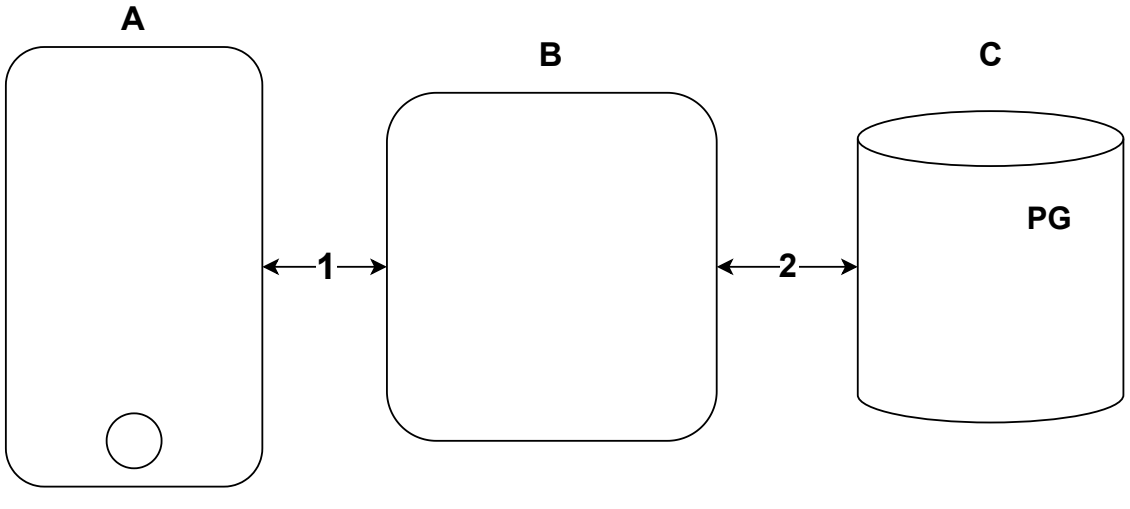
1. Архитектура

Фронтенд – это схематичный веб или мобильное приложение системы. Также фронтенд называют клиентом
Бэкенд – это внутренний сервер системы проекта. Сервер обращается к базе данных с полученными данными от клиента
База данных – это хранилище данных, с которыми будет работать сервер.

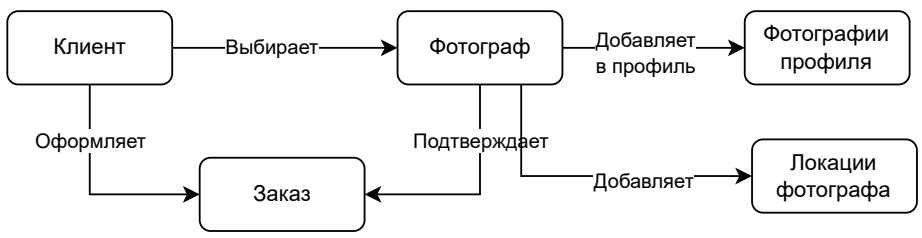


A - фронтенд, мобильное приложение
B - бэкенд/сервер
C - реляционная база данных, PG

1 - протокол взаимодействия http(s)
2 - JDBC

2. Модель данных

Модель данных – это представление данных, атрибутного состава сущностей, как сущности связаны друг с другом. Объект Заказ выходит за рамки рассматриваемой user story, однако считаю полезным ее отобразить для понимания связи Клиент-фотограф в финальном сценарии.



