# **ВВЕДЕНИЕ**

Автоматизированная банковская система (сокр. АБС) – комплекс программного и технического обеспечения, направленный на автоматизацию банковской деятельности.

Американская исследовательская и консалтинговая компания Gartner определяет основную банковскую систему как внутреннюю автоматизированную систему, которая обрабатывает ежедневные банковские операции и производит обновление счетов и других финансовых записей. Основные банковские системы обычно включают в себя возможности:

* открытия и ведения банковских счетов физических и (или) юридических лиц;
* обработки депозитов, займов и кредитов;
* подключения к стандартным системам бухгалтерского учета и инструментам отчетности.

Целью данной практической работы является реализация программного средства для автоматизации деятельности условного коммерческого учреждения (банка).

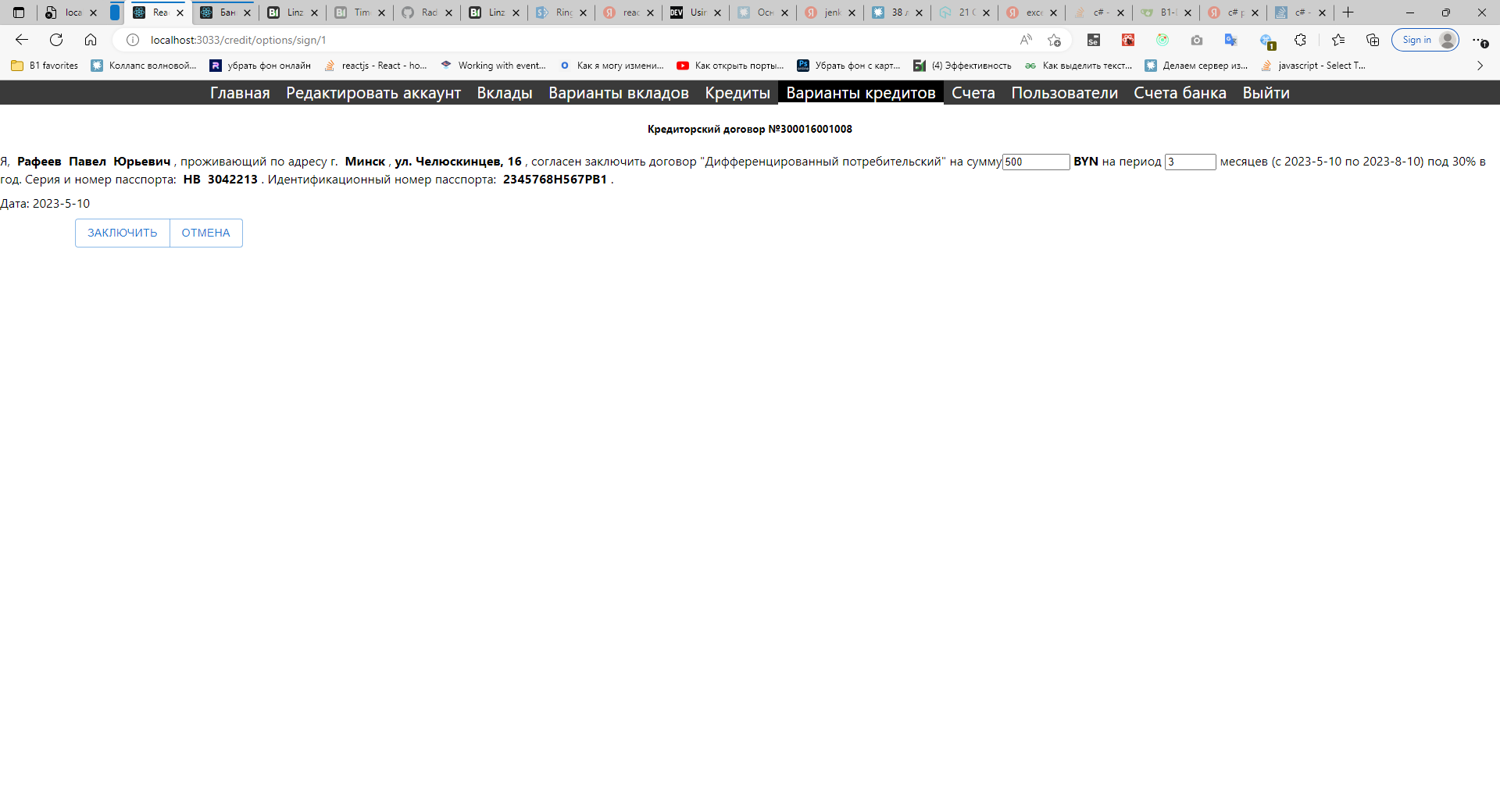


Рисунок 3.6 – Страница заключения договора

После заключения договора создаются 2 счета: основной и процентный. Информация по ним доступна в разделе «Счета» и включает в себя таблицы транзакций по каждому счету и график зависимости суммы на счете от веремени.

**ПРИЛОЖЕНИЕ А**

**Код программы**

файл **program.cs**

using Microsoft.AspNetCore.Identity;

using Microsoft.Extensions.Configuration;

using PiRiS\_back.Models;

using PiRiS\_back;

using PiRiS\_back.Middleware;

using Microsoft.EntityFrameworkCore;

using PiRiS\_back.Services;

using PiRiS\_back.Enums;

using Microsoft.AspNetCore.Authentication.Negotiate;

using Microsoft.AspNetCore.Server.IISIntegration;

var builder = WebApplication.CreateBuilder(args);

var services = builder.Services;

var configuration = builder.Configuration;

services.Configure<BankAccConfig>(configuration.GetSection("Accounts"));

services.AddTransient<UserInfoFillerService>();

services.AddTransient<AccountsService>();

services.AddSingleton<ContractsServiceSingletone>();

services.AddScoped<IdentityNameFilter>();

services.AddScoped<AuthFilter>();

services.AddControllers();

services.AddSession();

services.AddEndpointsApiExplorer();

services.AddSwaggerGen();

services.AddCors(options =>

{

options.AddPolicy("AllowAllHeaders",

builder =>

{

builder.WithOrigins("http://localhost:3033")

.AllowCredentials()

.AllowAnyHeader()

.AllowAnyMethod();

});

});

services.AddAuthentication(NegotiateDefaults.AuthenticationScheme)

.AddNegotiate();

services.AddAuthorization(options =>

{

// By default, all incoming requests will be authorized according to the default policy.

options.FallbackPolicy = options.DefaultPolicy;