Раздел 2. Разработка базы данных и приложения 17](#_Toc73228304)

[2.1 Разработка серверной части (базы данных) 17](#_Toc73228305)

[2.1.1 Таблицы, связи между таблицами, индексы 17](#_Toc73228306)

[2.2 Разработка клиентской части 21](#_Toc73228307)

[2.2.1 Главная (управляющая) форма, форма регистрации, авторизации, запрос пароля (разграничение доступа) 21](#_Toc73228308)

[2.2.2 Формы поиска, визуализации запросов 29](#_Toc73228309)

[2.3 Руководство по установке и работе с приложением и БД. 31](#_Toc73228310)

[2.4 Обеспечение информационной безопасности создаваемой базы данных 32](#_Toc73228311)

[Заключение 32](#_Toc73228312)

[Список используемой литературы 34](#_Toc73228313)

# **Введение**

“Разработка базы данных подсистемы расчета с клиентами в банке”

**Актуальность темы исследования** – данная тема очень актуальна, потому что с очень много людей пользуется услугами банков, которым нужно хранить очень много информации и получать данные по клиентам в кратчайшие сроки для того, чтобы оказывать свои услуги как можно быстрее и удобнее для клиентов. С течением времени всё больше людей предпочитают использовать интернет для управления своими банковскими счетами, нежели через офисы банков. Пандемия COVID-19 еще сильнее подтолкнула людей к прекращению посещения банков лично.

**Цель и задачи курсовой работы:**

Цель: разработка базы данных подсистемы расчета с клиентами в банке.

Задачи:

1. Выбрать инструменты разработки.

2. Построить модель базы данных.

3. Спроектировать базу данных в выбранной СУБД.

4. Разработать приложение для управления базой данных на выбранном языке программирования.

5. Тестировать БД и приложение.

**Предмет исследования –** база данных, хранящая информацию о пользователях.

**Объект исследования –** база данных банковского приложения.

Методы исследования:

1. метод анализа

2. метод абстрагирования

3. методы проведения тестирования,

4.методы разработки

5. метод моделирования.

**Структура работы:** Структура курсовой работы состоит из введения, двух разделов, приложений, списка использованных источников. Разработанная база данных имеет четыре таблицы. Главная таблица имеет связь с тремя другими таблицами. Это было сделано для того, чтобы было проще отслеживать карты и заявки пользователей. Для более удобного использования базы данных было разработано приложение, которое позволяет не только администраторам смотреть на данные, находящиеся в базе, но и пользователям решать собственные задачи. Такие как открытие или закрытие кредитных карт, оформление дебетовых карт и переводы денег на счета других пользователей. Всё это реализовано при помощи СУБД MySQL и языков программирования C#, Python и VBScript.

# **Основная часть**

# **Раздел 1. Анализ бизнес-процессов предметной области**

## **1.1 Описание предметной области**

### **1.1.1 Общее описание предметной области, функционал БД**

Областью применения БД является банковская сфера. Банковская сфера делится на розничный банкинг (обслуживание физических лиц), корпоративный банкинг (обслуживание организаций) и инвестиционный банкинг (инвестирование средств). В случае данного проекта больше всего подходит розничный банкинг. Розничный банкинг – это обслуживание физических лиц, предоставление им возможности для вкладов, кредитов, инвестирования, залогов, обмена валюты и обслуживания их банковских карт, дебетовых или кредитных.

В этом курсовом проекте была разработана база данных для банка “CyberBank”, в которой хранятся данные об аккаунтах пользователей, их дебетовых картах, заявки по кредитам и данные о кредитных картах. Для упрощения работы с базой данных, информация выводится в приложение, которое связано с базой данных. Данные, которые не являются строго конфиденциальными, администратор может видеть в приложении, если он зайдет под своим аккаунтом. Администратор также может просматривать заявки на кредитные карты, которые оставляют пользователи, через приложение, и подтверждать, либо отклонять их. Строго конфиденциальная информация зашифрована и недоступна сотрудникам банка, для защиты от утечек пользовательских аккаунтов, в случае если взломают аккаунт администратора.

Для пользователей реализована возможность перевода денег на счет других пользователей, а также возможность заказать кредитную карту, а затем закрыть её, если выполнены условия банка. При одобрении получения кредитной карты пользователю приходит письмо на почту, которую он указывает при оформлении заявки. Также имеется возможность регистрации через приложение, вследствие чего пользователь автоматически будет добавлен в базу данных, а затем сможет авторизоваться в приложении.

### **1.1.2 Описание входных документов**

В результате разработки базы данных в ней используются следующие входные документы:  
 -Таблица “users”;

-Таблица “e\_carts”;

-Таблица “credit\_requests”;

-Таблица “credit\_cards”;

### **1.1.3 Описание выходных документов и сообщений**

Для работы с базой данных в приложении были разработаны формы, которые выводят информацию на экран и упрощает работу с базой данных для администраторов.

База данных и приложение предоставляют следующие возможности:

-Закрытый доступа: только для администраторов банка

-Просмотр информации в формах.

-Добавление новой информации в базу данных.

### **1.1.4 Описание запросов к базе данных**

При работе с базой данных через приложение используется много различных запросов по типу SELECT, INSERT, UPDATE, DELETE.

1. Простой запрос на выборку по ID.

Код запроса на языке SQL: SELECT {what} FROM {from} WHERE {where} = @id;

Данный запрос делает выборку нужной информации из нужного места основываясь на id пользователя. Данные {what}, {from}, {where} и @id указываются для конкретного случая, в зависимости от того какая информация нужна для выборки.

2. Простой запрос на выборку по логину и паролю.

Код запроса на языке SQL: SELECT {what} FROM {from} where u\_username = @login and u\_password = @pass;

Данный запрос делает выборку нужной информации из нужного места основываясь на логине и пароле пользователя. Данные {what}, {from}, @login и @pass указываются для конкретного случая, в зависимости от того какая информация нужна для выборки.

3. Запрос на добавление нового пользователя в базу данных.

Код запроса на языке SQL: INSERT INTO `users` (`u\_username`, `u\_password`, `u\_name`, `u\_surname`, `u\_date\_of\_birth`) VALUES (@username, @pass, @name, @surname, @dob);

Данный запрос добавляет нового пользователя в базу данных после успешной регистрации. Параметры @username, @pass, @name, @surname, @dob заполняются пользователем при регистрации в приложении. Каждому пользователю присваивается уникальный id.

4. Запрос на создание дебетовой карты.

Код запроса на языке SQL: INSERT INTO `e\_carts` (`ec\_cartnumber`, `ec\_cache`, `ec\_cvv`, `ec\_cartholder\_id`) VALUES (@cardnumber, @cache, @cvv, @id) ;

Данный запрос добавляет новую дебетовую карту в базу данных. Параметры @cardnumber, @cache, @cvv определяют данные дебетовой карты. Параметр @id является id пользователя и связывает карту с данными пользователя в таблице users.

5. Запрос на изменение данных о владении пользователем дебетовой карты нашего банка.

Код запроса на языке SQL: UPDATE `users` SET `u\_havecart` = 1 where `u\_id` = @id;

Данный запрос изменяет параметр владения дебетовой карты пользователем в базе данных, что в последствии помогает приложению определять это автоматически. Параметр @id меняется в зависимости от id пользователя.

6. Запрос на добавление заявки о кредите.

Код запроса на языке SQL: INSERT INTO `credit\_requests` (`cr\_description`, `cr\_value`, `cr\_email`, `cr\_id\_user`) VALUES (@desc, @value, @email, @id);

Данный запрос добавляет новую заявку на кредитную карту в базу данных. Параметры @desc, @value, @email заполняются пользователем через приложение. Параметр @id является id пользователя, подающего заявку, он же связывает заявку с пользователем в таблице users.

7. Запрос на удаление заявки на кредитную карту из базы данных.

Код запроса на языке SQL: DELETE FROM `credit\_requests` where cr\_id\_user = @id;

Когда администратор одобряет или отклоняет заявку на кредитную карту она автоматически удаляется из базы данных. Параметром @id является id пользователя, подавшего заявку.

8. Запрос на изменение данных о владении пользователем кредитной карты нашего банка.

Код запроса на языке SQL: UPDATE `users` SET `u\_havecredit` = 1 where `u\_id` = @id;

Данный запрос изменяет параметр владения кредитной карты пользователем в базе данных, что в последствии помогает приложению определять это автоматически. Параметр @id меняется в зависимости от id пользователя.

9. Запрос на удаление кредитной карты из базы данных.

Код запроса на языке SQL: DELETE FROM `credit\_cards` where cr\_c\_id\_user= @id;

Когда пользователь решает закрыть кредитную карту банка, выполняется этот запрос, и его карта удаляется из базы данных. Параметр @id меняется в зависимости от id пользователя.

10. Запрос на перевод денег на счет другого пользователя в банке.

UPDATE e\_carts SET ec\_cache=ec\_cache+@value where ec\_cartnumber = @cardnumber

Запрос добавляет к полю денег определенного пользователя параметр @value который является количеством денег, которое хочет перевести пользователь на другой счет. Параметр @cardnumber является номером карты, на который переводятся деньги.

UPDATE e\_carts SET ec\_cache=ec\_cache-@value where ec\_cartnumber = @cardnumber;

Запрос убавляет из поля денег определенного пользователя параметр @value который является количеством денег, которое хочет перевести пользователь на другой счет. Параметр @cardnumber является номером карты, с которого переводятся деньги.

### **1.1.5 Список ограничений**

Ограничения представляют собой набор некоторых условий, налагаемых на элементы базы данных (таблицы, столбцы и т.д.) или всю базу данных, гарантирующие, что информация будет подчиняться определенным правилам целостности данных.

В данном курсовом проекте было использовано ограничение ссылочной целостности, т. к. значения одних столбцов таблиц связаны со значениями других столбцов в другой таблице. В каждой из таблиц проектируемой базы данных использовались первичный и внешний ключи, содержащие уникальные значения столбцов. Благодаря обеспечению ссылочной целостности данных была исключена возможность дублирования записей в базе данных, обеспечено каскадное обновление, вставка и удаление записей БД.