Объект Клиент

Родительская сущность	Атрибут	Описание
Client	-------	Профиль клиента в приложении
	tel_number	Номер телефона
	name	Имя
	surname	Фамилия
	patronymic	Отчество
	city	Город проживания
	email	Почта для связи

Объект Photographer

Родительская сущность	Атрибут	Описание
Photographer	-------	Профиль фотографа
	tel_number	Номер телефона
	name	Имя
	surname	Фамилия
	patronymic	Отчество
	email	Почта для связи
	type	Тип съемок (портрет, лав-стори, семейная и т.п)
	style	Стиль фотографий (фэшн, минимализм, чб и тп)
	rating	Оценка клиентов (рэйтинг)

Объект Заказ

Дочерняя сущность	Атрибут	Описание
Order	-------	Заказ на фотосессию
	client	Кто заказчик фотосессии
	photographer	Кто фотограф фотосесссси
	time_date	Дата и время проведения съемки
	location	Локация съемки
	type	Тип съемки (портрет, лав-стори и тп)
	style	Стиль съемки (фэшн, чб и тп)
	price	Итоговая стоимость заказа

Объект Фотографии

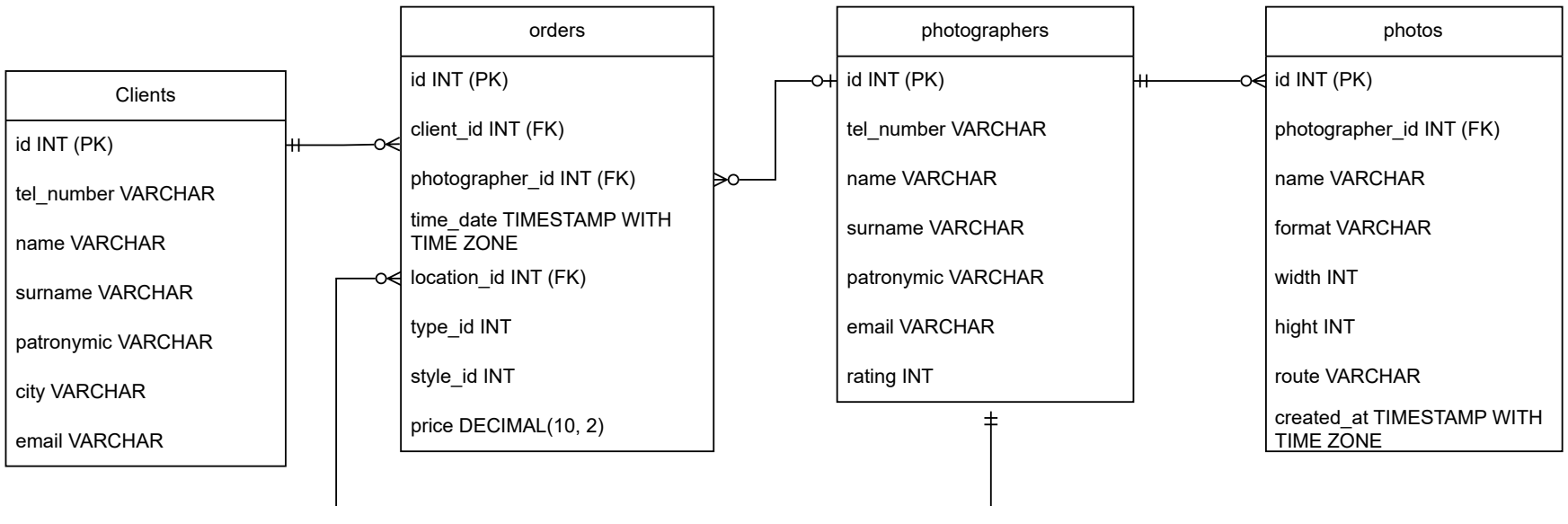
Дочерняя сущность	Атрибут	Описание
Photos	-------	Заказ на фотосессию
	photographer	Фотограф, которому принадлежит фото
	name	наименование файла фотографии
	format	формат фотографии (расширение)
	wight	ширина фото в пкс
	hight	высота фото в пкс
	route	расположение (адрес) фото
	created_at	дата добавления фото в профиль

