**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Образовательная программа «Программная инженерия»

|  |  |  |
| --- | --- | --- |
| СОГЛАСОВАНО |  | УТВЕРЖДАЮ |
| Приглашенный преподаватель департамента программной инженерии |  | Академический руководитель  Образовательной программы  «Программная инженерия»  Старший преподаватель департамента  программной инженерии |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Г.М. Сосновский |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Н.А. Павлочев |
| «1» апреля 2024 г. |  | «1» апреля 2024 г. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | |  |  |
| **«Rewind» Приложение для коллективного медиа-потребления** |  |
| **Пояснительная записка** |  |
| **ЛИСТ УТВЕРЖДЕНИЯ** |  |
| **RU.17701729.12.20-01 81 01-1-ЛУ** |  |
|  |  |
|  |  |
|  | Исполнитель |
|  | студент группы БПИ228 |
| \_\_\_\_\_\_\_\_\_/ Д. Д. Рахманов / | |
| «1» апреля 2024 г. | |
|  |  |
|  | |
|  | |

**Москва 2024**

|  |  |  |
| --- | --- | --- |
| УТВЕРЖДЕН  RU.17701729.12.20-01 81 01-1 |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  | **«Rewind» Приложение для коллективного медиа-потребления** |  |
|  | **Пояснительная записка** |  |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | | **RU.17701729.12.20-01 81 01-1** |  |
| **Листов 38** |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Москва 2024**

# ВВЕДЕНИЕ

## Наименование

Наименование программы: «“Rewind” Приложение для коллективного медиа-потребления».

Наименование программы на английском языке: «“Rewind” Application for Collective Media Consumption».

Краткое наименование программы: «Rewind».

## Документ, на основании которого ведется разработка

Учебный план подготовки бакалавров по направлению 09.03.04 «Программная инженерия» и утвержденная академическим руководителем программы тема курсового проекта.

# НАЗНАЧЕНИЕ РАЗРАБОТКИ

## Назначение программы

## Функциональное назначение

Приложение предназначено для создания группы и коллекционирования в них общих медиа-объектов в виде фотографий и цитат. Функционал приложения предоставляет возможность подбора случайного медиа-объекта по заданным фильтрам, добавления и удаления медиа-объекта, присваивания ему тегов.

## Эксплуатационное назначение

Приложение предназначено для эксплуатации в целях создания групп и совместного накопления в них значимых моментов в форме фотографий и цитат.

Целевой аудиторией приложения являются знакомые между собой люди: друзья, члены семьи, коллеги и все те, кому важно сохранить и иметь возможность вспомнить общие моменты.

Задача приложения состоит в том, чтобы укрепить связь между людьми и привнести разнообразие в их повседневную жизнь. Просмотр случайных воспоминаний может стать интересным и увлекательным занятием для участников.

## Краткая характеристика области применения

Приложение предназначено для создания и поддержания эмоциональной связи между людьми с помощью коллекционирования и просмотра общих ярких моментов в виде фотографий и цитат.

# ТРЕБОВАНИЯ К ПРОГРАММЕ

## Постановка задачи на разработку программы

Серверная часть должна представлять собой приложение, реализованное на языке программирования C# с использованием фреймворка .NET. Необходимо спроектировать надёжную структуру базы данных для эффективного управления данными. Для взаимодействия с базой данных должно быть предусмотрено использование СУБД MySQL. Требуется использовать ORM Entity Framework Core для работы с БД. C#-приложение должно включать в себя Unit-тесты, которые необходимы для обеспечения высокого уровня качества кода, упрощения его тестирования и обнаружения ошибок на ранних этапах разработки. Для изолирования слоёв программы друг от друга необходимо использовать Repository Pattern. Для более эффективного использования ресурсов сервера и увеличения производительности работы с данными требуется использовать представления сущностей.

## Описание задач и их решения

## Проектирование структуры БД

Во время разработки было необходимо спроектировать структуру базы данных для хранения данных, используемых пользователем в iOS-приложении. Цель состоит в том, чтобы при помощи правильный структуры БД обеспечить согласованность информации, устранить хранение избыточных данных, повысить эффективность выполнения запросов и производительность работы приложения.

Варианты выполнения задачи:

1. Создание отдельных таблиц для различных сущностей с их атрибутами и настройка их взаимосвязей. Использование EF Core ORM с этим подходом обеспечит удобство в работе с сущностями и отношениями между ними.
2. Использование EAV (Сущность-Атрибут-Значение) - универсальное решение, которое позволяет хранить различные типы данных в одной таблице, добавляя новые атрибуты без изменения структуры базы данных. Однако, такой подход может усложнить создание запросов и выполнение операций с данными.

Для решения данной задачи был выбран первый вариант — использование нескольких таблиц с использованием связей. Такое решение было принято, потому что это стандартный и достаточно популярный подход, который поспособствовал увеличению понимания и удобства обслуживания структуры базы данных. Использование этого подхода с EF Core упрощает работу с сущностями, автоматически учитывая отношения между таблицами и упрощая реализацию логики запросов.

