Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра информатики

**ОТЧЕТ**

к лабораторной работе

на тему

**РАЗРАБОТКА ПРИЛОЖЕНИЯ**

|  |  |
| --- | --- |
| Студент | Н. С. Слуцкий |
| Руководитель | Е. В. Тушинская |
| Нормоконтролер | Е. В. Тушинская |
| Рецензент | Е. В. Тушинская |

Минск 2023

СОДЕРЖАНИЕ

[Введение 3](#_Toc151730199)

[1 Обоснование инструментов разработки 4](#_Toc151730200)

[2 Описание частей приложения 5](#_Toc151730201)

[2.1 Поддержание сессии 5](#_Toc151730202)

[2.2 Проведение платежей 5](#_Toc151730203)

[2.3 Работа с базой данных и разграничение прав 6](#_Toc151730204)

[2.4 Другие особенности 6](#_Toc151730205)

[Заключение 8](#_Toc151730206)

[Приложение А 9](#_Toc151730207)

# Введение

Целью данной лабораторной работы ставится:

– разработать полноценное приложение в соответствии с требованиями, изложенными в отчётах лабораторным работам №1 и №2;

– спроектировать приложение таким образом, чтобы пользователю была понятна логика взаимодействия с интерфейсом и в целом задумкой автора;

– развернуть приложение в открытый доступ;

– оформить отчёт о проделанной работе в соответствии со стандартом предприятия БГУИР 2017 [1];

– протестировать работу приложения под разными пользователями и с разных устройств;

– во время разработки ознакомиться с новыми для автора технологиями: NextJS, Supabase.

Для достижения поставленных целей необходимо изучить подходы серверного рендеринга, клиентских компонентов, использования инструментария в подходе «Software as a service», а также вспомнить принципы работы с системой управления базой данных «PostgreSQL», написания запросов, хранимых процедур.

# 1 Обоснование инструментов разработки

В основе разработки программы используется библиотека для отрисовки интерфейсов ReactJS

В большей части начинаемых проектов используется утилита «create-react-app» для разворачивания рабочей конфигурации для разработки и последующей сборки.

Сейчас если посетить страницу документации React в разделе установки, можно увидеть различные предложенные варианты установки фреймворков React: Next.js, Remix. Но при этом нет варианта установки самого React в чистом виде.

Create React App решал множество проблем, но со временем происходила стагнация, приложения, созданные с использованием Create React App становятся медленнее альтернативных. При всем при этом Create React App сильно отстает в технологиях, например, не поддерживает подходы SSR/SSG. Поэтому к React надо относиться именно как к библиотеке и по возможности использовать фреймворки, которых много и это хорошо.

На данный момент Create React App не соответствует цели быть лучшей утилитой для создания приложения.

Исходя из описанных доводов и преимуществ, которые предоставляет NextJS, выбор и был сделан в его пользу. Также интересом являлось изучить подробнее этот фреймворк и создать с его помощью первое приложение.

Удобным инструментом для работы с готовой исполняющейся СУБД является Supabase. Этот сервис предоставляет доступ ко всем возможностям PostgreSQL, а также некоторые другие удобные возможности в виде аутентификации и так далее.

# 2 Описание частей приложения

Как и любое другое современное NextJS-приложение, разрабатываемый продукт включает в себя серверные и клиентские компоненты. Серверные компоненты рендерятся на сервере Next и возвращаются в виде разметки. В них можно безопасно делать запрос в базу данных и возвращать в виде разметки, минуя возможность кражи данных для доступа к базе.

По условию лабораторной работы необходимо описать три главных решения из проекта. Ниже приведены эти решения.

## 2.1 Поддержание сессии

Для поддержания сессии создан файл middleware.ts, который при каждом новом запросе на страницу проверяет средствами Supabase факт того, залогинен пользователь или нет.

Из минусов такого подхода можно выделить частоту запросов, но, вероятно, внутри реализации Supabase API эти запросы кешируются и их количество оптимизируется. Из плюсов можно выделить поддержание актуального состояния сессии. На рисунке 2.1 приведена блок-схема алгоритма работы middleware.