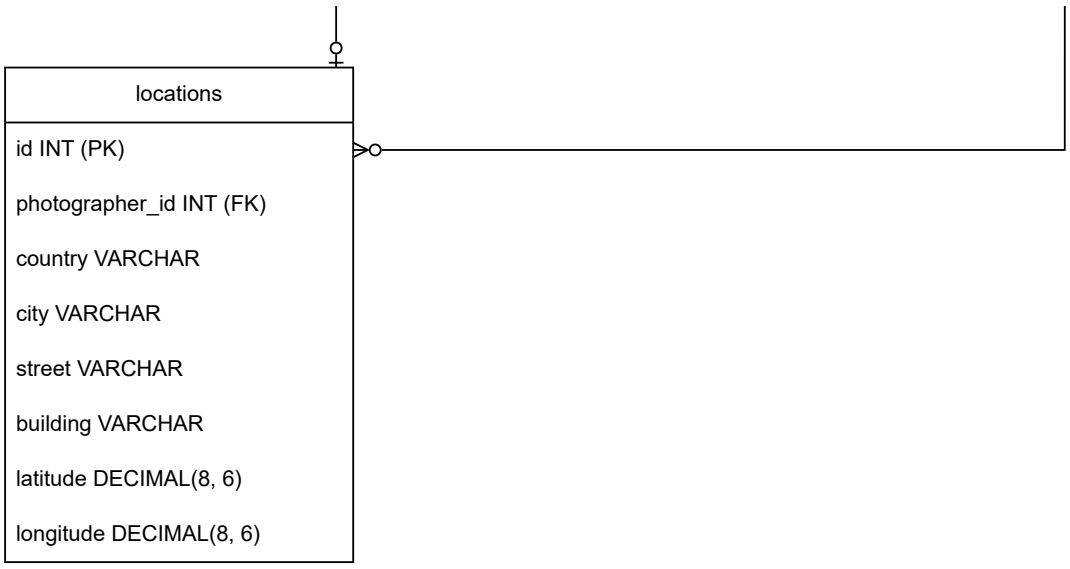
Объект Локации фотографа

Дочерняя сущность	Атрибут	Описание
locations	-------	Локации, которые выбрал фотограф
	photographer	Фотограф, который выбрал локацию
	country	страна
	city	город
	street	улица
	building	номер дома, строения, корпуса
	latitude	широта координаты
	longitude	долгота координаты

3. ERD-диаграмма

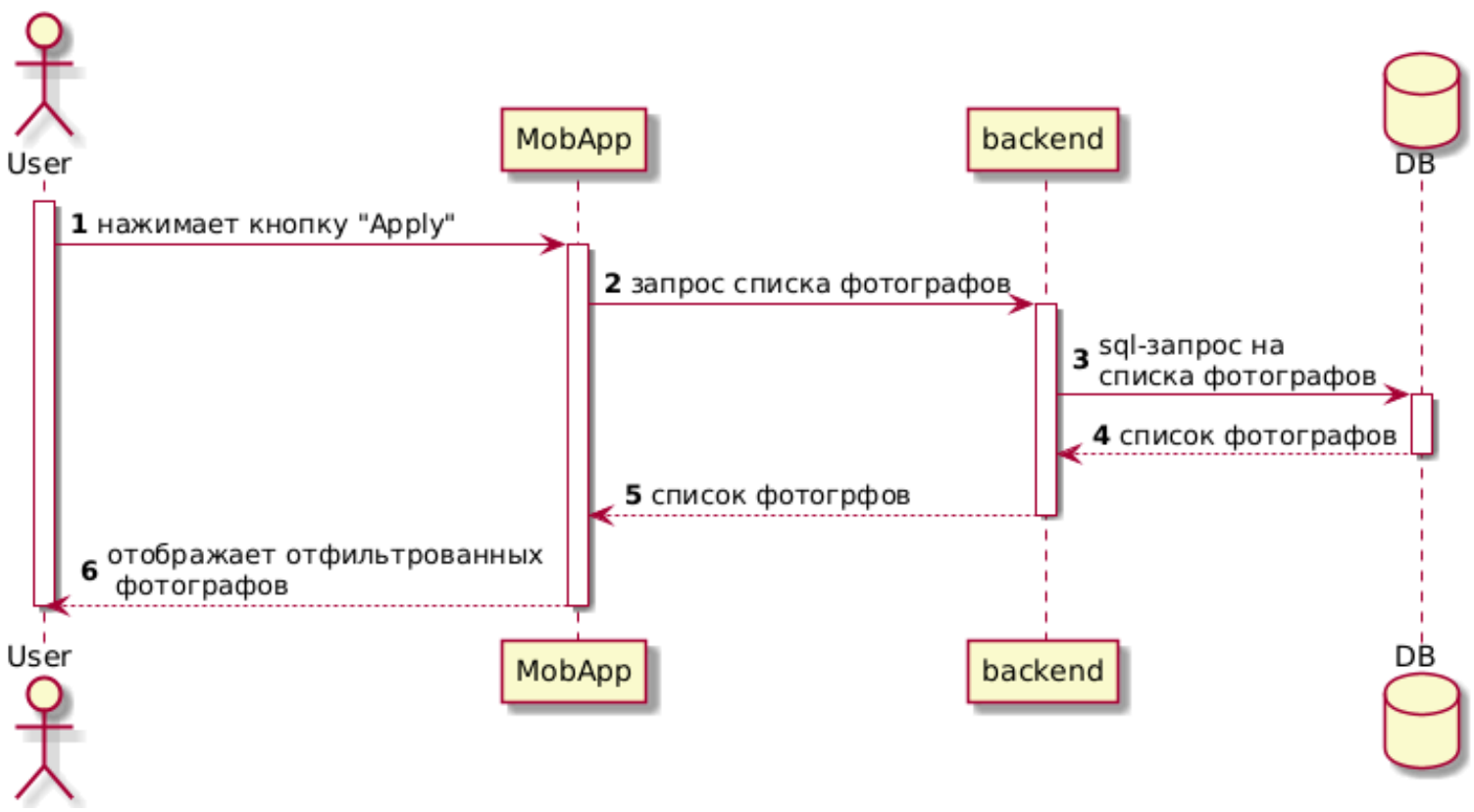
ERD-диаграмма — диаграмма где показано, как разные «сущности» (люди, объекты, концепции и так далее) связаны между собой внутри системы.