Чтобы улучшить процесс управления данными, можно было бы добавить индексы к таблицам, чтобы ускорить поиск нужной и часто запрашиваемой клиентом информации. В свою очередь, для отслеживания изменений данных можно использовать механизмы логирования, что поможет при анализе данных или поиске изменений или проблем.

## Тестирование работы серверной части

Необходимо было провести тестирование и отладку серверной части приложения с использованием Unit-Test’ов при помощи фреймворка xUnit. Это необходимо для обеспечения корректности работы каждого отдельного модуля и компонента приложения, а также для выявления и исправления возникающих ошибок.

Фреймворк xUnit предоставляет все необходимые инструменты для проведения автоматизированного тестирования, что позволяет эффективно проверить работу отдельных компонентов.

Процесс реализация тестирования:

1. Написание тестовых сценариев для проверки работы отдельных модулей, классов и методов серверной части приложения.
2. Запуск тестов для проверки корректности работы серверной части и выявления возможных ошибок.
3. Использование результатов тестирования для выявления и исправления ошибок в коде.

Для генерации атрибутов каждой из сущностей был выбран инструмент AutoFixture, который может создать объект, заполненный случайно сгенерированными при помощи GUID данными. Также, AutoFixture позволяет настроить генерацию данных у объектов, если возникает такая необходимость.

Для тестирования приложения используется In-Memory DataBase, сгенерированная при помощи GUID для каждого отдельного теста. Это необходимо, чтобы для каждого из написанных тестов код работал с отдельной базой данных, это поможет обеспечить изоляцию данных, используемых в разных тестах, друг от друга. In-Memory базы данных позволяют сохранять данные локально, не занимая ресурсов сервера, и тестировать приложение в условиях, максимально приближенных к реальной среде без необходимости подключения к реальной базе данных.

Для еще большего приближения к реальным условиям эксплуатации приложения можно внедрить реалистичную генерацию каждого из атрибутов сущностей, а не генерировать их при помощи GUID. Для того, чтобы сделать содержание тестов более понятным для других разработчиков, можно внедрить использование BDD-фреймворков, которые позволят описывать поведение и ожидаемый результат работы каждого теста.

* + 1. **Разделение кода на изолированные слои при помощи Repository Pattern**

Необходимо выполнить абстрагирование слоев приложения для обеспечения гибкости поддержки, расширения и тестирования кода. Это важно для создания модульной и гибкой архитектуры приложения. Repository Pattern, применяемый при реализации Clean Architecture, используется для абстрагирования слоя доступа к данным от остальных слоев программы. Он используется для создания слоя репозиториев, который предоставляет унифицированный интерфейс для работы с данными независимо от источника данных.

Варианты выполнения задачи:

* 1. Использование прямого подхода через ORM при получении доступа к данным без абстракций
  2. Разделение кода на слои и абстрагирование слоя доступа к данным от других частей программы при помощи Repository Pattern

При разработке проекта было решено использовать Repository Pattern вместо прямого использования ORM без абстракций обеспечивает изоляцию слоя доступа к данным от других частей приложения. Задача была выполнена путем создания отдельных репозиториев для контроллеров, каждый из который предоставляет интерфейс для доступа к данным независимо от их источника.

Для усовершенствования решения можно рассмотреть внедрение других известных при разработке паттернов проектирования, таких как UnitOfWork, который помогает упростить работу со слоем репозитория и дает уверенность, что все репозитории будут использовать один и тот же DbContext.

* + 1. **Использование представлений сущностей для сбора информации об объектах**

Необходимо использовать представления сущностей для сбора только необходимой информации об объекте. Это позволит улучшить производительность приложения и снизить нагрузку на базу данных, так как будут извлекаться только нужные для отображения и использования данные, а не все столбцы подряд. Также, использование представлений поспособствует сокрытию многочисленных полей у сущности, что упрощает работу другим разработчикам.

Варианты решения задачи:

1. Создание дополнительных представлений в базе данных, содержащих только необходимые столбцы для конкретных сценариев использования.
2. Использование проекций при запросах к базе данных, чтобы выбирать только нужные поля из основных таблиц.
3. Извлечение всех данных из основных таблиц, даже если некоторые данные не нужны.

Для реализации этого подхода были созданы дополнительные представления в базе данных (например, UserView, GroupView, MediaView), содержащие только необходимые столбцы для конкретных сценариев использования. Эти представления могут быть использованы в запросах к базе данных для извлечения только нужных пользователю данных.

Такая реализация использует данные намного эффективнее, чем проекции или извлечение всей информации о сущности, ведь лишние и ненужные в определённом сценарии данные не запрашиваются из базы данных и не используются в дальнейшем.

В целях увеличения производительности при поиске и фильтрации данных в таблицах можно рассмотреть возможность создания индексов. Причем такое решение сохранит уникальность и целостность значений.

## Описание выбора состава технических и программных средств

## Состав технических средств

Для работы серверной части приложения потребуются следующие технические средства:

* + - 1. Операционная система Windows 10 и выше;
      2. C# .NET версии 6.0;
      3. JetBrains Rider 2023+ или Visual Studio 2022+.

## Состав программных средств

Для разработки серверной части проекта был выбран язык программирования C# с использованием фреймворка .NET 6.0. В качестве СУБД была выбрана MySQL, а для взаимодействия с базой данных было выбрано ORM Entity Framework Core.