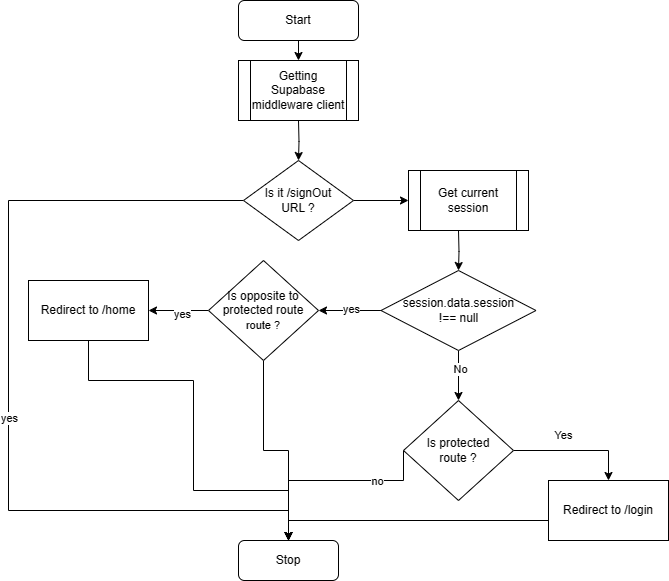


Рисунок 2.1 – Блок-схема работы функции в middleware

В случае отсутствия актуальной сессии и попытке перехода на маршрут, требующий наличие этой сессии, производится автоматическое перенаправление на страницу входа.

## 2.2 Проведение переводов

Для проведения платежей создана серверная страница и клиентский компонент. Страница при загрузке запрашивает список счетов текущего пользователя. Клиентский компонент позволяет выбрать счёт для оплаты, ввести номер счёта получателя и ввести переводимую сумму.

Здесь же используется библиотека для сканирования QR-кодов. Для каждого своего виртуального счёта клиент может сгенерировать постоянный QR-код, который может быть использован при переводах. Таким образом при желании перевести средства можно не вводить длинный номер счёта, а отсканировать QR-код получателя – и поле заполнится автоматически. Это очень удобная возможность, значительно упрощающая опыт использования разрабатываемой системы. На стороне сервера Supabase в созданной базе данных создана хранимая процедура, которая позволяет выполнить транзакцию.

На рисунке 2.1 приведена блок-схема алгоритма хранимой функции для проведения перевода.

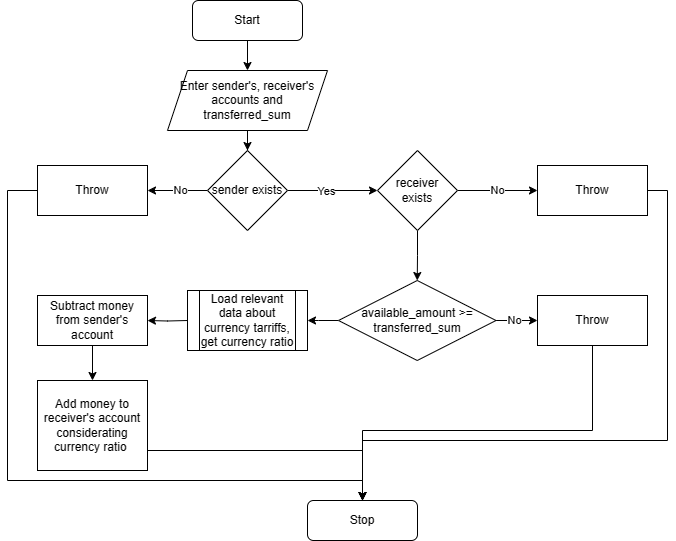


Рисунок 2.2 – Блок схема алгоритма проведения перевода

Она проверяет существование счетов получателя и отправителя, наличие необходимых средств на счету отправителя, производит конвертацию средств из одной валюты в другую в соответствии с актуальными курсами банка, если валюты счёта-получателя и счёта-отправителя отличаются.