## **1.2 Выбор и описание используемой среды разработки**

### **1.2.1 Описание используемой среды разработки базы данных**

Для создания базы данных системы расчета с клиентами в банке была использована СУБД MySQL. MySQL – это свободная реляционная система управления базами данных, принадлежащая компании Oracle, которая использует язык SQL. SQL – язык программирования использующийся для создания, изменения и управления данными в реляционных базах данных, которые находятся в соответствующей СУБД. Для визуальной наглядности во время разработки базы данных было использовано веб-приложение phpMyAdmin (см. рисунок 1), которое позволяет наблюдать за изменениями, внесенными в БД. Так же phpMyAdmin позволяет управлять сервером MySQL и запускать SQL команды. Была выбрано именно эта реализация управления БД, потому что она показалась наиболее удобной для разработки.

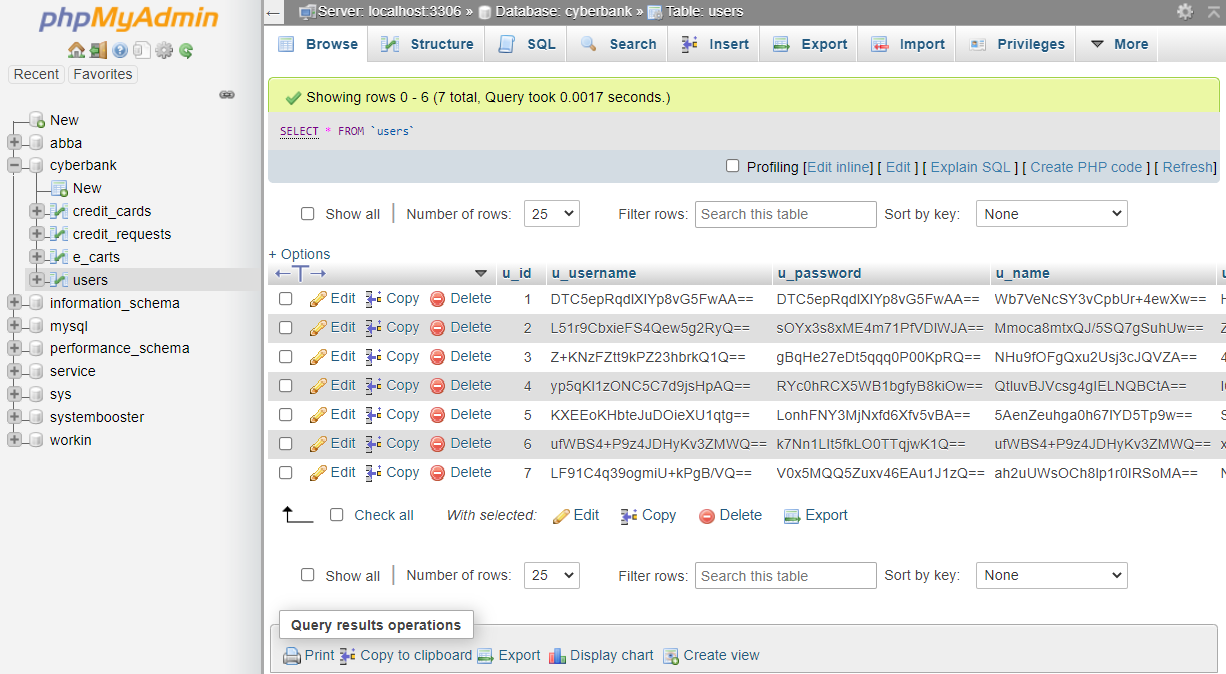


Рисунок 1. Внешний вид phpMyAdmin

### **1.2.2 Описание используемой среды разработки приложения**

Для того, чтобы пользователям и администраторам было проще и удобнее работать с базой данных было принято решение о создания приложения. Был разработан уникальный, интуитивно понятный и современный интерфейс приложения, с использованием основных цветов компании (оранжевый, черный). Вид интерфейса основывается на предпочтениях пользователей. Для разработки приложения были использованы несколько языков программирования:

* C# - 98.6%
* Python – 1.3%
* Visual Basic Script – 0,1%

Большая часть приложения написана на С#. Это объектно-ориентированный язык программирования, разрабатывающийся компанией Microsoft. С# относится к языкам семейства C. С# был выбран из-за того, что он обладает некоторыми удобными функциями для разработки пользовательских интерфейсов для приложений.

Весь интерфейс приложения написан на WPF (Windows Presentation Foundation). Это система для разработки пользовательских приложений с красивыми визуальными возможностями и использует язык разметки XAML. Из-за этого и было решено разрабатывать приложение именно на этой системе, т.к. она позволяет сделать наиболее привлекательный и удобный для пользователя. Она входит в состав .NET Framework, который является одним из главных расширений для С#.

Для комфортной и удобной разработки приложения была выбрана интегрированная среда разработки Microsoft Visual Studio 2019. Visual Studio включает в себя редактор исходного кода и возможность рефакторинга. Так же упрощает работу с классами, что было очень полезно при разработке приложения. С помощью встроенного отладчика было очень удобно проверять работоспособность приложения сразу после написания какой-либо новой функции. Так же благодаря встроенному диспетчеру пакетов NuGet очень удобно устанавливать плагины для разработки, которые были очень полезны. К тому же Visual Studio обладает привлекательным интерфейсом (см. рисунок 2), который делает разработку приложения не только проще, но и эстетически приятнее.

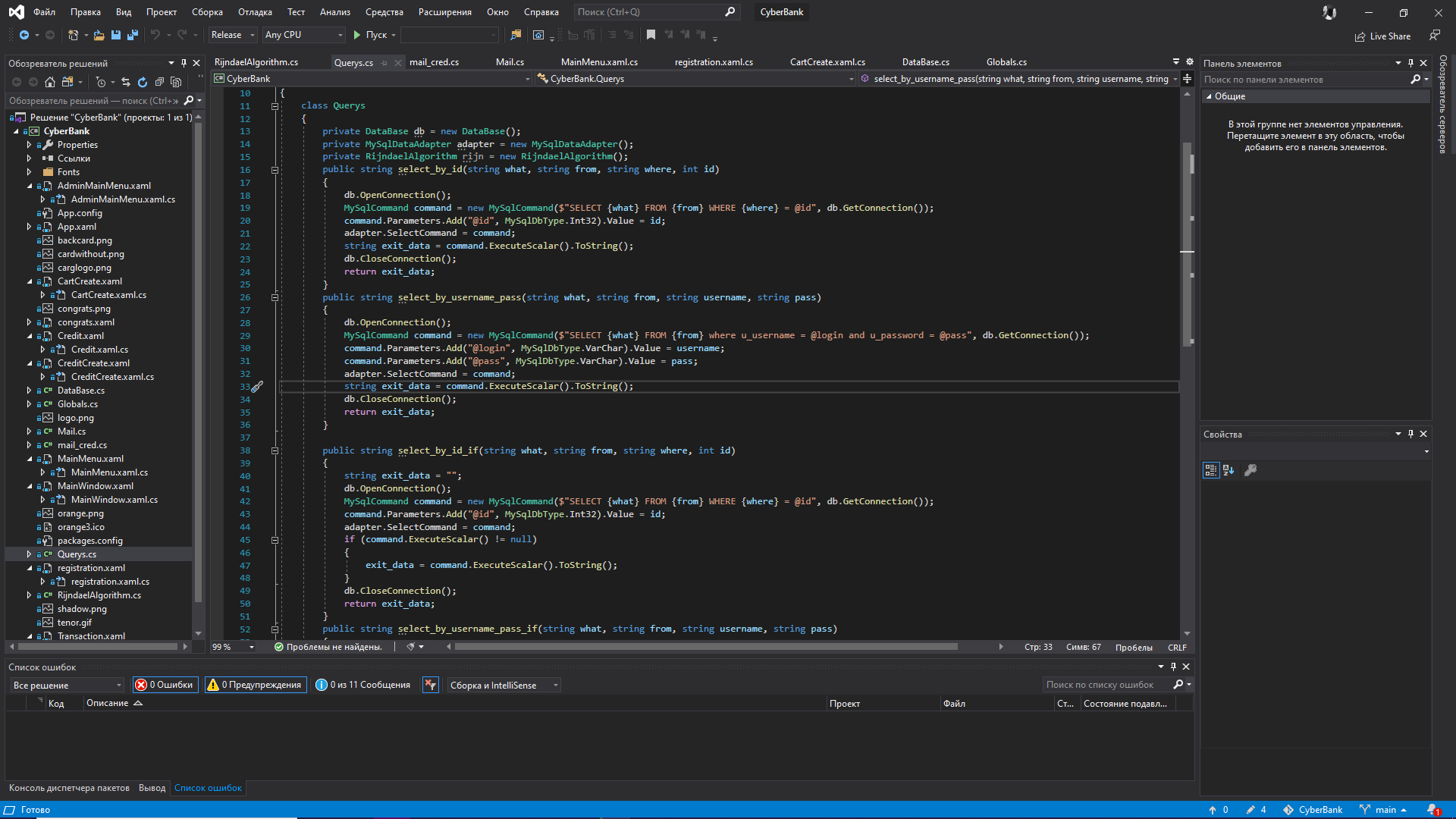


Рисунок 2. Внешний вид Microsoft Visual Studio 2019

Для небольшой части приложения был использован язык программирования Python. Python – это язык программирования высокого уровня разрабатываемый компанией Python Software Foundation. Язык полностью объектно-ориентированный, и обладает минималистичным синтаксисом ядра языка. В основном он используется для написания скриптов, как и было в случае разработки этого приложения.