## Обоснование выбора технических и программных средств

Выбор данного набора технологий обусловлен следующими причинами:

1. С# - очень широко распространенный язык программирования, который имеет многочисленное количество удобных и проверенных инструментов разработки.
2. Выбор MySQL был сделан исходя из ее популярности, оптимальной производительности и возможности масштабирования, которая может обеспечить эффективное хранение и управление данными приложения.
3. Entity Framework Core - это мощный ORM, который обеспечивает простое и эффективное взаимодействие с базой данных, упрощая разработку и поддержку кода.

Такой стек технологий позволяет создать производительное решение, которое можно масштабировать и поддерживать в будущем. Использование EF Core упрощает работу с базой данных, MySQL обеспечивает надежное хранение данных, а C# предоставляет все необходимые инструменты для работы с Web-API.

## Описание и обоснование выбора способов организации входных и выходных данных

## Описание организации входных и выходных данных

Входные данные приложения представляют собой разнообразную информацию, получаемую от пользователя в процессе взаимодействия с приложением. Это включает в себя не только личные данные пользователя, такие как имя пользователя, изображение профиля и адрес электронной почты, но также и данные о группах, в которых состоит пользователь, загруженные медиа-объекты и теги. Эти входные данные проходят обработку, после этого данные записываются в базу данных для последующего использования.

Выходные данные приложения представляют собой отображаемый интерфейс, который формируется на основе полученных от сервера данных. Клиент получает данные от сервера в формате JSON. Перед тем, как данные будут отправлены на клиентское устройство для отображения пользователю, они могут быть также подвергнуты определенным преобразованиям или фильтрации в зависимости от требований пользователя.

* + 1. **Обоснование выбора метода организации входных и выходных данных**

Выбор указанных выше входных данных был обоснован эффективностью в получении информации от пользователя, которая затем передается на сервер. Сервер получает данные из базы данных, что позволяет дальнейшей обработке данных происходить точно и быстро. Представление выходных данных, поступивших от сервера, в виде удобного интерфейса существенно упростит навигацию пользователя в приложении.

# ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

## Ориентировочная экономическая эффективность

В рамках проекта расчет экономической эффективности программного продукта не производился.

## Предполагаемая годовая потребность

Данная программа будет востребована в кругу друзей, знакомых или коллег, которые хотят коллекционировать и вспоминать яркие моменты их общения.

## Экономические преимущества разработки по сравнению с лучшими отечественными и зарубежными образцами или аналогами

Цель приложения «Rewind» – предоставить возможность создания групп людей и совместного накопления в них значимых моментов в форме фотографий и цитат.

В таблице 1 представлен сравнительный анализ приложения «Rewind» с лучшими аналогами.

Таблица 1 – Сравнительный анализ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Категория | Критерии | Аналоги | | | Rewind |
| Locket | Фотопленка iOS | TimeHop |
| Доступность | Наличие платной подписки | + | - | - | - |
| Дизайн | Смена иконки приложения | - | - | - | + |
| Пользование приложением с друзьями | Возможность объединяться в группы | - | - | - | + |
| Общая галерея медиа-объектов у нескольких пользователей | + | + | - | + |
| Возможность видеть автора медиа-объекта | + | + | - | + |

Продолжение таблицы 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Пользование приложением с друзьями | Просмотр воспоминаний происходит по случайным датам | - | + | - | + |
| Возможность удалить медиа-объект | - | + | - | + |
| Фильтрация медиа-объектов | По типу медиа-объекта | - | + | - | + |
| По избранным | - | + | - | + |
| Особый функционал | Добавление цитаты с помощью генерации изображения с текстом цитаты и автором | - | - | - | + |
| Автоматическая генерация тегов при нажатии на кнопку | - | - | - | + |
| Возможность пометить медиа-объект избранным | - | + | - | + |

В таблице 2 представлен сравнительный анализ «Rewind» с косвенными аналогами.

Таблица 2 – Сравнительный анализ (косвенные аналоги)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Категория | Критерии | Аналоги | | | | Rewind |
| Telegram | VK | Yandex Disk | Google Disk |  |
| Доступность | Наличие платной подписки | + | + | + | + | - |
| Дизайн | Смена иконки приложения | + | + | - | - | + |
| Пользование приложением с друзьями | Возможность объединяться в группы | + | + | + | + | + |

Продолжение таблицы 2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Пользование приложением с друзьями | Общая галерея медиа-объектов у нескольких пользователей | + | + | + | + | + |
| Возможность видеть автора медиа-объекта | + | + | + | + | + |
| Просмотр воспоминаний происходит по случайным датам | - | - | - | - | + |
| Возможность удалить медиа-объект | + | - | + | + | + |
| Фильтрация медиа-объектов | По типу медиа-объекта | - | - | - | + | + |
| По избранным | - | - | - | + | + |
| Особый функционал | Добавление цитаты в галерею с помощью генерации изображения с текстом цитаты и автором | - | - | - | - | + |
| Автоматическая генерация тегов при нажатии на кнопку | - | - | - | - | + |
| Возможность пометить медиа-объект избранным | - | + | + | + | + |

