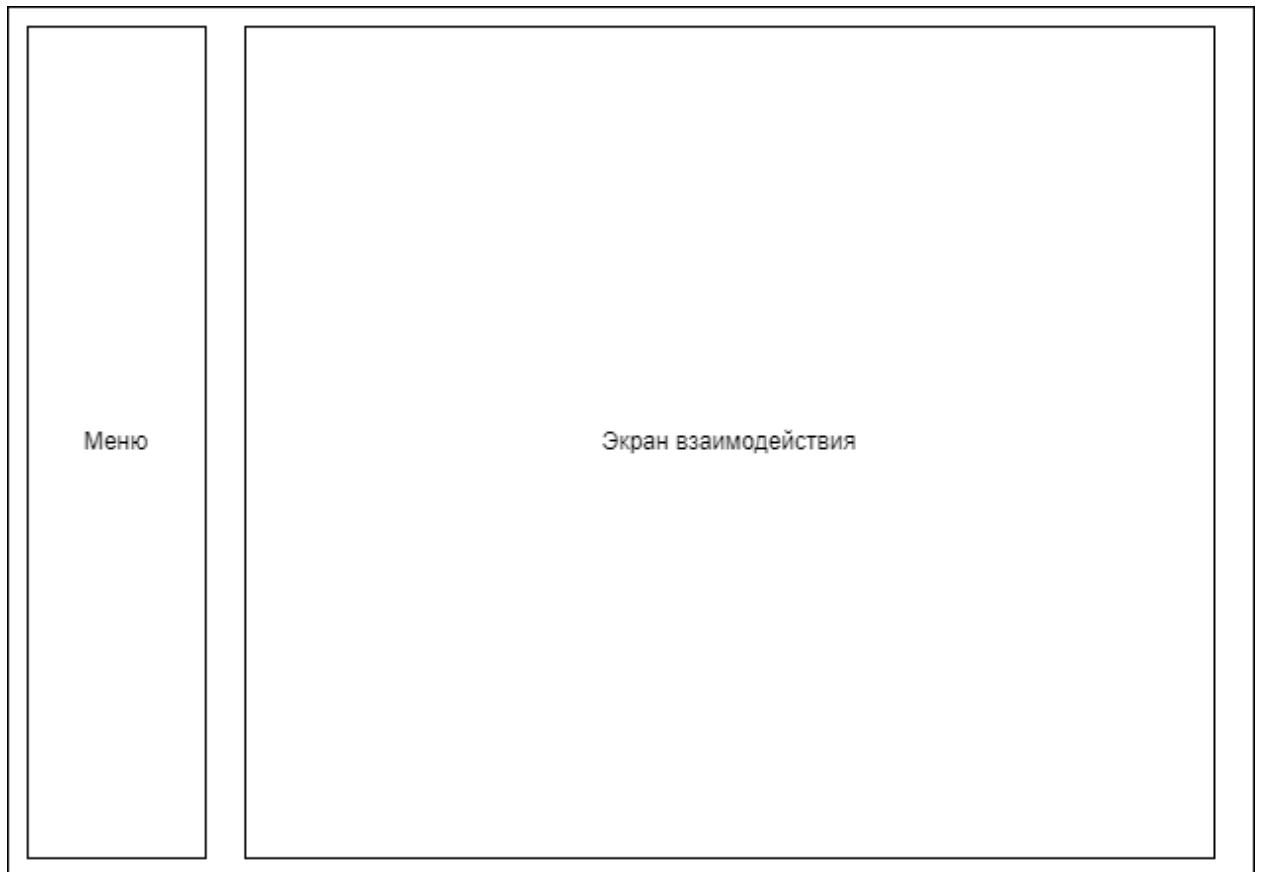


## Учёт личных финансов

Прототипы экранных форм:

### 1. Прототип главного экрана:

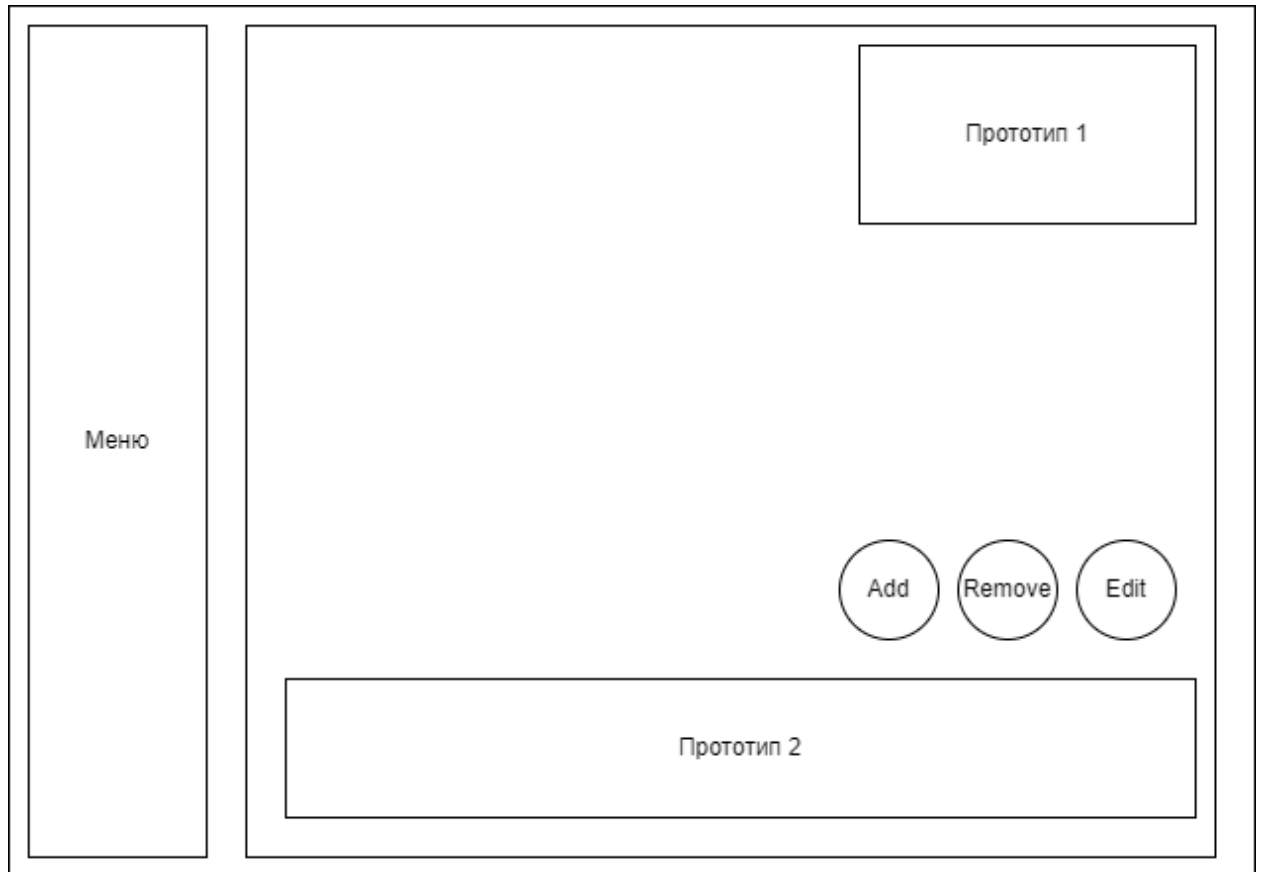
Главный экран состоит из меню и из экрана взаимодействия с пользователем. Меню – это набор кнопок для переключения между вкладками приложения (главная, информация о транзакциях, категории, аккаунты). Экран взаимодействия с пользователем – это то место, куда будет выводиться основная информация в процессе работы приложения.



## 2. Прототип экрана о транзакциях:

Экран о транзакциях состоит из группы кнопок (добавить, удалить, изменить), календаря (прототип 1) и карточки категории (прототип 2).

Также при нажатии на кнопку “Add” открывается карточка создания транзакции (прототип 3).



### 3. Календарь

Прототип календаря состоит из трёх групп выбора (день, месяц, год).

Группа дней состоит из 5 элементов – это соответственно дни месяца, который можно прокручивать и выбирать.

