Поставленная задача требует создать приложение, которое должно иметь такие требования:

1. Возможность отслеживать денежные средства на всех счетах и показывать переводы между счетами. Подсчет наличных денег. Отображение остатка общего счета.
2. Разделение на общую сумму и на конкретный счет. Общие доходы/расходы и по конкретному счету.
3. Планирование бюджета по разделам на день/неделю/месяц/год. Ручное внесение расходов/доходов пользователя по дням.
4. Категории расходов/доходов (здоровье, досуг, дом, продукты, кафе, образование, подарки, одежда и обувь, транспорт, спорт, зарплата, проценты по вкладам, другие покупки/пополнения). Здесь пользователь заранее указывает суммы, которые готов выделить на тот или иной раздел с какого периода и с какой периодичностью (единоразово/раз в неделю/раз в месяц и т.д.) И также пользователем указываются суммы, которые он получает.
5. Отображение осуществленных операций. Процентное соотношение операции к выделенным денежным средствам на данную категорию.
6. Наличие графического отображения расходов/доходов (Графики, столбчатые или круговые диаграммы)
7. Информация о регулярных платежах пользователя. Уведомления о них и внесение в общие расходы по дате списания (Мобильная связь, интернет, кредит, ипотека и т.д.). Информация о запланированных тратах (Отметка в календаре приложения).
8. Анализ курса валют с помощью ИИ, который будет описаться на актуальную информацию одного или нескольких банков. Получение рекомендаций по хранению денежных средств в других валютах, по размену валюты (К примеру, в каком банке или в каком размере стоит разменять валюту).
9. Добавление ИИ для анализа финансов, для получения рекомендаций в распределении бюджета. К примеру, в каком процентном соотношении можно распределить бюджет по различным категориям на основе расходов пользователя. Рекомендации по уменьшению/увеличению выделенной суммы на категорию. Рекомендации по накоплению денежных средств. Рекомендации по распределению бюджета при незапланированных покупках.

**Что нужно для реализации бэкенда**

**1. Выбрать Фреймворк для работы с сервером.**

Перед сервером стоят такие задачи:

1. Быстродействие;
2. Возможность обрабатывать большое количество запросов;
3. Безопасность данных.

Так как 2-курс не имеет большого опыта работы с созданием серверов, то можно выделить такие требования:

1. Простота написания кода;
2. Достаточно легко обучиться;
3. Фреймворк уже имеет в своем арсенале решения множества частых проблем.

Под все критерии подходит Django:

* Невероятно быстрый;
* **Высокая степень безопасности;**
* **Масштабируемость;**
* **Полнофункциональный;**
* **Простое управление;**
* **Гибкость и универсальность;**
* **Расширяемость;**
* **Бесплатный и с открытым исходным кодом;**
* **Простота изучения, благодаря использованию Python;**
* **Разработчики уже решили многие проблемы при создании фреймворка.**

**Пример**:

Создание моделей происходит с использованием встроенного функционала, что значит не требуется создавать модели с нуля, достаточно просто обратиться и задать нужные параметры.



**2. Выбор базы данных для хранения информации.**

Для хранения данных было решено использовать PostgreSQL. Он обладает такими плюсами:

1. Полная SQL-совместимость.
2. Открытый исходный код.
3. Расширенные настройки с возможностью создания собственных плагинов и персонализацией параметров.
4. Возможность индексирования геометрических (в том числе географических) объектов.
5. Наличие расширения PostGIS.
6. Функция MVCC для управления параллельным доступом посредством многоверсионности.
7. Расширяемость, позволяющая масштабировать PostgreSQL с помощью хранимых процедур.
8. Возможность поддерживать некоторые объектно-ориентированные функции.
9. Частые обновления.

Все преимущества делают PostgreSQL отличным вариантом для создания нашего проекта.