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Microsoft Learn [Электронный ресурс] / Документация по C#. Режим доступа: <https://learn.microsoft.com/ru-ru/dotnet/csharp/> (Дата обращения: 11.02.2024).
2. Microsoft Learn [Электронный ресурс] / Документация по .NET. Режим доступа: <https://learn.microsoft.com/ru-ru/dotnet/> (Дата обращения: 11.02.2024).
3. MySQL [Электронный ресурс] / MySQL Documentation. Режим доступа: <https://dev.mysql.com/doc/> (Дата обращения: 11.02.2024).
4. ГОСТ 15150-69 Машины, приборы и другие технические изделия. Исполнения для различных климатических районов. Категории, условия эксплуатации, хранения и транспортирования в части воздействия климатических факторов внешней среды. – М.: Издательство стандартов, 1997.
5. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.404-79 Пояснительная записка. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.505-79 Руководство оператора. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.301-79 Программа и методика испытаний. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.401-78 Текст программы. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
10. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.

# ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ

Таблица 1. Описание и функциональное назначение классов

|  |  |
| --- | --- |
| Класс | Назначение |
| ChangeGroupController | Отвечает за управление запросами и обработку действий, связанных с изменением информации о группе |
| GroupsController | Отвечает за управление запросами и обработку действий, связанных с взаимодействием с группами |
| MediaController | Отвечает за управление запросами и обработку действий, связанных с взаимодействием с медиа-объектами |
| TagsController | Отвечает за управление запросами и обработку действий, связанных с взаимодействием с тегами |
| ChangeUserController | Отвечает за управление запросами и обработку действий, связанных с изменением информации аккаунта пользователя |
| LoginController | Отвечает за управление запросами и обработку действий, связанных с входом в аккаунт |
| RegisterController | Отвечает за управление запросами и обработку действий, связанных с регистрацией нового аккаунта |
| UsersController | Отвечает за управление запросами и обработку действий, связанных с взаимодействием пользователя со своим аккаунтом |
| TagsRepository | Отвечает за абстрагирование доступа к данным от их конкретного источника |
| DataContext | Предоставляет абстрактный интерфейс для выполнения операций, связанных с сущностью Tag в базе данных |
| Group | Отвечает за сущность группы |
| Media | Отвечает за сущность медиа-объекта |
| Tag | Отвечает за сущность тега |
| User | Отвечает за сущность пользователя |
| Extensions | Отвечает за пользовательские расширения классов |
| EmailRequest | Отвечает за данные о email, который должен быть передан в теле URL-запроса, когда выполняется операция изменения электронной почты |
| PasswordRequest | Отвечает за захешированный пароль, который должен быть передан в теле URL-запроса, когда выполняется операция изменения пароля |
| TextRequest | Отвечает за строковые данные, которые должны быть переданы в теле URL-запроса, когда выполняется операция, для выполнения которой необходима лишь строка текста |
| LoadMediaRequest | Отвечает за набор данных, которые должны быть переданы в теле URL-запроса, когда выполняется операция загрузки нового медиа-объекта в галерею группы |
| MediaRequest | Отвечает за данные о медиа-объекте, которые должны быть переданы в теле URL-запроса, когда выполняется операция изменения аватарки группы или пользователя |
| TagsRequest | Отвечает за массив тегов, которые должны быть переданы в теле URL-запроса, когда выполняется операция добавления одного или нескольких тегов к определенному медиа-объекту |
| UserLoginRequest | Отвечает за набор данных, которые должны быть переданы в теле URL-запроса, когда выполняется операция авторизации пользователя |
| UserRegisterRequest | Отвечает за набор данных, которые должны быть переданы в теле URL-запроса, когда выполняется операция регистрации пользователя |
| CreateGroupRequest | Отвечает за набор данных, которые должны быть переданы в теле URL-запроса, когда выполняется операция создания новой группы |
| FilterSettings | Отвечает за набор данных, которые должны быть переданы в теле URL-запроса, когда выполняется операция фильтрации медиа-объектов |
| RewindScreenDataResponse | Отвечает за сбор и представление данных, которые будут возвращены в ответ запрос о получении информации о главном экране при открытии приложения |
| EmailSender | Отвечает за методы, необходимые для отправки сообщения с кодами верификации на email пользователя |
| GlobalRoutePrefixMiddleware | Отвечает за добавление глобального префикса к путям в URL-запросах |
| SqlService | Отвечает за взаимодействие с БД через прямые SQL-запросы там, где в этом есть необходимость |
| UserService | Отвечает за функционал, связанный с обработкой пользовательских данных и генерацией кода подтверждения перед отправкой на email пользователя |
| GroupView | Отвечает за сбор и представление данных, которые будут возвращены в ответ запрос о получении информации о группе |
| SmallGroupInfoResponse | Отвечает за сбор и представление данных, которые будут возвращены в ответ запрос о получении наименее подробной информации о группе |
| GroupInfoResponse | Отвечает за сбор и представление данных, которые будут возвращены в ответ запрос о получении наиболее подробной информации о группе |
| MediaView | Отвечает за сбор и представление данных, которые будут возвращены в ответ запрос о получении наименее подробной информации о медиа-объекте |
| BigMediaInfoResponse | Отвечает за сбор и представление данных, которые будут возвращены в ответ запрос о получении информации о медиа-объекте |
| LargeMediaInfoResponse | Отвечает за сбор и представление данных, которые будут возвращены в ответ запрос о получении наиболее подробной информации о медиа-объекте |
| UserView | Отвечает за сбор и представление данных, которые будут возвращены в ответ запрос о получении информации о пользователе |

# ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ, МЕТОДОВ И СВОЙСТВ

Таблица 2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Описание полей, методов и свойств класса ChangeGroupController | | | | | |
| Поля | | | | | |
| Наименование | Модификатор доступа | Тип аргумента | | Назначение | |
| \_context | private | DataContext | | Контекст базы данных | |
| \_groupsController | private | IGroupsController | | Экземпляр контроллера, который отвечает за взаимодействие с определенной группой | |
| \_sqlService | private | ISqlService | | Экземпляр класса, который отвечает за взаимодействие с БД через прямые SQL-запросы при необходимости | |
| Методы | | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | | Назначение |
| ChangeName | public | Group | TextRequest request,  int groupId | | Меняет название группы |
| ChangeImage | public | void | MediaRequest request,  int groupId | | Меняет аватарку группы |

Таблица 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Описание полей, методов и свойств класса GroupsController | | | | |
| Поля | | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение | |
| \_context | private | DataContext | Контекст базы данных | |
| \_usersController | private | IUsersController | Экземпляр контроллера, который отвечает за взаимодействие с определенным пользователем | |
| Методы | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| GetGroups | public | IEnumerable<Group> | - | Возвращает все группы |
| GetRandomMedia | public | BigMediaInfoResponse | int groupId,  int userId,  bool images = true,  bool quotes = true,  bool onlyFavorites = false | Возвращает информацию о рандомном медиа-объекте в текущей группе и с выбранными фильтрами |
| GetGroupInfoById | public | GroupInfoResponse | int groupId,  int userId,  int dataSize | Возвращает информацию о группе по ее идентификатору |
| GetInitialRewindScreenData | public | RewindScreenDataResponse | int groupId,  int userId | Возвращает нужную для отображения главного экрана приложения информацию |
| GetGroupsByUser | public | IEnumerable<GroupView> | int userId | Возвращает все группы, в которых числится определенный пользователь |
| GetUsersByGroup | public | IEnumerable<UserView> | int groupId | Возвращает всех пользователей, которые числятся в определенной группе |
| GetMediaByGroup | public | IEnumerable<MediaView> | int groupId,  int mediaId | Возвращает все медиа-объекты определенной группы |
| CreateGroup | public | int | CreateGroupRequest request | Создает группу и возвращает идентификатор новой группы |
| AddUserToGroup | public | SmallGroupInfoResponse | int groupId,  int userId | Добавляет пользователя в существующую группу и возвращает информацию об этой группе |
| DeleteUserFromGroup | public | void | int groupId,  int userId | Удаляет пользователя из группы |
| DeleteGroup | public | void | int groupId | Удаляет группу |
| GetGroupById | public | Group | int groupId | Возвращает саму группу по ее идентификатору |
| GetGroupInfo | public | GroupInfoResponse | Group group,  int groupId,  int userId,  int dataSize | Возвращает всю информацию о группе по ее идентификатору |
| GetSmallGroupInfo | public | SmallGroupInfoResponse | Group group | Возвращает минимальное количество информацию о группе по ее идентификатору |

Таблица 4

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Описание полей, методов и свойств класса MediaController | | | | | | |
| Поля | | | | | | |
| Наименование | Модификатор доступа | | Тип аргумента | | Назначение | |
| \_context | private | | DataContext | | Контекст базы данных | |
| \_groupsController | private | | IGroupsController | | Экземпляр контроллера, который отвечает за взаимодействие с определенной группой | |
| \_usersController | private | | IUsersController | | Экземпляр контроллера, который отвечает за взаимодействие с определенным пользователем | |
| \_sqlService | private | | SqlService | | Экземпляр класса, который отвечает за взаимодействие с БД через прямые SQL-запросы при необходимости | |
| \_tagsRepository | private | | ITagsRepository | | Экземпляр репозитория, который отвечает за промежуточный слой взаимодействия с тегами у медиа-объектов | |
| Методы | | | | | | |
| Имя | Модификатор доступа | Тип | | Аргументы | | Назначение |
| GetMediа | public | IEnumerable<Media> | | - | | Возвращает все медиа-объекты |
| GetMediaById | public | Media | | int mediaId | | Возвращает медиа-объект по его идентификатору |
| GetMediaInfoById | public | LargeMediaInfoResponse | | int mediaId,  int userId | | Возвращает информацию о медиа-объекте по его идентификатору |
| LikeMedia | public | Media | | int userId,  int mediaId | | Осуществляет добавление медиа-объекта в Избранные для определённого пользователя |
| UnlikeMedia | public | Media | | int userId,  int mediaId | | Осуществляет удаление медиа-объекта из Избранных для определённого пользователя |
| LoadMediaToGroup | public | void | | LoadMediaRequest mediaRequest,  int groupId,  int authorId | | Осуществляет загрузку медиа-объекта в определенную группу |
| UnloadMediaFromGroup | public | void | | int mediaId,  int groupId | | Осуществляет удаление медиа-объекта из определенной группы |