4. Sequence diagramm (диаграмма последовательностей)

Диаграмма последовательности — это тип UML-диаграммы, который используется для моделирования взаимодействий между объектами в системе. Она отображает, в каком порядке и как взаимодействуют между собой различные объекты или компоненты во времени. Основное назначение — показать, как запросы и ответы передаются между элементами системы.



№	Описание
1.	Пользователь принимает все введенные параметры фильтрации, нажимая кнопку "Apply"
2.	Мобильное приложение направляет запрос поиска фотографов с параметрами фильтрации по локации
3.	Бэк направляет sql запрос на получение списка фотографов
4.	СУБД выполняет запрос и возвращает список фотографов, соответствующих критериям пользователя
5.	Бэк формирует json со списком фотографов и отправляет в приложение
6.	Приложение отображает список отфильтрованных фотографов

5. REST. Табличный вид

REST API подход использует HTTP-методы (GET, POST, PUT, DELETE и т.д.) для управления сущностями с уникальными URL.

GET

GET/photographers?radius={}&latitude={}&longitude={}
Запрос на получение списка фотографов по фильтру локации.

Request

Query параметр	Тип данных	Описание	Обязательность
radius	decimal	длина радиуса, покрываемой области по поиску фотографа	нет
latitude	decimal	широта центра окружности для поиска фотографов	нет
longitude	decimal	долгота центра окружности для поиска фотографов	нет

Response

Code 200

Параметр	Тип данных	Описание	Обязательность
total_count	int	общее количество найденных профилей фотографов	да
photographers	array object	массив объектов с отфильтрованными по локации фотографами	да
name	string	имя фотографа	да
patronymic	string	отчество фотографа	да
surname	string	фамилия фотографа	да
rating	int	средний рейтинг фотографа	да
avatar	string	url, расположение аватарки фотографа	да
location	string	локация фотографа (город+улица) наиболее близкие к входным параметрам	да
latitude	decimal	широта наиболее близкой локации фотографа к центру поиска (для отображения на карте)	да
longitude	decimal	долгота наиболее близкой локации фотографа к центру поиска (для отображения на карте)	да