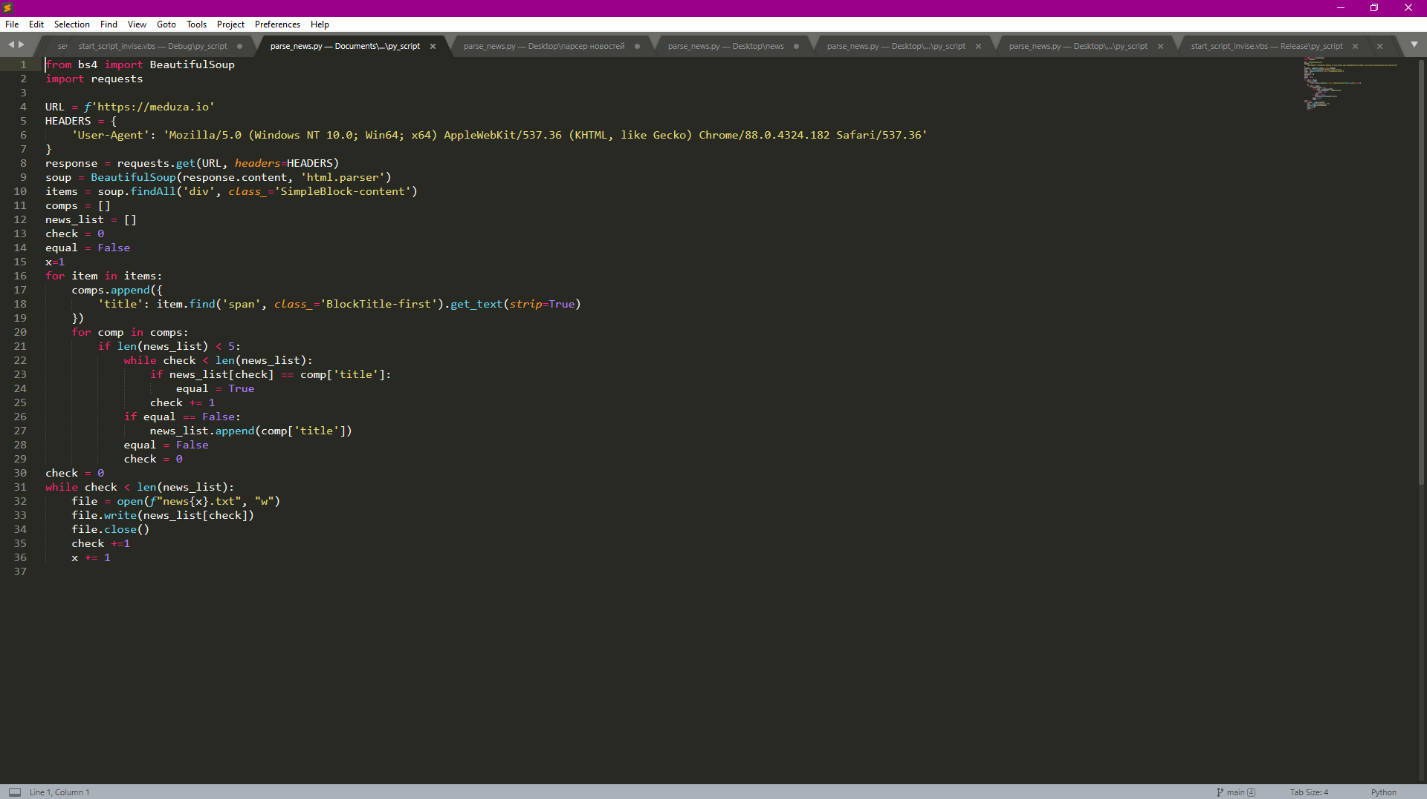
Для написания скрипта на этом языке был использован редактор кода Sublime Text, который обладает минималистичным интерфейсом (см. рисунок 3) и вполне удобен для редактирования кода и написания небольших скриптов. 

Рисунок 3. Внешний вид Sublime Text

Язык сценариев Visual Basic Script использовался для написания небольшого скрипта. Он разработан компанией Microsoft и основывается на языке программирования Visual Basic.

Для написания скрипта также был использован редактор кода Sublime Text.

## **1.3 Инфологическая модель (ИЛМ)**

Er-модель базы данных необходима для того чтобы концептуально описать предметную область базы данных и применяется при проектировании оной, при необходимости выделить ключевые сущности и связи между ними. Благодаря данной модели людям, которые работают над разработкой базы данных будет проще понять принципы её работы.

Приведенная ниже (см. рисунок 4) er-модель описывает 4 сущности базы данных, их атрибуты и связи между ними.

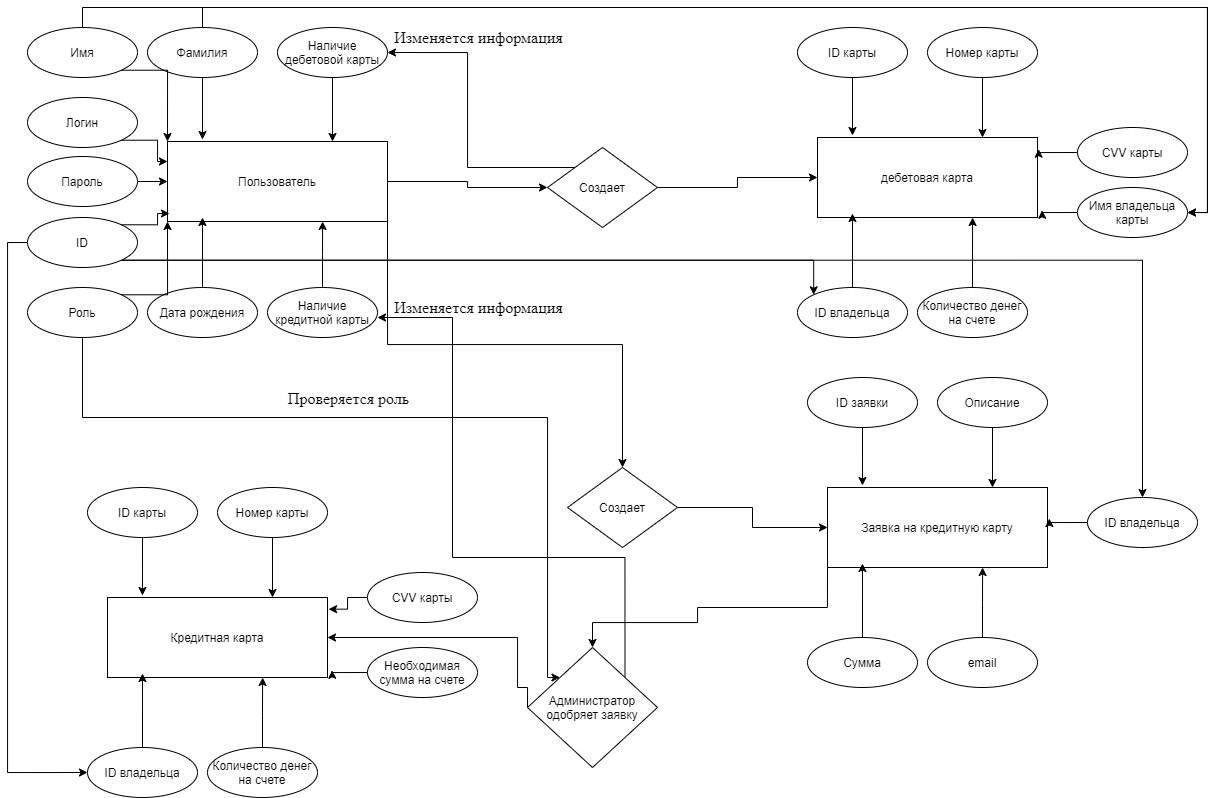
****

Рисунок 4. ER-модель базы данных CyberBank

## **1.4 Даталогическая модель**

### **1.4.1 Нормализованная реляционная модель**

На рисунке 5 можно наблюдать модель базы данных CyberBank. Она состоит из 4 таблиц, одна из которых, таблица users, содержит все данные и id пользователей, а остальные три таблицы связаны внешними ключами с пользовательскими id в таблице users.

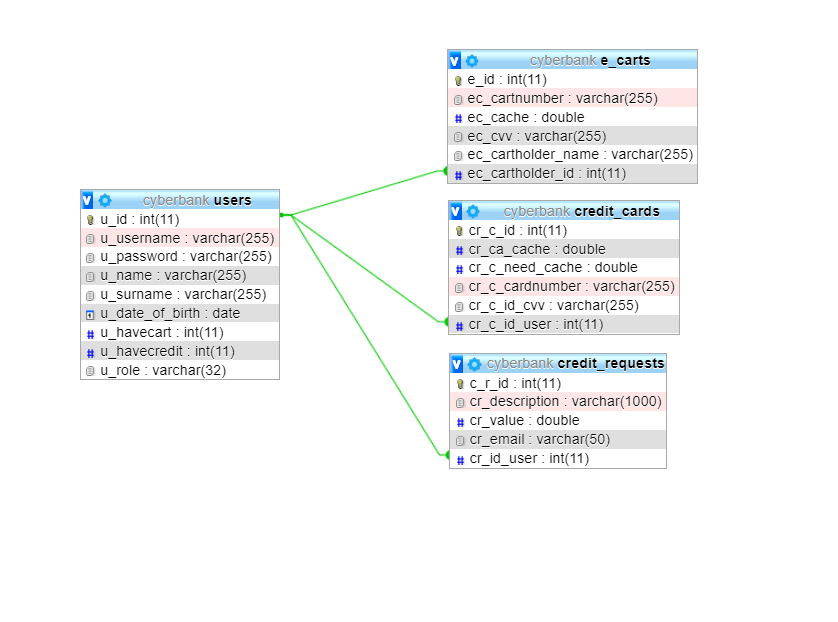
****

Рисунок 5. Модель базы данных CyberBank

### **1.4.2 Состав таблиц базы данных**

Пройдемся по таблицам, которые используются в базе данных.

|  |  |  |
| --- | --- | --- |
| Наименование атрибутов | Тип полей | null |
| u\_id | int (11) | нет |
| u\_username | varchar (255) | нет |
| u\_password | varchar (255) | нет |
| u\_name | varchar (255) | нет |
| u\_surname | varchar (255) | нет |
| u\_date\_of\_birth | date | нет |
| u\_havecart | int (11) | да |
| u\_havecredit | int (11) | нет |
| u\_role | varchar (32) | да |

Таблица 1. Состав таблицы users

Ключи таблицы:

- u\_id - первичный ключ;

|  |  |  |
| --- | --- | --- |
| Наименование атрибутов | Тип полей | null |
| e\_id | int (11) | нет |
| ec\_cartnumber | varchar (255) | нет |
| ec\_cache | double | да |
| ec\_cvv | varchar (255) | нет |
| ec\_cartholder\_name | varchar (255) | нет |
| ec\_cartholder\_id | int (11) | нет |

Таблица 2. Состав таблицы e\_carts

Ключи таблицы:

- e\_id - первичный ключ;

- ec\_cartholder\_id - внешний ключ;

|  |  |  |
| --- | --- | --- |
| Наименование атрибутов | Тип полей | null |
| c\_r\_id | int (11) | нет |
| cr\_description | varchar (1000) | нет |
| cr\_value | double | нет |
| cr\_email | varchar (50) | нет |
| cr\_id\_user | int (11) | нет |

Таблица 3. Состав таблицы credit\_requests

Ключи таблицы:

- c\_r\_id - первичный ключ;

- cr\_id\_user - внешний ключ;

|  |  |  |
| --- | --- | --- |
| Наименование атрибутов | Тип полей | null |
| cr\_c\_id | int (11) | нет |
| cr\_ca\_cache | double | нет |
| cr\_ca\_need\_cache | double | нет |
| cr\_c\_cardnumber | varchar (255) | нет |
| cr\_c\_cvv | varchar (255) | нет |
| cr\_c\_id\_user | int (11) | нет |

Таблица 4. Состав таблицы credit\_cards

Ключи таблицы:

- cr\_r\_id - первичный ключ;

- cr\_c\_id\_user - внешний ключ;

# **Раздел 2. Разработка базы данных и приложения**

## **2.1 Разработка серверной части (базы данных)**

### **2.1.1 Таблицы, связи между таблицами, индексы**

Всего в базе данных используются четыре таблицы, которые имеют связи между собой, первичные ключи и индексы. Всё это необходимо для удобно обращения с базой данных, что в последствии уменьшает задержки при разработке и эксплуатации БД.

Первая таблица – users (см. таблица 1). В неё загружаются все основные данные аккаунта пользователя, которые в последствии будут там храниться и использоваться банком для оформления карт.

В этой таблице присутствует 9 атрибутов.

Первый атрибут u\_id — это пользовательский ID, который хранится в таблице в типе данных Int, то есть в целочисленных числах. Данный атрибут обладает свойством автоматической инкрементации, что означает, что он будет увеличиваться на один с появлением каждого нового пользователя и будет оставаться уникальным всегда. Также этот атрибут является первичным ключом таблицы.

Второй атрибут u\_username — логин пользователя. Он хранится в типе данных VarChar, который является хранилищем для текстовой информации. Любые символы помещенные внутрь будут преобразованы в формат текста. Максимальная длина логина – 255 символов.

Третий атрибут u\_password — пароль пользователя. Он также принимает тип данных VarChar и имеет такую же максимальную длинну.

Четвертый атрибут u\_name — имя пользователя. Тоже VarChar. Длинна тоже 255 символов.

Пятый атрибут u\_surname — фамилия пользователя. Тоже VarChar. Длинна тоже 255 символов.

Шестой атрибут u\_date\_of\_birth — дата рождения пользователя. Атрибут хранится в типе данных date, который служит для того, чтоб хранить даты.

Седьмой атрибут u\_havecard – это атрибут для хранения данных о том, имеет ли пользователь дебетовую карту. Если она у пользователя имеется, то этот атрибут принимает значение 1, если она отсутствует, то 0. В последствии благодаря этому приложение определяет наличие карты и изменяет свое поведение относительно конкретного пользователя, а также администратор видит эти данные и знает то, есть ли у пользователя карта. Для хранения используется тип int.

Восьмой атрибут u\_havecredit – это данные о наличии кредитной карты. Атрибут также принимает значения 1 или 0 в зависимости от ее наличия и выполняет те же самую функцию помощи приложению и администратору определить наличие карты у пользователя. Тип данных атрибута – int.

Девятый атрибут u\_role – это роль пользователя. В случае если пользователь является администраторам она будет хранить текст “admin”. Если же пользователь им не является атрибут будет равен null. Благодаря этому приложение автоматически определяет какая роль у пользователя и ограничивает, либо разрешает ему определенные функции.

Вторая таблица — это таблица e\_carts (см. таблица 2) в которой хранятся данные о дебетовых картах пользователей, которые туда автоматически загружаются в тот момент, когда пользователь регистрирует новую дебетовую карту. В таблице находятся 6 атрибутов.

Первый атрибут e\_id – это id дебетовой карты. Он является уникальным первичным ключом с автоматической инкрементацией и хранится в типе данных int.

Второй атрибут ec\_cartnumber– это номер дебетовой карты. Он генерируется автоматически с использованием случайного выбора цифр, для того чтобы номера были уникальными. Он хранится в базе данных в типе VarChar.

Третий атрибут ec\_cache– это количество денег на счете дебетовой карты. Для его хранения используется тип данных double, который позволяет хранить большие числа с плавающей точкой.

Четвертый атрибут ec\_cvv – это CVV номер карты. Он генерируется случайным образом и хранится в типе VarChar.

Пятый атрибут ec\_cartholder\_name – это полное имя владельца карты заглавными буквами. Он необходим для того, чтобы потом напечатать его на карте. Хранится в типе данных VarChar.

Шестой атрибут ec\_cartholder\_id– это ID владельца карты. Этот атрибут является внешним ключом и образует связь между таблицей e\_carts и таблицей users, необходимую для того, чтобы было проще определять какая карта у какого пользователя.

Третья таблица credit\_requests (см. Таблица 3) – это таблица, которая хранит в себе данные о заявках на кредитную карту, которые автоматически заносятся туда после того, как пользователь оставит заявку через приложение. Содержит в себе пять атрибутов.

Первый атрибут c\_r\_id – это ID запроса. Он является числом в типе данных int и является первичным ключом и обладает свойством автоматической инкрементации.

Второй атрибут cr\_description – это описание заявки. Оно составляется пользователем в приложении, а затем автоматически заносится в базу данных. Хранится в типе данных VarChar.

Третий атрибут cr\_value – это сумма, которую пользователь хочет иметь на своей кредитной карте. Она описывается пользователем в приложении и загружается в базу данных автоматически. Хранится в типе данных double.

Четвертый атрибутcr\_email – это адрес электронной почты пользователя. Он оставляется пользователем в приложении, а затем заносится в базу данных для того, чтобы в дальнейшем, при одобрении заявки, было отправлено письмо на эту почту.

Пятый атрибут cr\_id\_user – это ID пользователя, который связывает эту таблицу с таблицей users. Является внешним ключом.

Четвертая таблица — это таблица credit\_cards. В ней хранятся данные по кредитным картам пользователей. Она содержит 6 атрибутов

Первый атрибут cr\_c\_id – это id кредитной карты. Он является уникальным первичным ключом с автоматической инкрементацией и хранится в типе данных int.

Второй атрибут cr\_ca\_cache– это количество денег на счете кредитной карты. Для его хранения используется тип данных double, который позволяет хранить большие числа с плавающей точкой.

Третий атрибут cr\_ca\_need\_cache– это количество денег, которое необходимо иметь пользователю на счете кредитной карты для того, чтобы закрыть её. Для его хранения используется тип данных double, который позволяет хранить большие числа с плавающей точкой.

Четвертый атрибут cr\_c\_cardnumber– это номер кредитной карты. Он генерируется автоматически с использованием случайного выбора цифр, для того чтобы номера были уникальными. Он хранится в базе данных в типе VarChar.

Пятый атрибут cr\_c\_id\_cvv – это CVV номер карты. Он генерируется случайным образом и хранится в типе VarChar.

Шестой атрибут cr\_cr\_id\_user– это ID владельца карты. Этот атрибут является внешним ключом и образует связь между таблицей credit\_cards и таблицей users, необходимую для того, чтобы было проще определять какая карта у какого пользователя.

## **2.2 Разработка клиентской части**

### **2.2.1 Главная (управляющая) форма, форма регистрации, авторизации, запрос пароля (разграничение доступа)**

Приложение разрабатывалось для связки базы данных банка и может использоваться как администраторами, так и простыми пользователями, но у пользователей, конечно же, отсутствуют некоторые возможности доступные только администраторам.

Для того, чтобы приложение было более удобным, было принято решение добавить некоторые полезные функции, которые могут улучшить пользовательский опыт клиентов. Была добавлена функция показа курса валют, т.к. это очень важная вещь в банковской сфере. Продвинутый пользователь всегда хочет знать точные котировки валют, которые берутся из интернета постоянно и обновляются каждую минуту даже во время работы приложения. На рисунках 4, 5 и 6 в правом верхнем углу можно наблюдать это.

Главная форма приложения представляет собой меню приложения (см. рисунок 4), которое позволяет пользователю выбрать, что он хочет сделать.

В колонке в левой части экрана (см. рисунок 4) приложения сверху есть кнопка “Вернуться на главную” и кнопка “Выйти из аккаунта”. Под ними указывается имя пользователя, в случае если он авторизовался. Под именем указывается роль пользователя, если пользователь является администратором, то там будет написано admin, а в случае, если это обычный пользователь, там не будет написано ничего. Далее внизу указываются данные дебетовой карты пользователя: CVV код, номер карты, а также количество денег на счете. Еще ниже отображаются те же самые данные, но по кредитной карте, в случае если таковая имеется у пользователя.

В центре экрана (см. рисунок 4) можно заметить кнопки перехода на другие формы. Кнопка “Перевод” ведёт на форму перевода денег на счета других пользователей, кнопка “администрирование” ведёт на форму администрирования и доступна только в том случае, если пользователь является администратором, если же пользователь им не является, то эта кнопка будет отсутствовать. Кнопка “Кредит” отправляет пользователя на форму оформления кредитной карты, если она отсутствует, а ежели она присутствует, то будет открыта форма управления кредитной картой.

В правой части главной формы (см. рисунок 4) показывается визуальное представления дебетовой карты пользователя, со стилистикой банка и данными кредитной карты, которую можно перевернуть и посмотреть на неё с задней стороны.

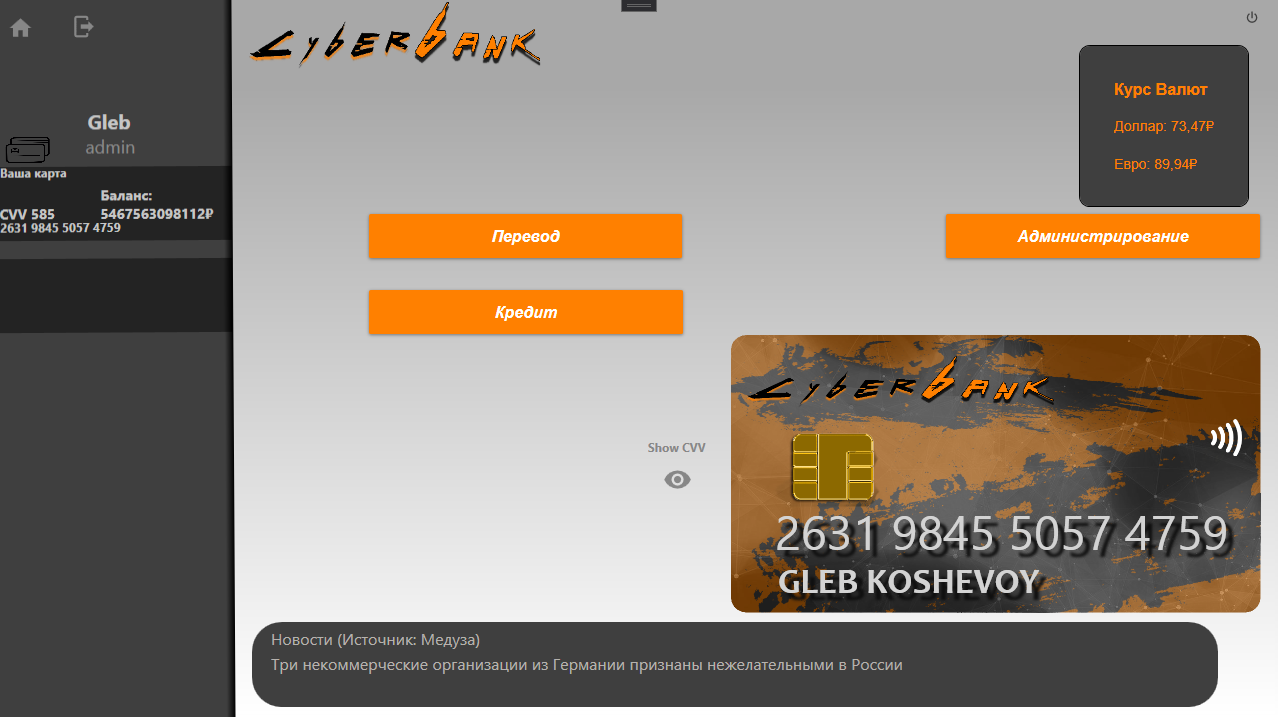
Внизу экрана (см. рисунок 6) отображается новостная строка, которая позволяет пользователям оставаться в курсе последних событий во время того, когда они управляют своими денежными сбережениями. Новости берутся из новостного издания прямо во время работы приложения, что позволяет им оставаться актуальными всегда.****

Рисунок 6. Главная форма CyberBank

Далее представлена форма регистрации пользователей (см. рисунок 7). На ней, в центре экрана, расположены поля для ввода данных пользователя, которые будут использоваться в его персональном аккаунте и перенесены в базу данных в зашифрованном виде.

Всего при регистрации необходимо ввести пять пунктов: логин, пароль, имя, фамилия и дата рождения. Затем будет необходимо нажать кнопку “Регистрация” для выполнения данного действия. Если все данные соответствуют необходимому виду для занесения в базу данных, то регистрация пройдет успешно. Если какое-то из полей заполнено неверно или произошла ошибка при связи с базой данных, то выведется сообщение о том, что регистрация не была выполнена.

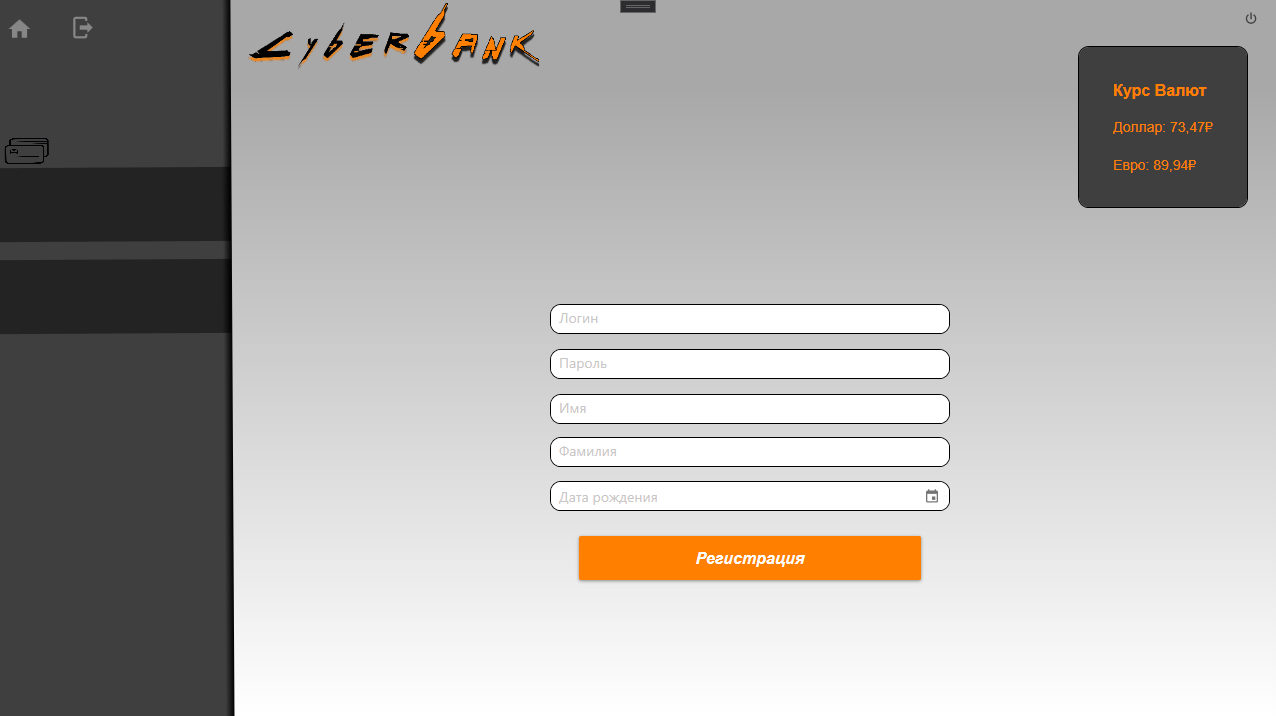
****

Рисунок 7. Форма регистрации CyberBank

После успешной регистрации пользователь автоматически переходит на страницу оформления дебетовой карты. Без наличия карты пользователь не сможет пользоваться приложением до тех пор, пока не оформит её. При оформлении дебетовой карты каждый пользователь совершенно бесплатно участвует в лотерее, в которой он имеет шанс выиграть до тысячи рублей на свой счет дебетовой карты. Это сделано для привлечения новых пользователей.

Для того что бы оформить дебетовую карту необходимо ввести свой логин и пароль в поля (см. рисунок 8) “Логин” и “Пароль”, а затем нажать кнопку “Оформить”. В случае если пользователь с таким логином и паролем существует в базе данных всё пройдет успешно и клиент сможет полноценно пользоваться всеми доступными ему функциями приложения.

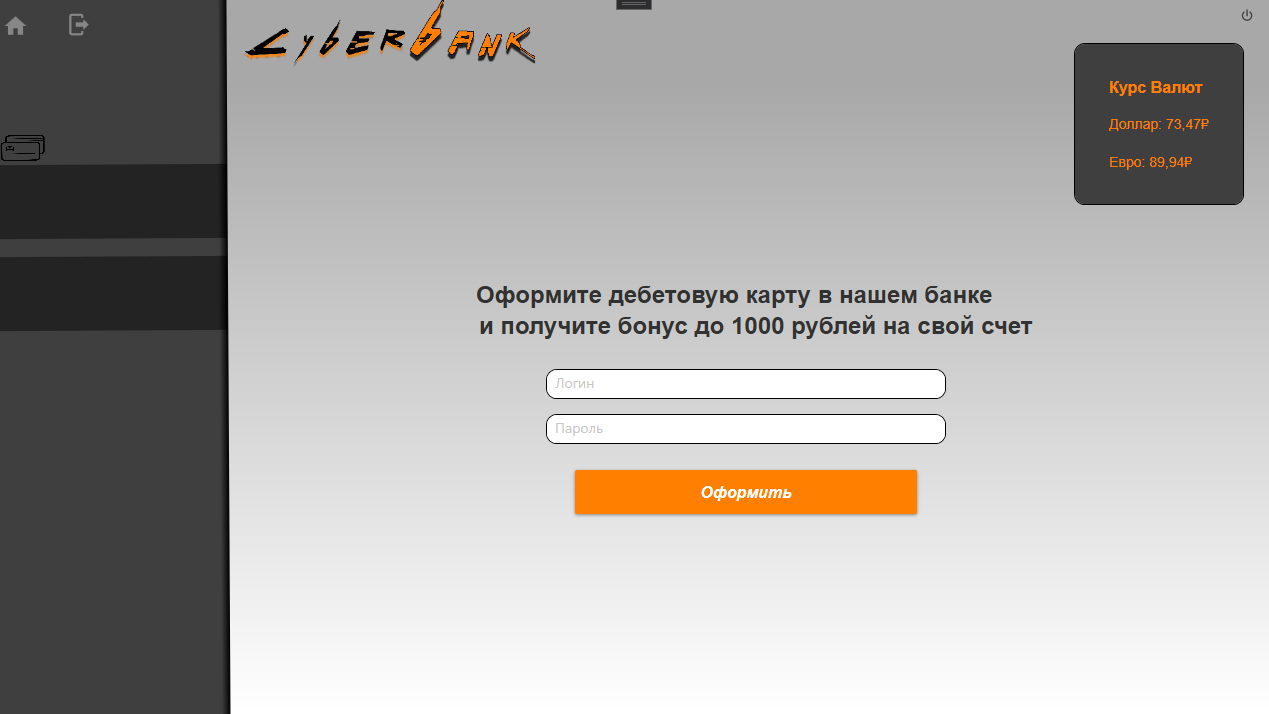


Рисунок 8. Форма оформления дебетовой карты CyberBank

Еще необходимо рассказать о форме авторизации (см. рисунок 9), которая во многом напоминает форму оформления дебетовой карты. В центре экрана отображены поля для ввода логина и пароля пользователя, а под ними располагается кнопка “Войти”. После нажатия этой кнопки происходит проверка на наличие пользователя с такими данными, а также на наличие связи с базой данных. Если пользователь и связь есть, то авторизация проходит успешно и приложение переносит клиента в главную форму, где он спокойно может распоряжаться своими счетами.

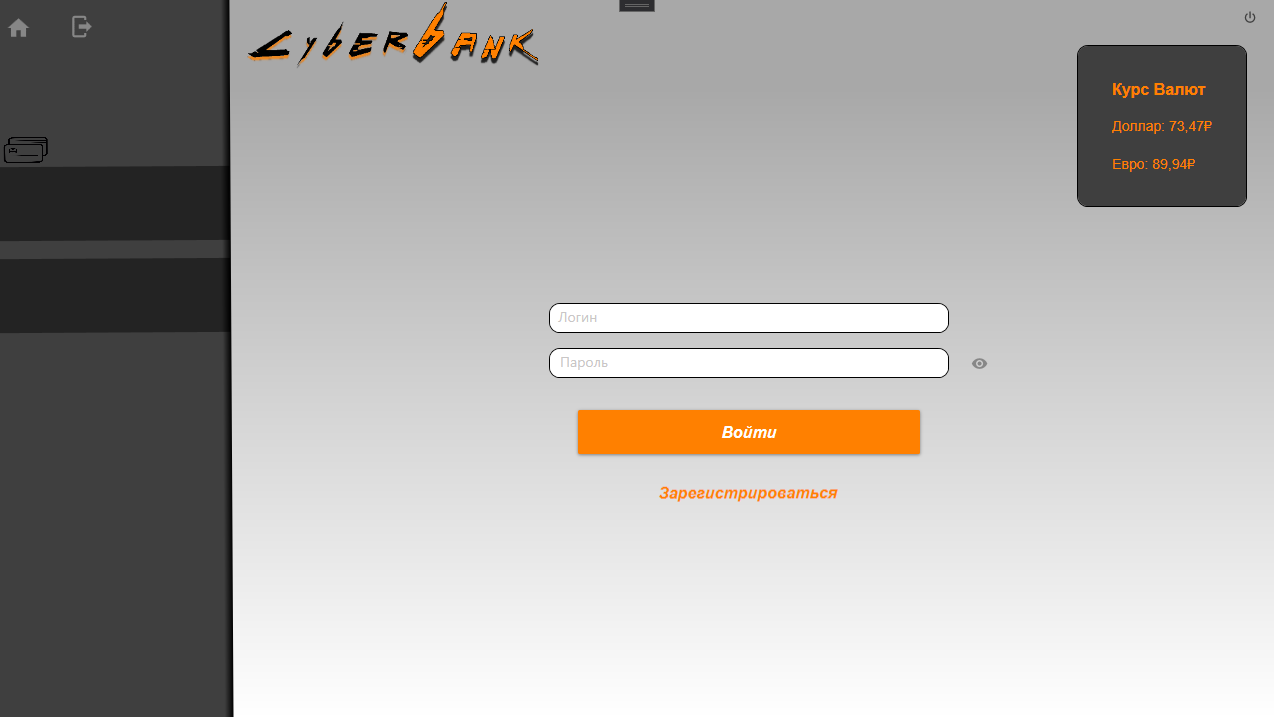
****

Рисунок 9. Форма авторизации CyberBank

Также есть возможность перевода денег с дебетовой карты на счет другого пользователя (см. рисунок 10). Для этого пользователю необходимо нажать кнопку “Перевод” в главной форме. После этого он попадет на меню перевода. Здесь есть два поля, которые необходимо заполнить, а под ними располагается кнопка “Перевести”, после нажатия на которую выполнится перевод, но только в том случае если данные, которые ввел пользователь верны. Пользователь должен указать в первом поле логин или номер карты того пользователя, которому он хочет перевести деньги, а во втором поле необходимо ввести сумму перевода. При нажатии кнопки происходит проверка на существование пользователя с указанным логином или номером карты в базе данных, а также проверка на то, достаточно ли денег у пользователя для выполнения этого перевода.

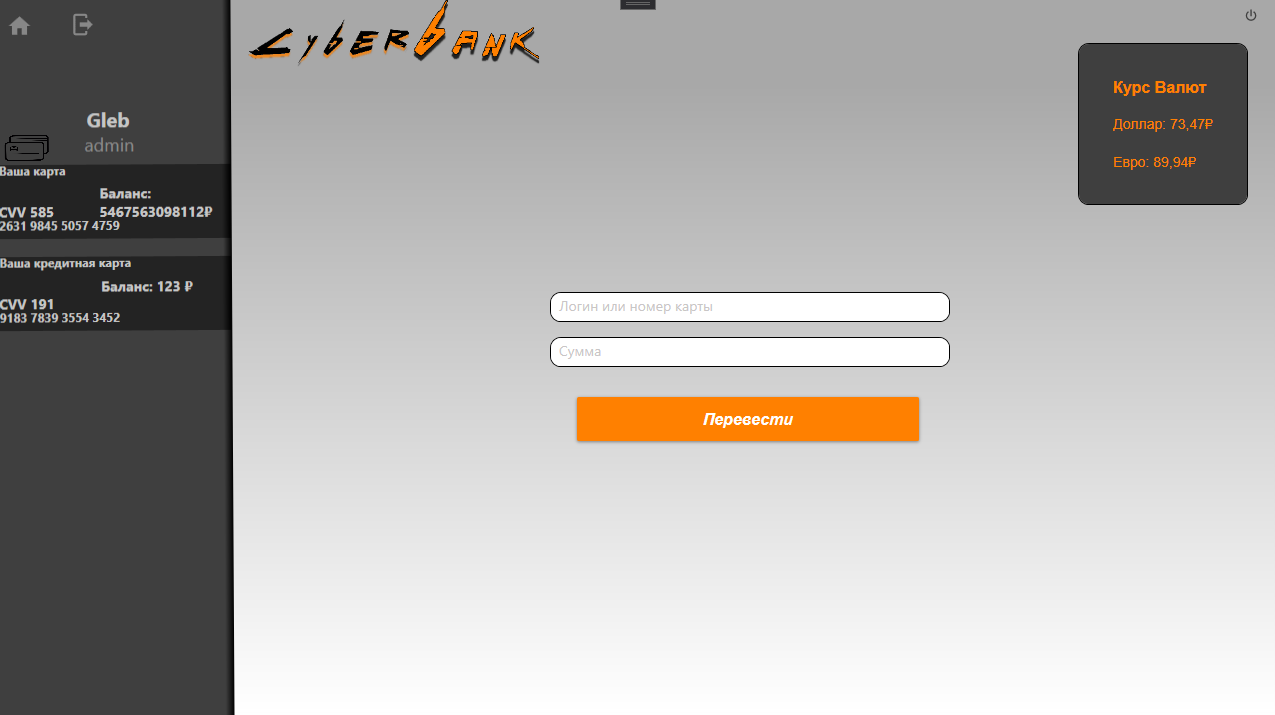


Рисунок 10. Форма перевода CyberBank

Также пользователи могут оформить заявку на получение кредитной карты CyberBank. Для этого необходимо нажать на кнопку “Кредит” в главном меню приложения, а затем заполнить необходимую информацию в появившемся меню (см. рисунок 11) оформления кредитной карты.

В это меню располагаются три поля для ввода информации и кнопка “Оформить”. Необходимо указать цель получения кредитной карты в первом поле, желаемую сумму, в рублях, на ней во втором, а также оставить свой адрес электронной почты в третьем, чтобы в последствии получить на него уведомление о решении банка по поводу заявки. После того как все поля заполнены нажатие кнопки “Оформить” отправит эти данные в базу данных, где заявка будет храниться до тех пор, пока её не рассмотрит администратор и не решит, что делать.

После одобрения заявки будет создана кредитная карта пользователя и ее данные будут отображаться в главном меню приложения, а также кнопка “Кредит” будет вести не на меню подачи заявки на кредитную карту, а в меню управления кредитной картой.

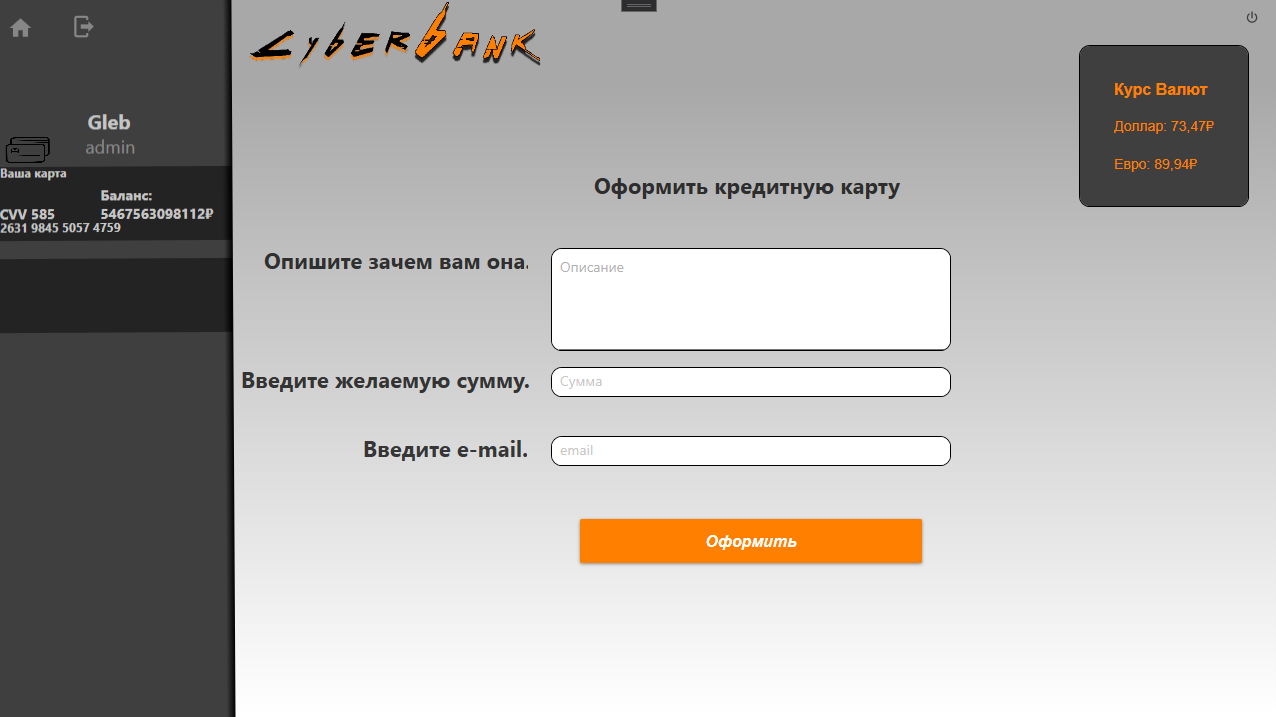


Рисунок 11. Форма подачи заявки на кредитную карту CyberBank

Меню управления кредитной картой (см. рисунок 12) позволяет пользователю закрыть кредитную карту в случае, если на ней достаточно денег для этого. В нём написано количество денег на кредитной карте пользователя и количество денег, которое необходимо для того, чтобы банк позволил закрыть карту. Снизу находится кнопка “Закрыть”, после нажатия на которую идет проверка на то, действительно ли денег больше или равно той сумме, которая необходима для закрытия. Если всё верно, то карта удаляется из базы данных. После этого пользователь может оформить новую, если ему понадобится.

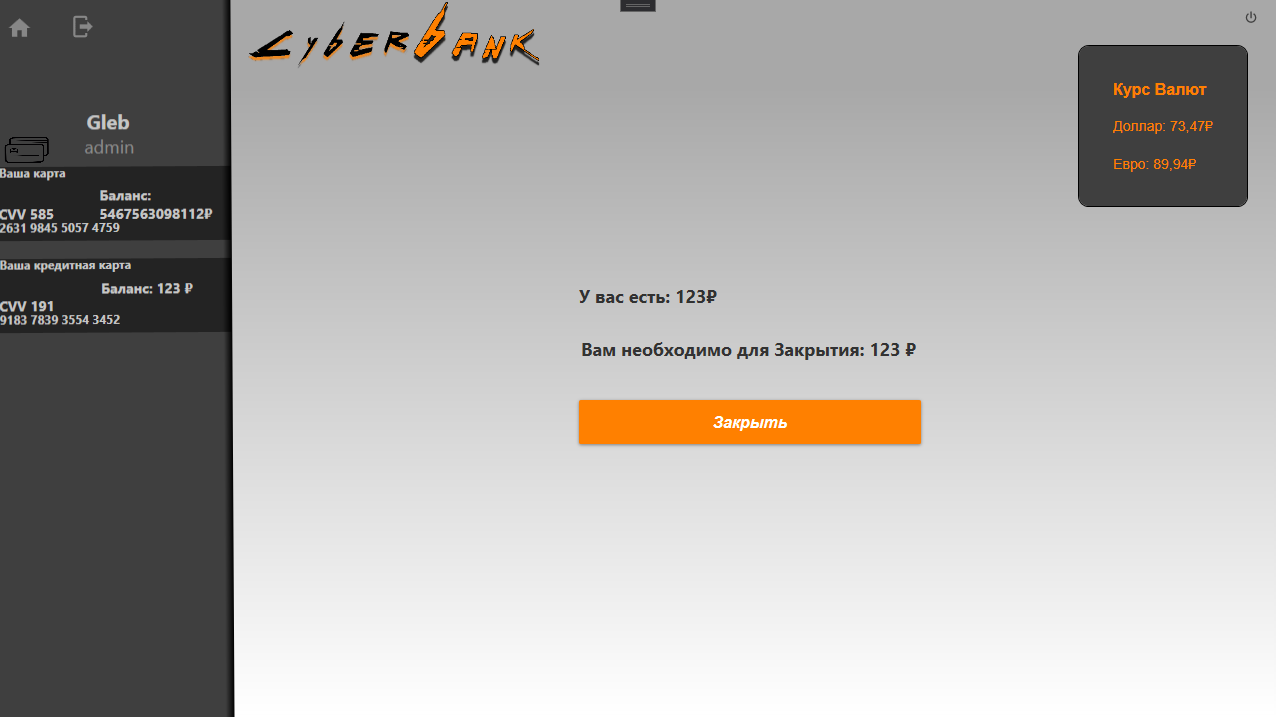


Рисунок 12. Форма управления кредитной картой CyberBank

### **2.2.2 Формы поиска, визуализации запросов**

Теперь необходимо рассказать о функциях доступных администратору. Администратор может просматривать список всех пользователей банка и видеть их данные, которые не являются строго конфиденциальными, то есть всё кроме пароля. Это список открыт по умолчанию в меню администрирования, но также можно его открыть по кнопке “Пользователи” в верхней части экрана. Это сделано для того, чтобы администраторы не могли воспользоваться ими в корыстных целях или в том случае, если аккаунт администратора окажется в руках злоумышленника.

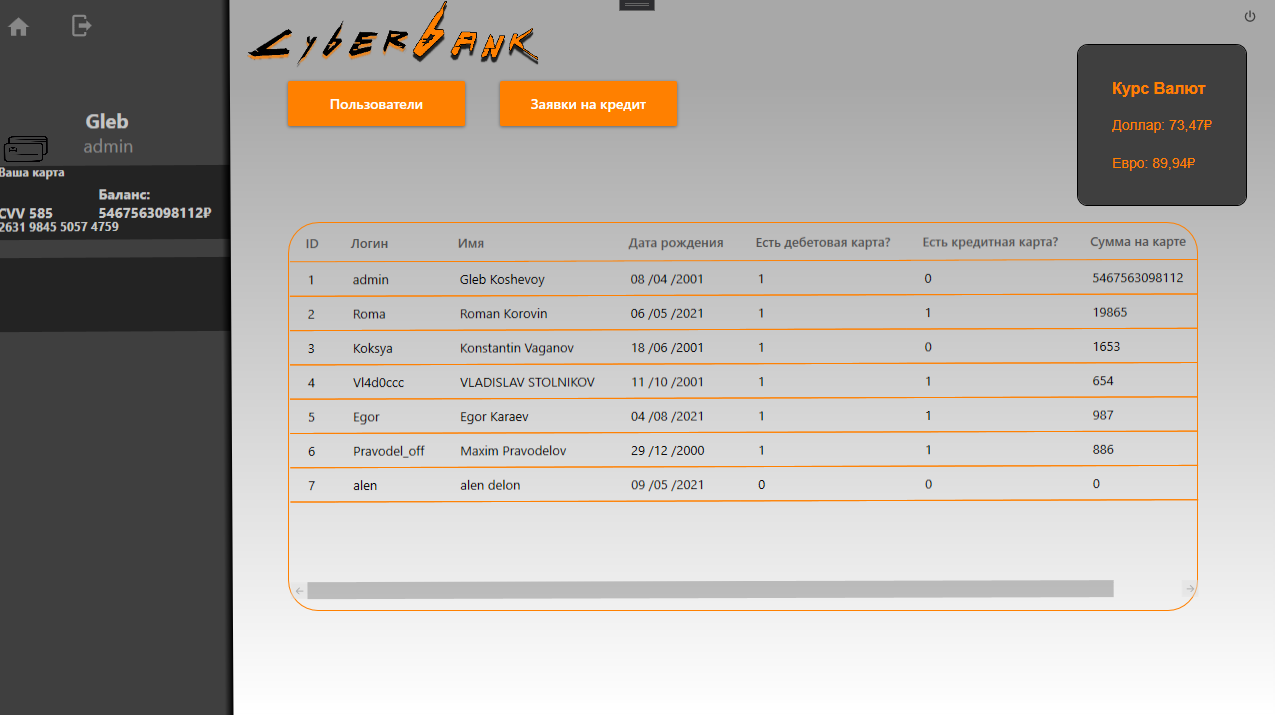
Администратор может видеть (см. рисунок 13) id пользователя, его логин, имя, фамилию, дату рождения, имеет ли пользователь дебетовую и кредитную карты, а также количество денег на дебетовой карте пользователя и его роль. Так же администратор может сортировать список пользователей.****

Рисунок 13. Список пользователей CyberBank

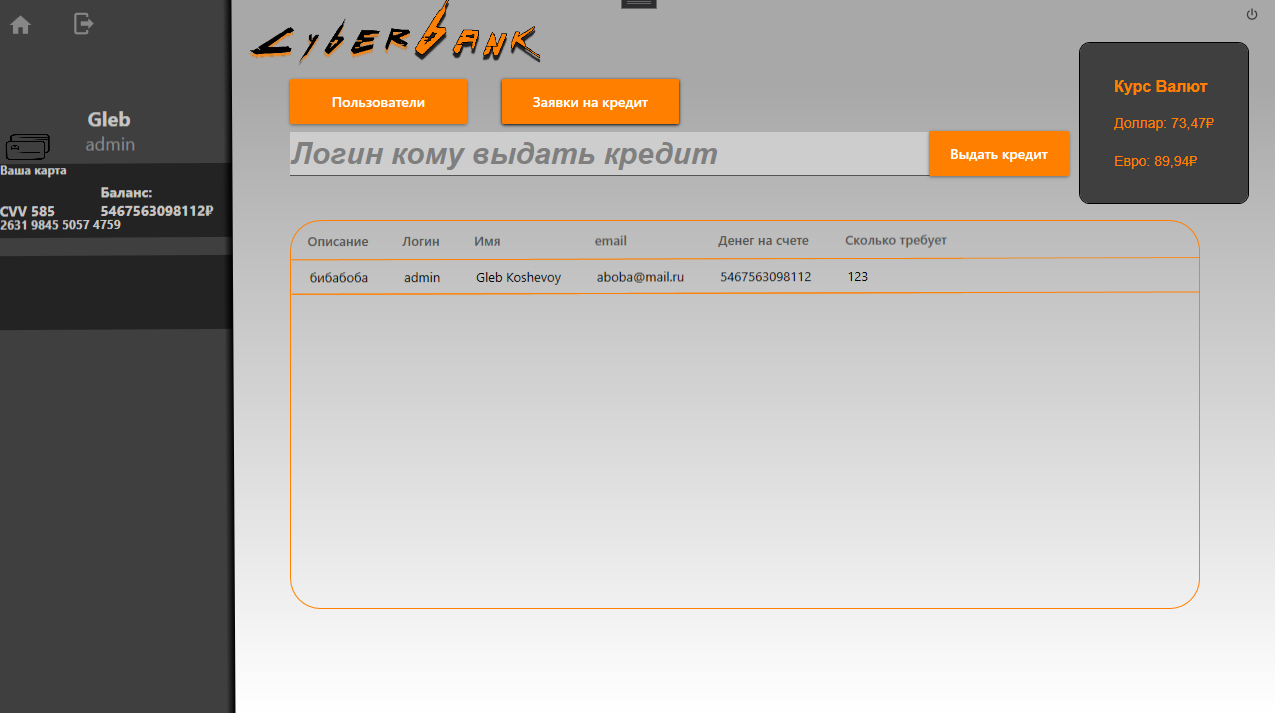
Кроме того, администратор может просматривать список запросов (см рисунок 14) на получение кредитной карты. Для этого необходимо нажать на кнопку “Заявки на кредит” в верхней части экрана. Администратор может одобрять заявки в случае, если запрос устраивает банк. Администратор видит описание запроса, логин пользователя, который подал заявку, его имя, адрес электронной почты, указанный при подаче заявки, количество денег на дебетовом счете, для оценки благополучности финансового положения пользователя, а также количество денег, которое пользователь хочет иметь на своей кредитной карте. Для одобрения запроса администратору необходимо ввести в специальное поле логин пользователя, который просит кредит, а затем нажать на кнопку “Одобрить”, которая находится справа от поля ввода. Это сделано для того, чтобы избежать случайного нажатия кнопки. Когда администратор одобрит кредит на почту пользователя придет письмо, ****информирующее его о том, что кредитная карта одобрена.

Рисунок 14. Заявки на получение кредитной карты CyberBank

## **2.3 Руководство по установке и работе с приложением и БД.**

Для того чтобы загрузить базу данных на локальный сервер необходимо установить СУБД MySQL на свой компьютер. Затем необходимо импортировать базу данных любым удобным способом, либо через консоль, либо через какое-либо приложение с графическим интерфейсом для управления базой данных. В случае разработки этого проекта это было приложение MAMP, которое позволяет управлять базой данных через веб-приложение phpMyAdmin.

Для работы с приложением можно либо запустить его через exe файл, либо открыть код проекта через Microsoft Visual Studio. В зависимости от того какой локальный сервер используется, может быть, необходимость изменить данные сервера в коде приложения для того, чтобы подключиться к базе данных. Данные о подключении к базе данных можно посмотреть в классе DataBase.cs. Необходимо будет указать название сервера, порт, логин и пароль для доступа, а также название базы данных (по умолчанию cyberbank).

Если подключение установлено успешно, то приложение заработает и позволит зарегистрироваться или авторизоваться. Если же произошла ошибка соединения, то приложение уведомит об этом.

## **2.4 Обеспечение информационной безопасности создаваемой базы данных**

Для обеспечения информационной безопасности базы данных все конфиденциальные данные пользователей шифруются с помощью алгоритма шифрования Рэйндаля (Rijndael) со статическим ключом. Шифрование данных происходит при помощи метода хэширования хэшем MD5.

# **Заключение**

Основной задачей этой курсовой работы была разработка базы данные для системы расчета с клиентами в банке и создание интерфейса для взаимодействия с ней. После анализа предметной области были спроектированы инфологическая и даталогическая модели базы данных. Также было разработано приложение, которое было задумано для того, чтобы обычным пользователям и администраторам банка было удобнее выполнять необходимые им запросы. В процессе выполнения курсовой были изучены множество форумов, сайтов, видео, а также книг и технических документаций, которые были необходимы для продвижения в разработке. Была изучена программа phpMyAdmin, которая использовалась для наглядного изучения процессов происходящих в базе данных, а также предоставила вид на датологическую модель базы данных. Для реализации er-модели была использован сайт Draw.io. Для разработки приложения был использован язык C# и фреймворк .NET Framework, с использованием платформы WPF, языка программирования python, языка Visual Basic Script, а также в виде редактора кода были использованы интегрированная среда разработки Microsoft Visual Studio и Sublime Text 3. Для создание локального сервера phpMyAdmin была использована программa MAMP.

После реализации курсовой работы были выполнены следующие задачи:

1) Проанализирована предметная область, спроектированы инфологическая и даталогическая модели данных.

2) Разработана база данных, содержащая 4 сущности. Созданы атрибуты сущностей и связи между ними.

3) Был спроектирован и разработан графический интерфейс приложения, которое работает и для простых пользователей, и для администраторов.

4) Реализовано шифрование данных для их защиты.

После выполнения курсовой работы были сделаны выводы:

1) Базы данных необходимы при работе с большим количеством информации. Это ускоряет процесс и делает его более удобным для сотрудников и пользователей.

2) Данные, которые содержатся в базе данных нужно хорошо шифровать для обеспечения конфиденциальности данных пользователей.

3) Пользовательский интерфейс важная часть приложения, благодаря ему у приложения есть шанс стать более популярным среди людей, но не стоит забывать о функционале, который тоже важен. Необходимо находить баланс между красотой и практичностью.

Результатом данной курсовой стала база данных системы расчета с клиентами в банке, а также приложение с приятным глазу и удобном в использовании пользовательским интерфейсом, которое позволяет пользователем создавать дебетовые и кредитные карты и переводить свои деньги с них на счета других пользователей, а администраторам просматривать список пользователей и выдавать кредитные карты.

# **Список используемой литературы**

1. Чернышев, С. А.  Основы программирования на python: учебное пособие для вузов / С. А. Чернышев. — Москва: Издательство Юрайт, 2021.
2. Маркин, А. В.  Программирование на SQL: учебное пособие для среднего профессионального образования / А. В. Маркин. — Москва: Издательство Юрайт, 2021.
3. Федоров, Д. Ю.  Программирование на языке высокого уровня Python: учебное пособие для среднего профессионального образования / Д. Ю. Федоров. — 2-е изд. — Москва: Издательство Юрайт, 2021.
4. Маркин, А. В.  Программирование на SQL в 2 ч. Часть 1: учебник и практикум для вузов / А. В. Маркин. — 2-е изд., перераб. и доп. — Москва: Издательство Юрайт, 2021.
5. Маркин, А. В.  Программирование на SQL в 2 ч. Часть 1: учебник и практикум для вузов / А. В. Маркин. — 2-е изд., перераб. и доп. — Москва: Издательство Юрайт, 2021.
6. Коннолли Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика / Т. Коннолли, К. Бегг - М.: Вильямс, 2017.
7. Гордеев, С. И. Организация баз данных в 2 ч. Часть 1: учебник для вузов / С. И. Гордеев В. Н. Волошина. — 2-е изд., испр. и доп. — Москва: Издательство Юрайт, 2019.
8. Базы данных: учебно-методический комплекс /сост.: М.В. Копейкин В.В., Спиридонов Е.О. Шумова. - СПб.: Изд-во СЗТУ, 2016
9. Стружкин Н. П. Базы данных: проектирование. Практикум: учеб. пособие для СПО / Н. П. Стружкин, В. В. Годин. — Москва: Издательство Юрайт, 2019
10. Илюшечкин, В. М.  Основы использования и проектирования баз данных: учебник для вузов / В. М. Илюшечкин. — Москва: Издательство Юрайт, 2021.
11. Нестеров, С. А.  Базы данных: учебник и практикум для вузов / С. А. Нестеров. — Москва: Издательство Юрайт, 2021.
12. Тузовский, А. Ф.  Объектно-ориентированное программирование: учебное пособие для вузов / А. Ф. Тузовский. — Москва: Издательство Юрайт, 2020.
13. Фомичёв, В. М.  Криптографические методы защиты информации в 2 ч. Часть 1. Математические аспекты: учебник для вузов / В. М. Фомичёв, Д. А. Мельников; под редакцией В. М. Фомичёва. — Москва: Издательство Юрайт, 2021.
14. Фомичёв, В. М.  Криптографические методы защиты информации в 2 ч. Часть 2. Системные и прикладные аспекты: учебник для вузов / В. М. Фомичёв, Д. А. Мельников; под редакцией В. М. Фомичёва. — Москва: Издательство Юрайт, 2021.
15. Гниденко, И. Г.  Технология разработки программного обеспечения: учебное пособие для среднего профессионального образования / И. Г. Гниденко, Ф. Ф. Павлов, Д. Ю. Федоров. — Москва: Издательство Юрайт, 2021.
16. Гниденко, И. Г.  Технологии и методы программирования: учебное пособие для вузов / И. Г. Гниденко, Ф. Ф. Павлов, Д. Ю. Федоров. — Москва: Издательство Юрайт, 2021.
17. Токарев, В. В.  Методы оптимизации: учебное пособие для вузов / В. В. Токарев. — Москва: Издательство Юрайт, 2021.
18. Cтасышин, В. М.  Базы данных: технологии доступа: учебное пособие для среднего профессионального образования / В. М. Стасышин, Т. Л. Стасышина. — 2-е изд., испр. и доп. — Москва: Издательство Юрайт, 2021.
19. Кудрина, Е. В.  Основы алгоритмизации и программирования на языке C#: учебное пособие для вузов / Е. В. Кудрина, М. В. Огнева. — Москва: Издательство Юрайт, 2021.
20. Внуков, А. А.  Защита информации в банковских системах: учебное пособие для вузов / А. А. Внуков. — 2-е изд., испр. и доп. — Москва: Издательство Юрайт, 2021.