Таблица 5

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Описание полей, методов и свойств класса TagsController | | | | | |
| Поля | | | | | |
| Наименование | Модификатор доступа | Тип аргумента | | Назначение | |
| \_mediaController | private | IMediaController | | Экземпляр контроллера, который отвечает за взаимодействие с медиа-объектами | |
| \_tagsRepository | private | ITagsRepository | | Экземпляр репозитория, который отвечает за промежуточный слой взаимодействия с тегами у медиа-объектов | |
| Методы | | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | | Назначение |
| GetTags | public | IEnumerable<Tag> | - | | Возвращает все теги |
| GetTagsByMediaId | public | IEnumerable<Tag> | int mediaId | | Возвращает все теги, принадлежащие определенному медиа-объекту |
| AddTags | Public | IEnumerable<Tag> | TagsRequest tagsRequest,  int mediaId | | Осуществляет добавление одного или нескольких тегов к медиа-объекту |
| DeleteTag | public | void | TextRequest textRequest,  int mediaId | | Осуществляет удаление тега для медиа-объекта |

Таблица 6

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Описание полей, методов и свойств класса ChangeUserController | | | | | |
| Поля | | | | | |
| Наименование | Модификатор доступа | Тип аргумента | | Назначение | |
| \_context | private | DataContext | | Контекст базы данных | |
| \_usersController | private | IUsersController | | Экземпляр контроллера, который отвечает за взаимодействие с определенным пользователем | |
| \_sqlService | private | SqlService | | Экземпляр класса, который отвечает за взаимодействие с БД через прямые SQL-запросы при необходимости | |
| Методы | | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | | Назначение |
| ChangeName | public | void | int userId,  TextRequest request | | Изменяет имя пользователя |
| ChangeEmail | public | void | int userId,  EmailRequest request | | Изменяет адрес электронной почты пользователя |
| ChangePassword | public | void | int userId,  PasswordRequest request | | Изменяет пароль пользователя |
| ChangeAppIcon | public | void | int userId,  string newIcon | | Изменяет выбранную иконку приложения пользователя |
| ChangeProfileImage | public | void | int userId,  MediaRequest mediaRequest | | Изменяет аватарку пользователя |

Таблица 7

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Описание полей, методов и свойств класса LoginController | | | | | |
| Поля | | | | | |
| Наименование | Модификатор доступа | Тип аргумента | | Назначение | |
| \_usersController | private | IUsersController | | Экземпляр контроллера, который отвечает за взаимодействие с определенным пользователем | |
| Методы | | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | | Назначение |
| CheckEmail | public | void | string email | | Проверяет, существует ли в БД пользователь с таким email |
| Login | public | User | UserLoginRequest request | | Осуществляет авторизацию пользователя в аккаунт и возвращает информацию о нем |

Таблица 8

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Описание полей, методов и свойств класса RegisterController | | | | | |
| Поля | | | | | |
| Наименование | Модификатор доступа | Тип аргумента | | Назначение | |
| \_context | private | DataContext | | Контекст базы данных | |
| \_usersController | private | IUsersController | | Экземпляр контроллера, который отвечает за взаимодействие с определенным пользователем | |
| Методы | | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | | Назначение |
| CheckEmail | public | void | string email | | Проверяет, не существует ли в БД пользователя с таким email |
| Register | public | void | UserRegisterRequest request | | Осуществляет регистрацию пользователя и возвращает информацию о созданном аккаунте |

Таблица 9

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Описание полей, методов и свойств класса UsersController | | | | | | | |
| Поля | | | | | | | |
| Наименование | Модификатор доступа | | | Тип аргумента | | Назначение | |
| \_context | private | | | DataContext | | Контекст базы данных | |
| \_emailSender | private | | | IEmailSender | | Экземпляр класса, который отвечает за реализацию логики отправки сообщений на электронную почту пользователя | |
| \_userService | private | | | IUserService | | Экземпляр класса, который отвечает за дополнительные операции с пользовательскими данными, такие как хеширование данных и отправку кода верификации на электронную почту | |
| Методы | | | | | | | |
| Имя | | Модификатор доступа | Тип | | Аргументы | | Назначение |
| GetUsers | | public | IEnumerable<User> | | - | | Возвращает всех пользователей |
| DeleteUser | | public | void | | int userId | | Осуществляет удаление пользователя |
| SendVerificationCode | | public | int | | string receiverEmail | | Осуществляет сбор данных и отправку кода подтверждения на адрес пользователя и возвращает сгенерированный код |
| GetUserByEmail | | public | User | | string email | | Возвращает пользователя по его email |
| GetUserById | | public | User | | int userId | | Возвращает пользователя по его идентификатору |
| GetLikedMediaByUser | | public | IEnumerable<Media> | | int userId | | Возвращает медиа-объекты, который пользователь добавлял в Избранные |