6. Критерии приёмки

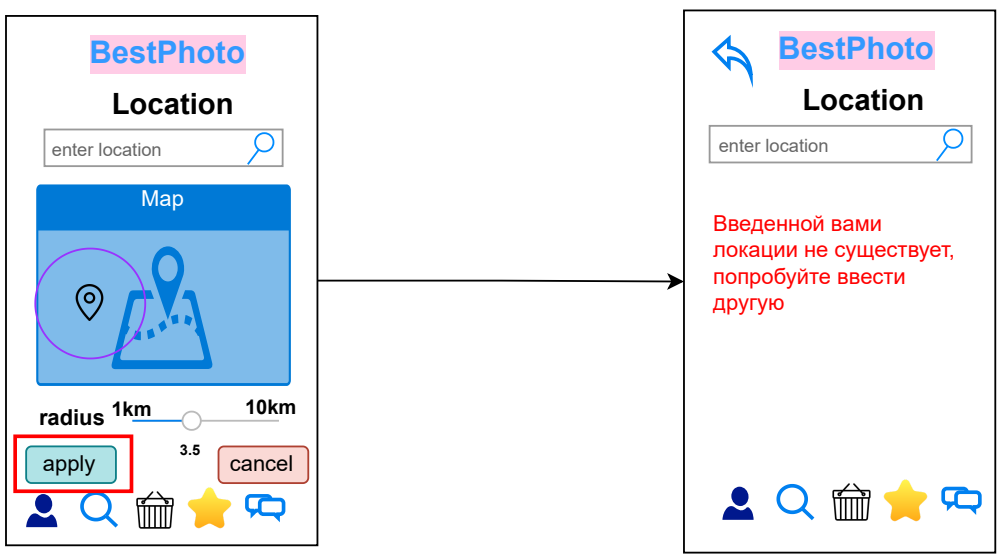
Критерии приёмки - это формализованные требования, которые описывают, каким образом система должна работать, чтобы пользователь или заказчик признали функциональность выполненной. Критерии приёмки состоят из кейсов "Дано - Когда (событие или триггер) - Тогда (реакция системы, ожидаемый результат)". При составлении КП аналитик должен подумать, что пользователь может "сломать" в процессе прохождения основного сценария, предусмотреть нестандартное поведение пользователя. В качестве кейсов рассматриваются альтернативные и исключительные сценарии. Основной потребитель этой документации - тестировщики.

Шаблон описания кейса

Функциональность: формулировка US
Номер кейса: N
Дано: предварительные условия или начальный контекст
Когда: событие или триггер
Тогда: ожидаемый результат.

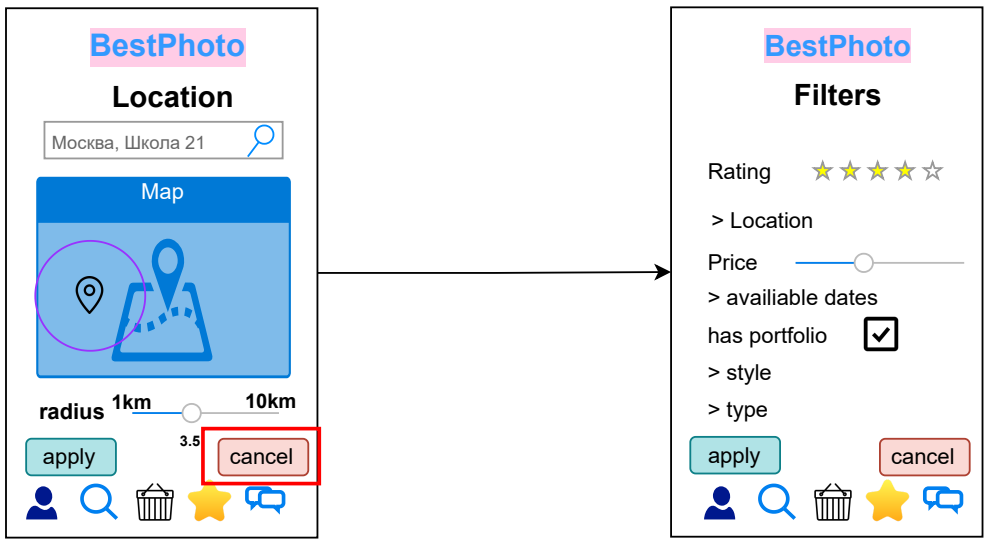
КП для приложения BestPhoto

Функциональность: поиск фотографов (доступных в приложении) по фильтру города/локации
Номер кейса: 1
Дано: пользователь выбирает локацию для поиска фотографа