Написание процедуры позволяет не уходить от настроек RLS внутри Supabase, то есть ограничивать право доступа к таблицам на уровне базы данных.

## 2.3 Работа с базой данных и разграничение прав

В проекте используется механизм RLS, предоставляемый Supabase. Таким образом появляется возможность разграничить доступы на уровне базы данных путём создания политик безопасности внутри PostgreSQL. Примеры создания политик, а также функция, которая может вызываться с правами доступа ко всему, приведены в приложении к настоящему отчёту. Это значит, что, например, запрос select вернёт только те строки, которые ассоциированы с текущим пользователем. Можно прописывать разные ограничения для разных запросов. Хорошая практика заключается в том, что нужно использовать по возможности хранимые процедуры и функции внутри базы данных, а не писать весь функционал на промежуточной стороне в виде сервиса. В рамках приложения созданы хранимые процедуры и триггеры для совершения оплаты, заполнения таблицы с кастомными данными о пользователе при регистрации, проверке соответствия пароля и логина при входе в приложение.

## 2.4 Система валидации

Всё строго затипизировано с использованием лучших практик TypeScript. Отдельного внимания заслуживает имплементация наследования компонентов LabeledInput и Input, где построена система наследующихся generic-типов.

Также можно обратить внимание на скоро-написанную систему валидации для полей ввода. Система типов простая, но для упрощённого аналога настоящего банковского приложения она очень хорошо интегрируется. На рисунке 2.1 представлена такая схема.

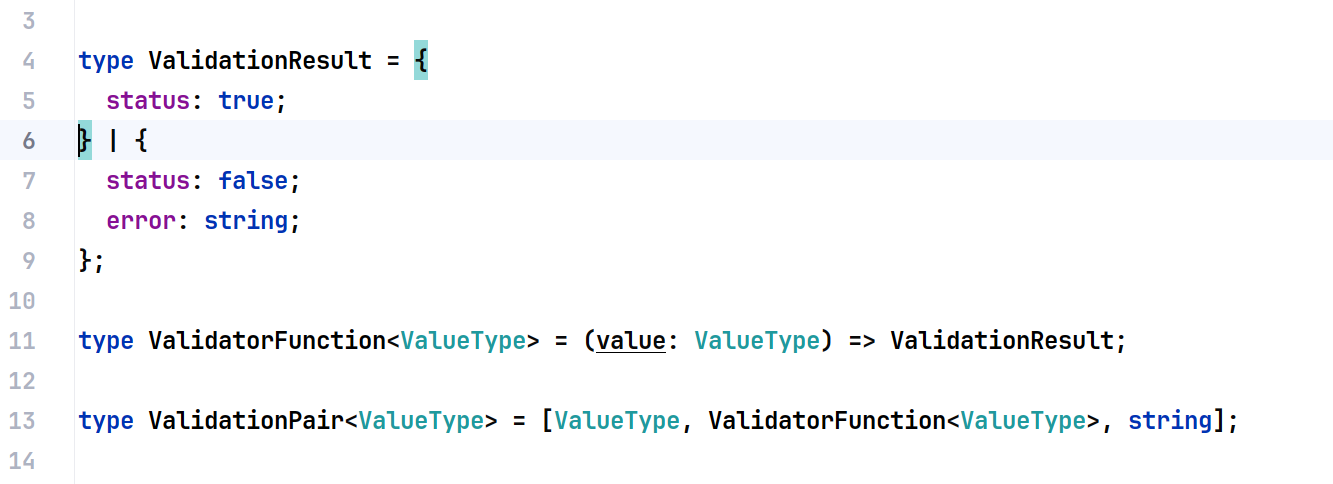


Рисунок 2.3 – Система типов валидации

Функция валидации принимает в себя массив пар с объектом данных и указателем на валидатор для него. Валидатор может вернуть успех, либо ошибку с некоторым сообщением. Таким образом, если в форме много полей, их можно быстро провалидировать и получить сообщение о первой ошибке путём вызова данной функции. На рисунке 2.4 представлена блок-схема, соответствующая описанному процессу.

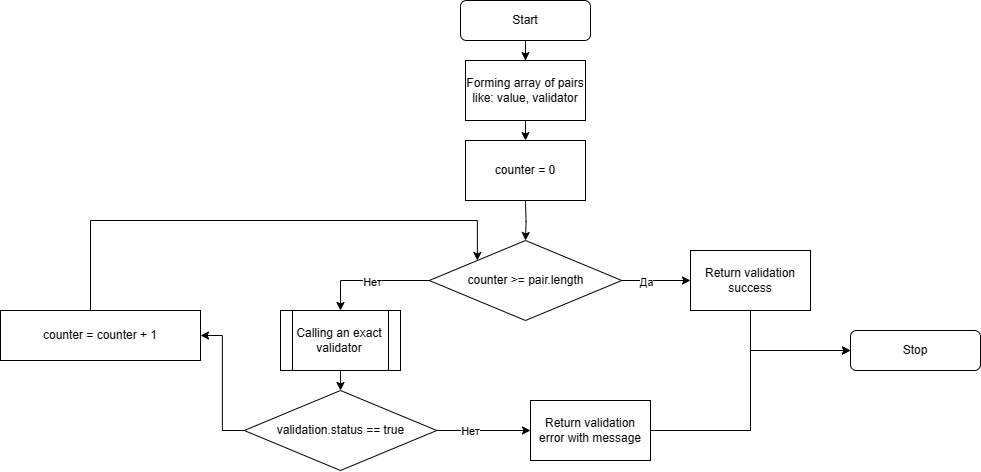


Рисунок 2.4 – Блок-схема с работой функции валидации

После завершения разработки первой версии приложения необходимо его развернуть и сделать общедоступным. Приложение удобно деплоится на сервис Vercel – именно эта компания и разрабатывает используемый фреймворк NextJS.

# Заключение

В результате выполнения лабораторной работы была детально продумана функциональность разрабатываемого веб-приложения виртуального банка.

Приложение было разработано и задеплоеено. Произведено ознакомление с современным стеком разработки веб-приложений.

В ходе выполнения был составлен и оформлен отчёт по большей части в соответствии с общими требованиями стандарта предприятия БГУИР.

Цели лабораторной работы можно считать достигнутыми.

# Приложение А

(Листинг кода)

drop function if exists transfer\_money;

create or replace function transfer\_money (sender\_id uuid , receiver\_id uuid , transfer\_amount float)

returns void

security definer

language plpgsql

as

$$

declare

current\_amount numeric;

currency\_id\_from NUMERIC;

currency\_id\_to NUMERIC;

ratio NUMERIC;

begin

if not exists (Select 1 from "public"."BankAccounts" where number = receiver\_id) then

RAISE 'Invalid receivers account (card) number. Not found';

end if;

if not exists (Select 1 from "public"."BankAccounts" where number = sender\_id) then

RAISE 'Invalid senders account (card) number. Not found';

end if;

select "public"."BankAccounts"."amount" into current\_amount from "public"."BankAccounts" where number = sender\_id;

if current\_amount < transfer\_amount then

RAISE 'Not enough money on this card (money account)';

end if;

select currency into currency\_id\_from from "public"."BankAccounts" where number = sender\_id;

select currency into currency\_id\_to from "public"."BankAccounts" where number = receiver\_id;

select coefficient into ratio from "public"."CurrencyTariffs" where "public"."CurrencyTariffs"."from" = currency\_id\_from and "public"."CurrencyTariffs"."to" = currency\_id\_to;

update "public"."BankAccounts"

set amount = amount + transfer\_amount \* ratio

where number = receiver\_id;

update "public"."BankAccounts"

set amount = amount - transfer\_amount \* ratio

where number = sender\_id;

end;

$$;

create policy "Public profiles are viewable by everyone."

on Profiles for select

using ( true );

create policy "Users can insert their own profile."

on Profiles for insert

with check ( auth.uid() = id );

create policy "Users can update own profile."

on Profiles for update

using ( auth.uid() = id );