Таблица 10

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Описание полей, методов и свойств класса TagsRepository | | | | | | | |
| Поля | | | | | | | |
| Наименование | Модификатор доступа | | | Тип аргумента | | Назначение | |
| \_context | private | | | DataContext | | Контекст базы данных | |
| Методы | | | | | | | |
| Имя | | Модификатор доступа | Тип | | Аргументы | | Назначение |
| GetTagsAsync | | public | IEnumerable<Tag> | | - | | Возвращает все теги |
| GetTagsByMediaAsync | | public | IEnumerable<Tag> | | Media media | | Возвращает все теги, принадлежащие определенному медиа-объекту |
| AddTagAsync | | public | Tag | | Media media, string text | | Осуществляет добавление одного или нескольких тегов к медиа-объекту |
| DeleteTagAsync | | public | void | | Media media, Tag tag | | Осуществляет удаление тега для медиа-объекта |

Таблица 11

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Описание полей, методов и свойств класса DataContext | | | | | | |
| Поля | | | | | | |
| Наименование | Модификатор доступа | | Тип аргумента | | Назначение | |
| Users | public | | DbSet<User> | | Коллекция объектов сущности User, которая соответствует таблице Users в БД | |
| Groups | public | | DbSet<Group> | | Коллекция объектов сущности Group, которая соответствует таблице Groups в БД | |
| Media | public | | DbSet<Media> | | Коллекция объектов сущности Media, которая соответствует таблице Media в БД | |
| Tags | public | | DbSet<Tags> | | Коллекция объектов сущности Tag, которая соответствует таблице Tags в БД | |
| Методы | | | | | | |
| Имя | Модификатор доступа | Тип | | Аргументы | | Назначение |
| GetDbConnection | public | string | | - | | Возвращает информацию о строке подключения к серверу с базой данных |

Таблица 12

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса Group | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Id | public | int | Идентификатор группы |
| OwnerId | public | int | Идентификатор владельца группы |
| Name | public | string | Название группы |
| Image | public | byte[] | Аватарка группы |
| TinyImage | public | byte[] | Миниатюра аватарки группы |
| Users | public | ICollection<User> | Коллекция пользователей, которые содержатся в группе |
| Media | public | ICollection<Media> | Коллекция медиа-объектов, которые содержатся в галерее группы |

Таблица 13

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса Media | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Id | public | int | Идентификатор медиа-объекта |
| Date | public | DateTime | Дата загрузки медиа-объекта |
| Object | public | byte[] | Медиа-объект |
| TinyObject | public | byte[] | Миниатюра медиа-объекта |
| IsPhoto | public | bool | Флаг, показывающий является медиа-объект фотографией или цитатой |
| Tags | public | ICollection<Tag> | Коллекция тегов, принадлежащих медиа-объекту |
| Users | public | ICollection<User> | Коллекция пользователей, которые добавили медиа-объект в Избранные |
| Author | public | UserView | Автор загрузки медиа-объекта |
| Group | public | Group | Группа, в которую был загружен медиа-объект |

Таблица 14

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса Tag | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Id | public | int | Идентификатор тега |
| Text | public | string | Текст тега |
| Media | public | Media | Медиа-объект, которому принадлежит тег |

Таблица 15

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса User | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Id | public | int | Идентификатор пользователя |
| UserName | public | string | Имя пользователя |
| Email | public | string | Электронная почта пользователя |
| Password | public | string | Пароль пользователя |
| RegistrationDateTime | public | DateTime | Дата и время регистрации пользователя |
| ProfileImage | public | byte[] | Аватарка пользователя |
| TinyProfileImage | public | byte[] | Миниатюра аватарки пользователя |
| AppIcon | public | string | Название выбранной пользователем иконки приложения |
| Groups | public | ICollection<Group> | Группы, в которых числится пользователь |
| Media | public | ICollection<Media> | Медиа-объекты, которые пользователь добавил в избранные |

Таблица 16

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Описание методов класса Extensions | | | | |
| Методы | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| Shuffle | Public | IEnumerable<T> | this IEnumerable<T> source | Пользовательский метод, нужный для перемешивания коллекции |

Таблица 17

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса EmailRequest | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Email | public | string | Электронная почта пользователя |

Таблица 18

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса MediaRequest | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Object | public | string | Медиа-объект в формате base64 |
| TinyObject | public | string | Миниатюра медиа-объекта в формате base64 |

Таблица 19

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса PasswordRequest | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Password | public | string | Пароль |

Таблица 20

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса LoadMediaRequest | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Object | public | string | Медиа-объект в формате base64 |
| TinyObject | public | string | Миниатюра медиа-объекта в формате base64 |
| IsPhoto | public | int | Флаг, показывающий является медиа-объект фотографией или цитатой |
| Tags | public | ICollection<string> | Теги медиа-объекта |

Таблица 21

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса TagsRequest | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Tags | public | ICollection<string> | Теги медиа-объекта |

Таблица 22

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса UserLoginRequest | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Email | public | string | Электронная почта пользователя |
| Password | public | string | Пароль |

Таблица 23

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса UserRegisterRequest | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| UserName | public | string | Имя пользователя |
| Email | public | string | Электронная почта пользователя |
| Password | public | string | Пароль |

Таблица 24

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса CreateGroupRequest | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| OwnerId | public | int | Идентификатор владельца группы |
| GroupName | public | string | Название группы |