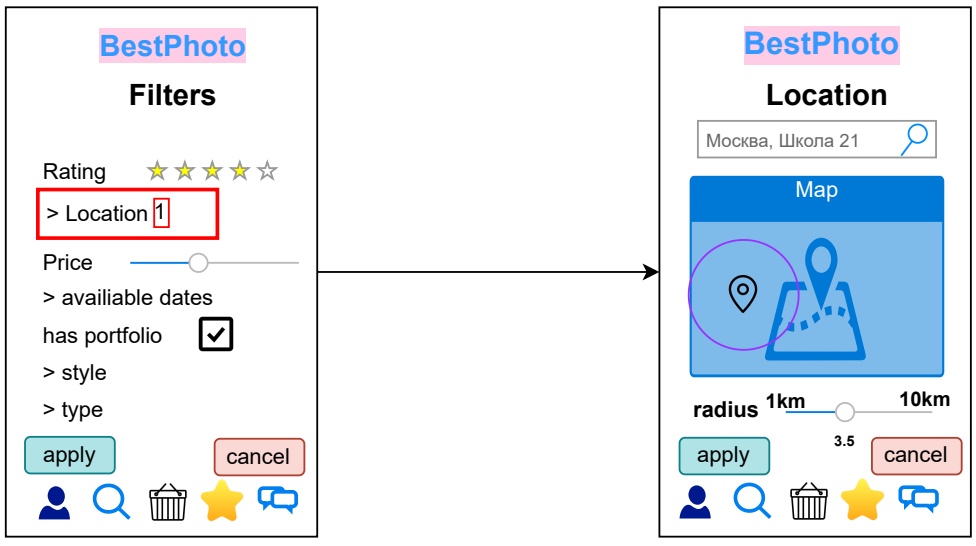
Когда: пользователь ввел несуществующую локацию и нажал кнопку "apply"
Тогда: приложение отображает сообщение об ошибке "Введенной вами локации не существует, попробуйте ввести другую"

Функциональность: поиск фотографов (доступных в приложении) по фильтру города/локации
Номер кейса: 2
Дано: пользователь выбирает локацию для поиска фотографа



Когда: пользователь нажал кнопку "cancel"
Тогда: приложение сбрасывает введенные на данном экране данные и открывает экран с выбором фильтров

Функциональность: поиск фотографов (доступных в приложении) по фильтру города/локации
Номер кейса: 3
Дано: пользователь ввел данные по локации и нажал кнопку "apply", приложение отображает экран с фильтрами.



Когда: пользователь снова нажимает на поле "Location"
Тогда: приложение открывает экран, где отображаются введенные раннее данные.

7. Нефункциональные требования

НФТ относятся к атрибутам качества системы, которые определяют, как она работает, а не что она делает. В НФТ описывают требования к:

- производительности
- надежности
- безопасности
- удобству использования
- масштабируемости

Нефункциональные требования	
Производительность	1. Страница поиска фотографов должна открываться не более 2 секунд. 2. Запрос поиска фотографов GET /photographers должен выдерживать нагрузку 1 rps.
Надежность	1. Система должна быть доступна 99% времени.
Безопасность	1. Личные данные пользователей должны быть защищены 2. Пользовательские пароли хранятся в зашифрованном виде
Удобство использования	1. Пользователь должен зарегистрироваться в системе не более чем за 2 минуты без использования инструкции
Масштабируемость	1. Система может поддерживать рост базы данных на 10% в год