The prototype shows a date selection interface. On the left, there is a vertical column of five boxes, each labeled 'Day'. To the right of this column are two boxes labeled 'Month' and 'Year'. At the bottom of the interface are two buttons labeled 'OK' and 'Cancel'.

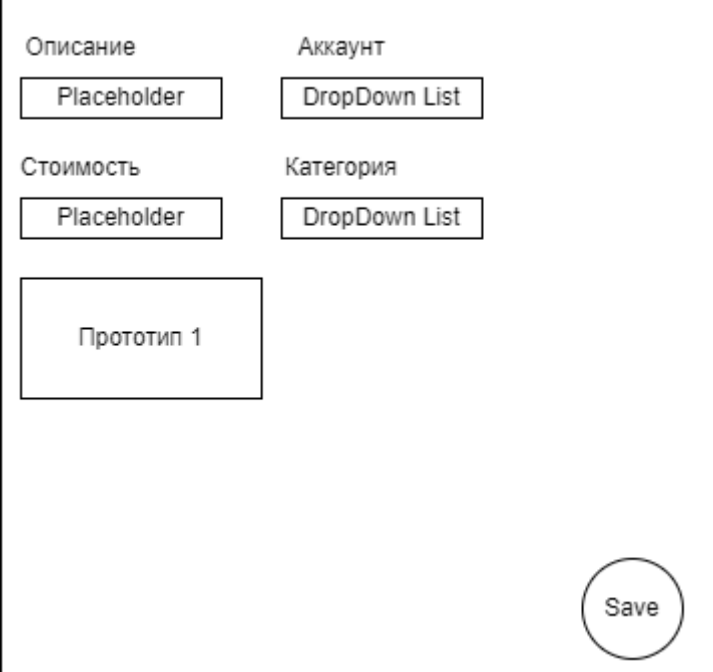
### 4. Карточка категории

Прототип состоит из следующих элементов: кружок, который разделяет категории по цвету, название категории, стоимость категории и также есть кнопка в виде стрелочки, которая открывает список транзакций по категории (прототип 4).

The prototype is a horizontal card. On the left is a circle. To its right is the text 'Название категории'. On the far right is the text 'Стоимость'. Below the circle is a small downward-pointing arrow 'v'.

## 5. Карточка создания транзакции

Данный прототип состоит из двух полей для текста (placeholder) для описания и стоимости транзакции и из двух элементов выбора из списка (dropDown List). Также присутствует календарь (прототип 1) и кнопка сохранения (Save).



Прототип карточки создания транзакции. Визуально это форма с четырьмя полями ввода, расположенными в два столбца. Верхний ряд содержит поле "Описание" с placeholder "Placeholder" и поле "Аккаунт" с "DropDown List". Нижний ряд содержит поле "Стоимость" с placeholder "Placeholder" и поле "Категория" с "DropDown List". В левом нижнем углу находится календарь, обозначенный как "Прототип 1". В правом нижнем углу расположена круглая кнопка "Save".

## 6. Карточка транзакции

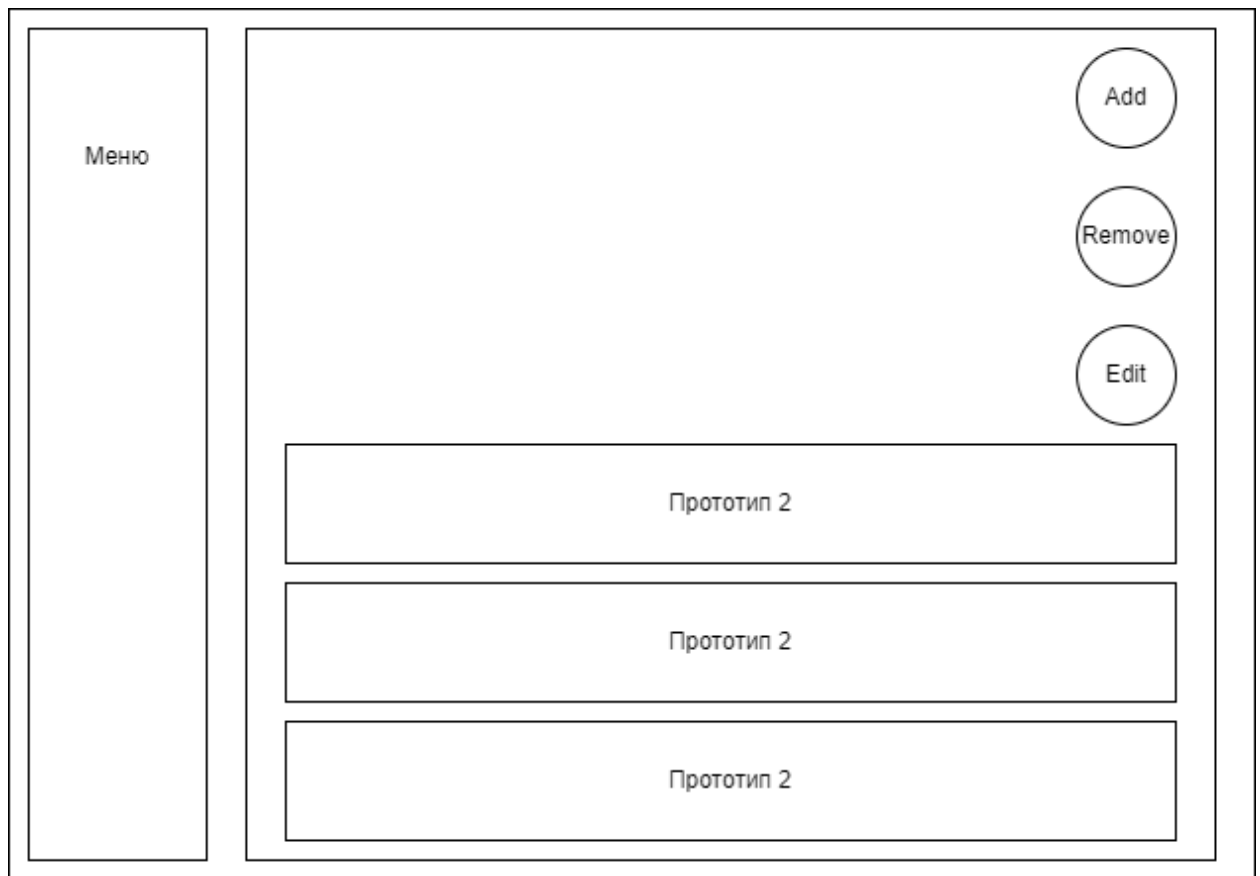
Карточка транзакции состоит из 4 элементов. Описание транзакции, дата и время транзакции, стоимость транзакции, аккаунт. Все 4 элемента являются текстом.



Прототип карточки транзакции. Визуально это прямоугольная форма, разделенная на четыре текстовых поля. Верхний левый угол содержит "Описание", верхний правый — "Стоимость". Нижний левый угол содержит "Дата и время", нижний правый — "Аккаунт".

## 7. Экран со списком категорий

Экран с категориями состоит из группы кнопок (Add, Remove, edit). Также в нижней части экрана находится список категорий транзакций, состоящий из карточек транзакций (прототип 2).



## 8. Экран со списком аккаунтов.

Экран с аккаунтами состоит из итоговой стоимости по всем аккаунтам, список аккаунтов и группа из кнопок (Add, Remove, Edit).

Меню

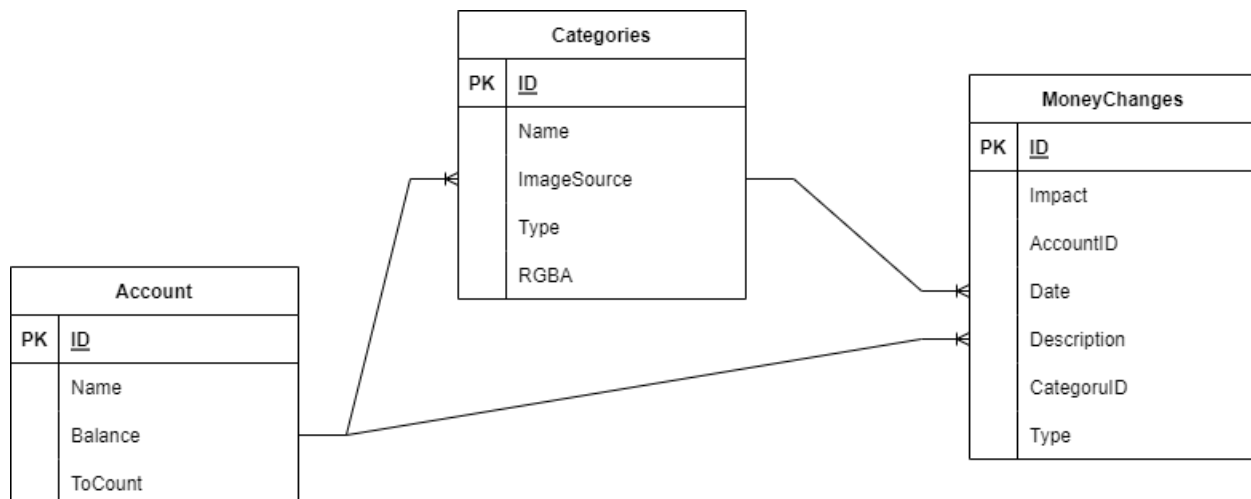
Total

Аккаунт

Аккаунт

Add Remove Edit

Диаграмма сущностей:



Разработка API системы:

1. AddAccount

Функция предназначена для добавления нового аккаунта в базу данных.

Функция использует Entity Framework.

Входная информация: переменная типа Account.

Выходная информация: Таблица БД.

2. RemoveAccount

Функция предназначена для удаления аккаунта из базы данных.

Функция использует Entity Framework.

Входная информация: переменная типа Account.

Выходная информация: Таблица БД.

3. AddCategory

Функция предназначена для добавления новой категории в базу данных.

Функция использует Entity Framework.

Входная информация: переменная типа Category.

Выходная информация: Таблица БД.

4. RemoveCategory

Функция предназначена для удаления категории из базы данных.

Функция использует Entity Framework.

Входная информация: переменная типа Category.

Выходная информация: Таблица БД.

5. AddMoneyChange

Функция предназначена для добавления новой транзакции в базу данных.

Функция использует Entity Framework.

Входная информация: переменная типа MoneyChange.

Выходная информация: Таблица БД.



#### 6. RemoveMoneyChange

Функция предназначена для удаления транзакции из базы данных.

Функция использует Entity Framework.

Входная информация: переменная типа MoneyChange.

Выходная информация: Таблица БД.

#### 7. LoadData

Функция предназначена для загрузки данных из базы данных на экран пользователя.

Входная информация: данные из таблиц БД.

Выходная информация: ObservableCollection<T>.

#### 8. UpdateData

Данная функция предназначена для обновления информации на пользовательском экране при обновлении данных в БД.

Входная информация: данные из таблиц БД.

Выходная информация: ViewModel.

#### 9. AccountToExcel

Данная функция предназначена для выгрузки информации с одного аккаунта в таблицу Excel.

Входная информация: переменная типа Account.

Выходная информация: Excel файл.

#### 10.AssemblyInfo

Данная функция предназначена для загрузки компонентов приложения перед началом работы.

Входная информация: дескриптор окна приложения.

Выходная информация: UI.

## Иерархическая структура работ:

1. Разработка технического задания
  - 1.1. Сбор требований;
  - 1.2. Определение стадий и этапов разработки
    - 1.2.1. Определение стадий разработки;
    - 1.2.2. Определение сроков разработки;
  - 1.3. Общее описание
    - 1.3.1. Назначение продукта;
    - 1.3.2. Взаимодействие продукта;
    - 1.3.3. Допущения и ограничения продукта;
    - 1.3.4. Определение функций продукта;
2. Разработка приложения
  - 2.1. Backend-разработка
    - 2.1.1. Проектирование базы данных
      - 2.1.1.1. Определение структуры базы данных;
      - 2.1.1.2. Определение связей между сущностями;
      - 2.1.1.3. Определение взаимодействия с базой данных;
    - 2.1.2. Разработка API приложения
      - 2.1.2.1. AddAccount;
      - 2.1.2.2. RemoveAccount;
      - 2.1.2.3. AddCategory;
      - 2.1.2.4. RemoveCategory;
      - 2.1.2.5. AddMoneyChange;
      - 2.1.2.6. removeMoneyChange;
      - 2.1.2.7. LoadData;
      - 2.1.2.8. UpdateData;
      - 2.1.2.9. AccountToExcel;
      - 2.1.2.10. AssemblyInfo;
    - 2.1.3. Сетевое взаимодействие
      - 2.1.3.1. Определение протокола взаимодействия;
      - 2.1.3.2. Обеспечение защищённости соединения;
      - 2.1.3.3. Определение местоположения хранения данных;
    - 2.1.4. Взаимодействие с UI
      - 2.1.4.1. Определение модели привязки данных;
      - 2.1.4.2. Создание механизма обновления данных;

## 2.2. Frontend-разработка

### 2.2.1. Дизайн

#### 2.2.1.1. Разработка макетов страниц

- 2.2.1.1.1. Разработка макета главной страницы;
- 2.2.1.1.2. Разработка макета страницы транзакций;
- 2.2.1.1.3. Разработка макета страницы календаря;
- 2.2.1.1.4. Разработка макета карточки категории;
- 2.2.1.1.5. Разработка макета страницы создания транзакций;
- 2.2.1.1.6. Разработка макета карточки транзакций;
- 2.2.1.1.7. Разработка страницы категорий;
- 2.2.1.1.8. Разработка страницы аккаунтов;

#### 2.2.1.2. Взаимодействие с backend

- 2.2.1.2.1. Определение модели привязки данных;
- 2.2.1.2.2. Настройка механизма обновления данных;
- 2.2.1.2.3. Определение динамического взаимодействия с данными;

#### 2.2.1.3. Разработка общего стиль-кода приложения

- 2.2.1.3.1. Создание логотипа приложения;
- 2.2.1.3.2. Создание иконок элементов;
- 2.2.1.3.3. Определение стиля и размера шрифтов;

## 3. Приемо-сдаточные испытания

### 3.1. Подготовка и проведение демонстрации;

### 3.2. Проведение испытаний;

## 4. Размещение приложения

### 4.1. Аренда сервера;

### 4.2. Развёртывание приложения;

### 4.3. Размещение в магазине приложений;

## 5. Поддержка приложения

### 5.1. Мониторинг работоспособности;

### 5.2. Получение и обработка обратной связи;

### 5.3. Улучшение работы приложения

### 5.4. Добавление новой функциональности;

## Оценка времени выполнения проекта по методу PERT:

Произведем вычисления для определения временных затрат на реализацию проекта. Для этого определим состав работ их оптимистичные, пессимистичные и средние трудозатраты в часах:

Работы	Количество	Оптимистичные трудозатраты	Пессимистичные трудозатраты	Наиболее вероятные трудозатраты
Создание сущностей	7	2	8	3
Создание макетов	7	30	60	50
Создание методов API	10	40	70	55

Посчитаем средние трудозатраты по каждой работе:

$$\text{Создание сущностей} = \frac{8 + 4 \times 3 + 2}{6} = 3.6 \text{ чел.* час.}$$

$$\text{Создание макетов} = \frac{60 + 4 \times 50 + 30}{6} = 48.3 \text{ чел.* час.}$$

$$\text{Создание методов API} = \frac{70 + 4 \times 55 + 40}{6} = 55 \text{ чел.* час.}$$

Посчитаем среднеквадратичную оценку:

$$\text{Создание сущностей} = \frac{8 - 2}{6} = 1 \text{ чел.* час.}$$

$$\text{Создание макетов} = \frac{60 - 30}{6} = 5 \text{ чел.* час.}$$

$$\text{Создание методов API} = \frac{70 - 40}{6} = 5 \text{ чел.* час.}$$

Посчитаем  $E_{\text{общ.}}$ :

$$E_{\text{общ.}} = 7 * 3.6 + 7 * 48.3 + 10 * 55 = 913.3 \text{ чел.* час.}$$

Посчитаем  $СКО_{\text{общ.}}$ :

$$СКО_{\text{общ.}} = \sqrt{7 * 1^2 + 7 * 5^2 + 10 * 5^2} = 20.8 \text{ чел.* час.}$$

Оценка суммарной трудоёмкости проекта с вероятностью 95%:

$$E_{95\%} = 913.3 + 2 * 20.8 = 954.9 \text{ чел.* час.}$$