Таблица 25

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса FilterSettings | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Images | public | bool | Фильтр, обозначающий наличие фотографий в будущей выборке |
| Quotes | public | bool | Фильтр, обозначающий наличие цитат в будущей выборке |
| OnlyFavorites | public | bool | Фильтр, обозначающий наличие только понравившихся медиа-объектов в будущей выборке |

Таблица 26

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса TextRequest | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Text | public | string | Текст тега |

Таблица 27

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса RewindScreenDataResponse | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Groups | public | IEnumerable<GroupView> | Группы, в которых содержится пользователь |
| RandomImage | public | BigMediaInfoResponse | Рандомный медиа-объект, который содержится в текущей группе и будет показан на главном экране |
| GallerySize | public | int | Размер галереи группы |

Таблица 28

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Описание полей, методов и свойств класса EmailSender | | | | | | | |
| Поля | | | | | | | |
| Наименование | Модификатор доступа | | | Тип аргумента | | Назначение | |
| Mail | private | | | string | | Почта-отправитель кодов верификации | |
| Pass | private | | | string | | Пароль для сторонних приложений для почты-отправителя | |
| Методы | | | | | | | |
| Имя | | Модификатор доступа | Тип | | Аргументы | | Назначение |
| SendEmail | | public | void | | string receiver,  string subject,  string body | | Осуществляет отправку сообщения на указанный адрес электронной почты |

Таблица 29

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Описание методов класса SqlService | | | | |
| Методы | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| RunCommand | public | int | string commandText | Осуществляет выполнение SQL-команды |
| UpdateUserImage | public | void | List<MySqlParameter> parameters | Осуществляет сбор данных для выполнения команды на обновление аватарки пользователя |
| UpdateGroupImage | public | void | Group group | Осуществляет сбор данных для выполнения команды на обновление аватарки группы |
| LoadMedia | public | int | byte[] rawData,  byte[] tinyData,  int groupId,  int authorId,  int isPhoto | Осуществляет сбор данных для выполнения команды на загрузку медиа-объекта в группу |

Таблица 30

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Описание полей, методов и свойств класса UserService | | | | | |
| Поля | | | | | |
| Наименование | Модификатор доступа | Тип аргумента | | Назначение | |
| VerificationCodeMinValue | private | int | | Минимальное значение кода верификации | |
| VerificationCodeMaxValue | private | int | | Максимальное значение кода верификации | |
| Методы | | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | | Назначение |
| ComputeHash | public | string | string input | | Создает хэш по определённой строке |
| GenerateCode | public | int | - | | Генерирует случайный код верификации в диапазоне [1000; 10000) |

Таблица 31

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса GroupInfoResponse | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Id | public | int | Идентификатор группы |
| DataSize | public | int | Количество нужных данных |
| Name | public | string | Название группы |
| Image | public | byte[] | Аватарка группы |
| Owner | public | UserView | Владелец группы |
| FirstMembers | public | IEnumerable<UserView> | {dataSize} случайных участников группы |
| GallerySize | public | int | Размер галереи группы |
| FirstMedia | public | IEnumerable<MediaView> | {dataSize} случайных медиа-объектов группы |

Таблица 32

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса GroupView | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Id | public | int | Идентификатор группы |
| Name | public | string | Название группы |
| OwnerId | public | id | Идентификатор владельца группы |
| TinyImage | public | byte[] | Миниатюра аватарки группы |
| GallerySize | public | int | Размер галереи группы |

Таблица 33

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса SmallGroupInfoResponse | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Id | public | int | Идентификатор группы |
| Name | public | string | Название группы |
| OwnerId | public | id | Идентификатор владельца группы |
| TinyImage | public | byte[] | Миниатюра аватарки группы |

Таблица 34

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса BigMediaInfoResponse | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Id | public | int | Идентификатор медиа-объекта |
| Object | public | byte[] | Медиа-объект |
| Author | public | Author | Автор загрузки медиа-объекта |
| Date | public | DateTime | Дата загрузки медиа-объекта |
| Liked | public | bool | Наличие лайка на медиа-объекте у определённого пользователя |

Таблица 35

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса LargeMediaInfoResponse | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Id | public | int | Идентификатор медиа-объекта |
| Object | public | byte[] | Медиа-объект |
| Author | public | UserView | Автор загрузки медиа-объекта |
| Date | public | DateTime | Дата загрузки медиа-объекта |
| Liked | public | bool | Наличие лайка на медиа-объекте у определённого пользователя |
| Tags | public | int | Теги медиа-объекта |

Таблица 36

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса MediaView | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Id | public | int | Идентификатор медиа-объекта |
| TinyObject | public | byte[] | Миниатюра медиа-объекта |

Таблица 37

|  |  |  |  |
| --- | --- | --- | --- |
| Описание полей класса UserView | | | |
| Поля | | | |
| Наименование | Модификатор доступа | Тип аргумента | Назначение |
| Id | public | int | Идентификатор пользователя |
| UserName | public | string | Имя пользователя |
| TinyProfileImage | public | byte[] | Миниатюра аватарки |

# ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц) в докум. | № документа | Входящий  № сопроводительного докум. и дата | Подп. | Дата |
| Изм. | Измененных | Замененных | Новых | Аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |