Данный документ представляет собой описание приложения для расчёта бюджета семьи «**Семейный калькулятор**» (BudgetCalculatorApp).

Приложение предназначено для ведения семейного бюджета. Решение представляет из себя четыре проекта:

* Бизнес-логика (BudgetCalculatorApp)
* Пользовательский интерфейс (BudgetCalculatorAppUI)
* Проект с юнит-тестами (BudgetCalculatorAppUnitTests)
* Класс для связи с БД (DBWrapper)

Обозреватель решения показан на рисунке 1.1:

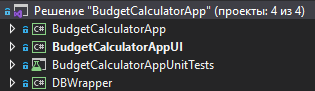


Рисунок 1.1 – Обозреватель решения с проектами

В проекте BudgerCalculatorApp реализованы классы пользователя(**User**), транзакции(**Transaction**), категории транзакции(**TransactiomCategory**) и перечисление обозначающее тип транзакции (**TransactionType**). У классов созданы поля хранящие необходимые данные, свойства для работы с ними без нарушения инкапсуляции и необходимые методы. Так же для связи в БД добавлены навигационные свойства и свойства ответственные за первичные и внешние ключи.

Добавление/редактирование/удаление реализуется с помощью определенных форм создающихся для пользователя или транзакции. Создании категории осуществляется в форме для создания транзакции, хоть в логике программы они отделены. С помощью класса AppContextExtensions при добавлении категории проверяется, существует ли уже такая категория или нет, если категория есть, то в транзакцию записывается Id существующей категории, иначе создаётся новая.

На изображении 1.2 показано главное окно приложения на котором располагаются все пользователи (члены семьи). При нажатии на любую ячейку строки, пользователь выделяется и считается текущим пользователем, информация о нём отображается в левом верхнем углу:

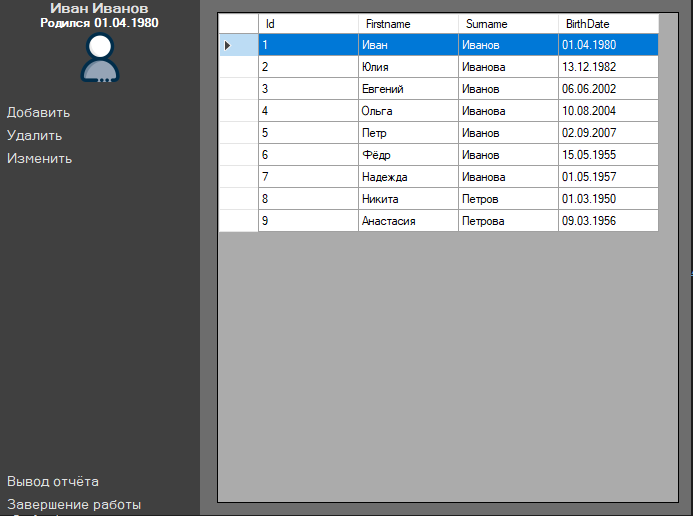


Рисунок 1.2 – Главное окно приложения

Так как данное приложение должно работать с БД в проекте DBWrapper был написан класс AppContext, являющийся классом контекста базы данных для Entity Framework с помощью PostgreSQL. Данный класс выполняет подключение к локальной БД находящуюся в pgAdmin 4, определения набора данных, создание связей между сущностями и позволяет выполнять операции с БД. Скриншот созданных таблиц в pgAdmin 4 после миграции и все добавленные пользователи на рисунках 1.3 и 1.4:

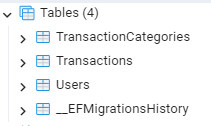


Рисунок 1.3 – Таблицы созданные в pgAdmin 4

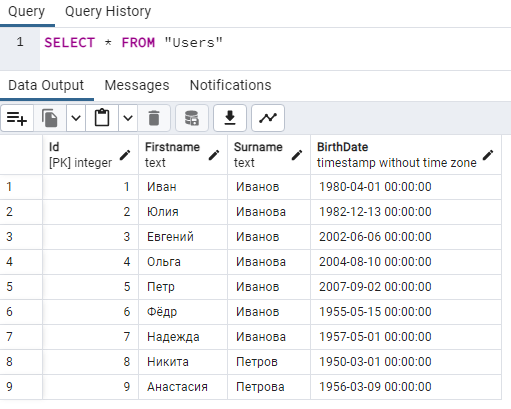


Рисунок 1.4 – Выходные данные по запросу вывода всех добавленных в БД пользователей

Навигация вглубь базы данных в приложении осуществляется с помощью двойного клика, а обратно с помощью кнопки “Назад”, на рисунке 1.5 показаны все транзакции определенного пользователя, а на рисунке 1.6 категория необходимой транзакции:

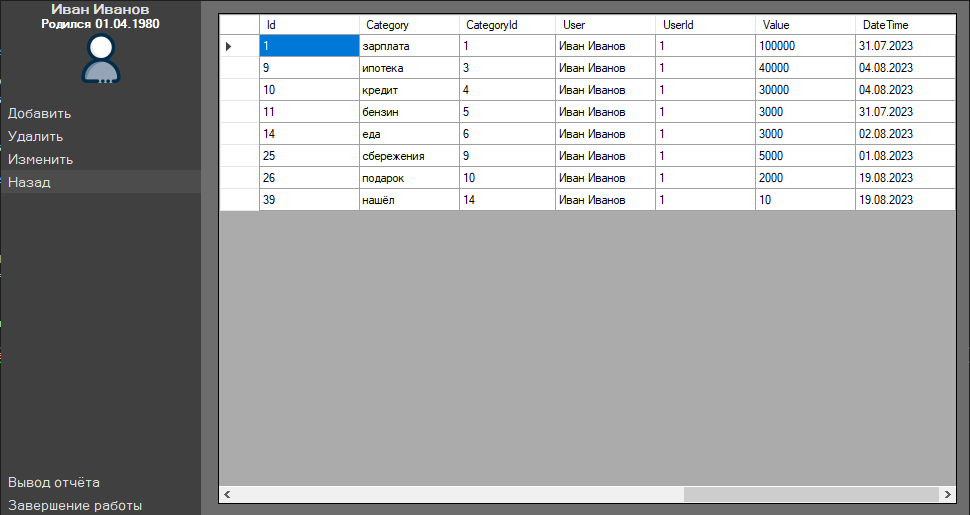


Рисунок 1.5 – Транзакции пользователя

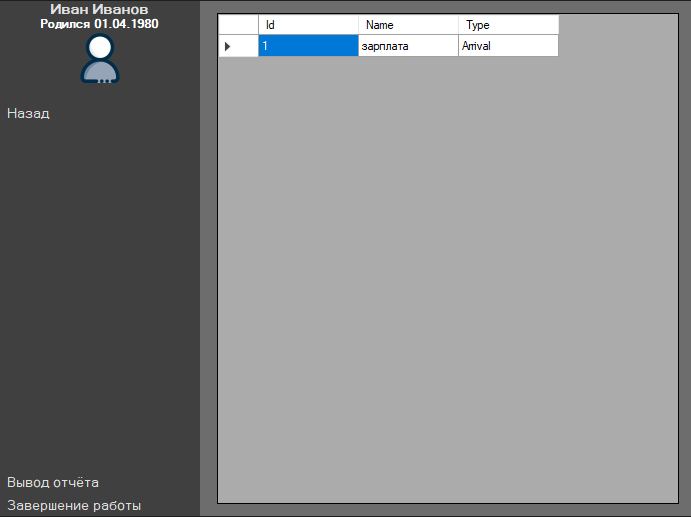


Рисунок 1.6 – Категория транзакции

На изображении 1.7 показан пример SQL запроса выдающий все транзакции пользователя в локальной БД:

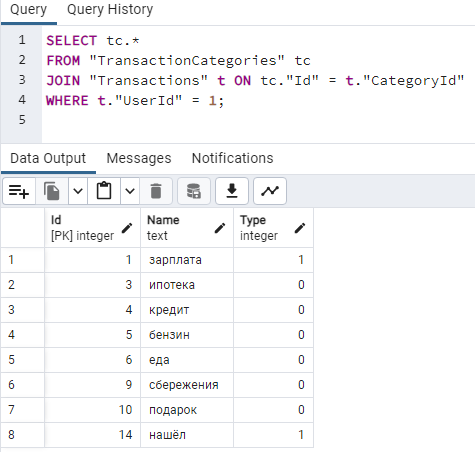


Рисунок 1.7 – Результат SQL запроса

Так же реализована функциональность отчёта расходов/приходов для отдельного пользователя (рисунок 1.8 - выводится при нажатии на кнопку “Вывод отчёта” в левом нижнем углу при одном выделенном пользователе) и общего отчёта всех пользователей с графиком для расходных категорий и соотношения расхода к приходу всей семьи (рисунок 1.9 выводится, если при нажатии на кнопку “Вывод отчёта” не выбран пользователь):

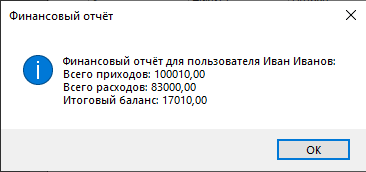


Рисунок 1.8 – Индивидуальный отчёт пользователя

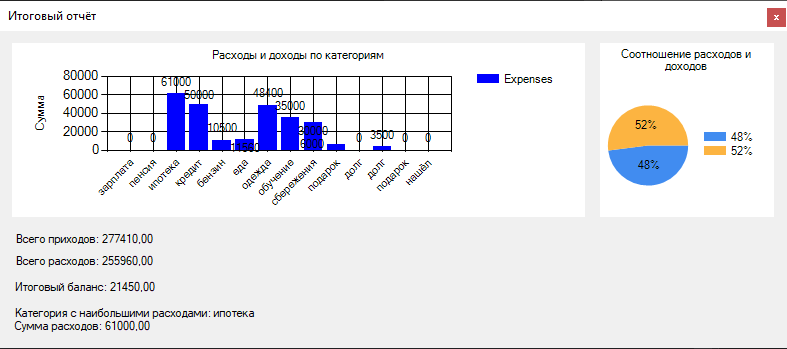


Рисунок 1.9 – Общий финансовый отчёт всех пользователей

Логика программы покрыта юнит-тестами на 100%, что продемонстрировано на изображении 1.10:

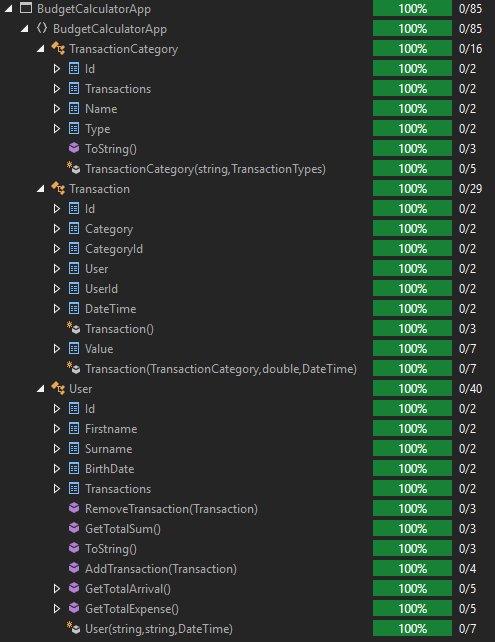


Рисунок 1.10 – Покрытие логики юнит-тестами