МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«ТЮМЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ НАУК
Кафедра информационных систем

РЕКОМЕНДОВАНО К ЗАЩИТЕ
В ГЭК И ПРОВЕРЕНО НА ОБЪЕМ
ЗАИМСТВОВАНИЯ
Заведующий кафедрой
д.т.н., профессор
_____И.Н. Глухих
_____2017 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

ПРИМЕНЕНИЕ РЕКОМЕНДАТЕЛЬНЫХ СИСТЕМ В МОБИЛЬНЫХ ПРИЛОЖЕНИЯХ

09.04.03 Прикладная информатика
Магистерская программа «Прикладная информатика в экономике»

Выполнил работу
Студент 2 курса
очной формы обучения
_____
Карпенко
Александр
Сергеевич

Научный руководитель
д.т.н., профессор
_____
Глухих
Игорь
Николаевич

Рецензент
Технический директор
ООО «КБ-Информ», к.т.н.
_____
Бабич
Андрей
Владимирович

Тюмень 2017

Диссертация состоит из 74 страниц содержит 4 главы, включает в себя 16 рисунков, 2 таблицы, 8 приложений, 18 источников.

Целью работы является разработка масштабируемого решения для улучшения отклика приложения, с использованием рекомендательных систем.

Для достижения поставленной цели был определен ряд задач:

- Выбор архитектуры приложения
- Проектирование системы
- Разработка бэкенд-части
- Разработка тестового мобильного приложения
- Разработка пакета классов, позволяющих использовать рекомендательную систему в целях предиктивной подгрузки данных

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В ходе преддипломной практики было проведено исследование, сбор данных и реализация подхода улучшения отклика мобильных приложений с помощью рекомендательных систем.

Данная работа является продолжением предыдущих работ, в которых были изучены рекомендательные системы, реализованы основные подходы рекомендательных систем, выявлены их проблемы и изучены решения проблем, были опробованы подходы в реализации мобильных приложений, реализовано экспериментальное мобильные приложение, в которое встроена рекомендательная система, с целью улучшения отклика, проведены эксперименты над этим приложениям, давшие положительный результат в виде значительного уменьшения времени открытия контента.

Результатом выполнения первой научно-исследовательской работы стали данные о подходах к реализации рекомендательных систем и реализация экспериментальной системы двумя способами, которая выдавала рекомендации. В ходе этой работы было выяснено, что существуют 3 основных способа построения рекомендательных систем, а именно: collaborative filtering, content-based filtering и гибридные системы. Были определены проблемы этих подходов, а именно: холодный старт, сложная реализация, трудномоделируемое поведение пользователей, фрод и масштабируемость. Найдены способы решения проблем, однако, для каждой ситуации разрабатывается свое, индивидуальное решение. Построена рекомендательная система с применением подходов collaborative filtering и content-based filtering, полученные результаты оценены по методу precision-recall. На основании оценки были сделаны выводы, что применение ollaborative filtering дает более качественные рекомендации, чем content based, в некоторых случаях в 10 раз. Однако, при этом производительность коллаборативной системы значительно ниже, требует больше ресурсов, более сложна в реализации [1].

В ходе практической работы, была внедрена рекомендательная система на сайт, содержащий товары и новости. Была реализована рекомендательная система использующая как collaborative filtering так и content-based filtering. Система работала в типичном для других подобных систем виде - давала рекомендации по товарам и предлагала новости пользователям сайта [2].

Следующая практика заключалась в том, что было разработано мобильное приложение ЗАГС 72 по заказу правительства Тюменской области. В ходе разработки данного приложения были опробованы способы построения мобильных приложений для операционной системы Android, выбрана оптимальная архитектура, опробованы передовые методы и инструменты. Приложение опубликовано и доступно для пользования [3].

Далее было проведено исследование на предмет применения рекомендательных систем в мобильных приложениях в целях улучшения отклика. В данной исследовательской работе выведена гипотеза о том, что рекомендательная система может давать рекомендации в скрытом для пользователя режиме, кэшировать контент, который может заинтересовать пользователя и тем самым улучшать отклик приложения в целом. Идея в том, что кэшированные данные открываются значительно быстрее тех данных, которые подгружаются из сети. Система определяет, что «угадала» предпочтение пользователя тогда, когда пользователь открывает кэшированный контент. В любом случае, чем чаще пользователь открывает различный контент, тем больше обучает рекомендательную систему. В результате было реализовано экспериментальное мобильное приложение, отображающее новости, и рекомендательная система. Приложение в фоновом процессе кэшировало рекомендуемые новости, а при открытии новости логировало время открытия контента. Были проведены эксперементы на двух тестовых мобильных утсройствах. По результатам эксперимента, данный метод показал, что отклик приложения значительно улучшился. Это послужило основанием дальнейших исследований и реализации библиотеки,

которую можно внедрять в другие мобильные приложения и улучшать их отклик [4].

В данной работе приведен отчет о реализации библиотеки, которая использует рекомендательный сервис для улучшения отклика. В ходе работы, было разработано тестовое мобильное приложение и набор классов, в последствии ставший библиотекой для встраивания в мобильные приложения на платформе Android.

Актуальность работы заключается в том, что в данный момент рынок мобильных устройств является быстрорастущим, а основной прирост устройств подкреплен в первую очередь дешевыми, низкопроизводительными устройствами, соответственно, для огромного количества устройств нет возможности оптимизировать каждое приложение, поэтому, помимо стандартных методов улучшения отклика приложений, остро необходимы масштабируемые решения, которые можно внедрить в большинство мобильных приложений.

На основании вышесказанного, была выведена цель данной работы, а именно: разработка масштабируемого решения для улучшения отклика приложения, с использованием рекомендательных систем.

Для достижения поставленной цели был определен ряд задач:

- Выбор архитектуры приложения
- Проектирование системы
- Разработка бэкенд-части
- Разработка тестового мобильного приложения
- Разработка пакета классов, позволяющих использовать рекомендательную систему в целях предиктивной подгрузки данных

Данная работа состоит из 4 глав.

В первой главе под названием «Предметная область» проводится обзор актуальных операционных систем для мобильных устройств, анализ распределения операционных систем, обзор подходов к реализации рекомендательных систем, приводится концепция применения

рекомендательных систем в мобильных приложениях с целью улучшения отклика, концептуальная схема взаимодействия элементов системы и диаграмма последовательностей, приводится обзор альтернатив.

Во второй главе описывается эксперимент, в котором разработано новостное мобильное приложение с применением рекомендательных систем в целях улучшения отклика и рассматриваются результаты применения рекомендательных систем.

В третей главе описывается постановка задачи.

В четвёртой главе описывается проектирование системы, выбор архитектуры приложения, разработка диаграммы последовательностей, классов, и схемы взаимодействия элементов системы. Кроме этого, описывается реализация разработки, и внедрение в другие приложения.

Результаты работы подводятся в заключении.

1. Предметная область

1.1 Обзор мобильных операционных систем

Развитие рынка мобильных устройств обуславливает повышение потребления контента из сети. По данным Cisco Visual Networking Index (VNI) в 2016 году мобильный трафик показал стабильный рост в 63%, а по прогнозу к 2021 году мобильный трафик достигнет 20% всего IP-трафика, на душу населения будет приходиться 1.5 мобильных устройства, около 12 миллиардов устройств, подключенных к мобильным сетям, а общее число смартфонов и подобных устройств, достигнет 6.2 миллиарда и превысит половину всех устройств и подключений [5]. В то же время такой рост мобильных устройств обеспечен выпуском дешевых девайсов с низкой производительностью, что делает их производство массовым и доступным для потребителей в развивающихся странах.

Если рассматривать рынок смартфонов, то это динамично развивающееся направление и с каждым днем смартфоны становятся доступнее и производительнее. Смартфон предполагает наличие операционной системы. Помимо стандартного функционала связи, предполагается расширение с помощью установки дополнительных приложений.

По данным сайта gs.statcounter.com в России в период с марта 2015 года по март 2017 года самыми популярными мобильными операционными системами являются Android(60.15), iOS(33.64), Windows(2.69), диаграмма распределения популярности мобильных операционных систем (среди активированных устройств, которые в этот промежуток времени хотя бы один раз подключались к сети интернет) представлена на рисунке 1.

Рисунок 1 – распределение мобильных операционных систем в России в период март 2015 – март 2017

Из диаграммы видно, что лидирующую позицию занимают устройства с операционной системой Android. Это обусловлено тем, что девайсы представлены практически во всех ценовых категориях, что делает их в первую очередь доступными для потребителей, кроме того операционная система Android является современной и обладает широким функционалом, множеством программ в официальном маркете приложений, активно

развивается и становится стабильнее. Фактически все приложения представленные в официальном маркете приложений так или иначе взаимодействуют с сетью интернет.

Рассмотрим две самые популярные операционные системы: android и ios.

Android – это операционная система рассчитанная на работу на портативных устройствах таких как смартфоны, планшеты, электронные книги, цифровые проигрыватели, умные часы, игровые приставки, нетбуки, очки Google, телевизоры, автомобильные системы [16]. Операционная система основана на ядре Linux и собственной реализации виртуальной машины Java от Google. Android поддерживается альянсом Open Handset Alliance, который занимается поддержкой и развитием платформы. Разработка приложений для операционной системы Android ведется на языке Java (7-8) версии, либо Kotlin. Управление устройством происходит через разработанные Google библиотеки. Кроме того, Android Native Development Kit позволяет портировать библиотеки и компоненты приложений написанные на C и других языках. В данный момент последней версией является Android 7.1.2 от 4 октября 2016 года. Для разработчиков приложений предлагается использование сервисов от Google в которые входят сервисы навигации, Firebase, сервисы работы с камерой и дополненной реальностью, магазин приложений для Android – Google Play Market и другие сервисы. Основным каналом распространения приложений для Android является Google Play.

iOS – операционная система рассчитанная на работу на портативных устройствах, таких как Apple iPhone, Apple iPod, Apple iPad, Apple TV [17]. В отличие от Android, распространяется только на устройства фирмы Apple, которая является разработчиком данной операционной системы и устройств для нее. Данная операционная система работает на ядре XNU, основанное на микроядре Mach. Разработка приложений ведется на языках Objective-C и Swift. В данный момент самой последней версией является 10.3.3 от 16 мая

2017 года. Основным каналом распространения приложений является App Store от Apple.

1.2 Обзор подходов построения рекомендательных систем

Рекомендательные системы – субкласс информационных систем фильтрации, пытающихся предсказать «рейтинг» или «предпочтение», которые даст пользователь такой системы какому-либо элементу [6].

Как правило, рекомендательные системы выдают список рекомендаций, используя два основных метода – collaborative filtering или content-based filtering.

Collaborative filtering полагается на построение рекомендаций на основе оценок (рейтинга, просмотра, проявления интереса) пользователя и поиска схожих оценок других пользователей. Иначе говоря, такой подход основывается на теории коллективного разума – то что нравится одним, скорее всего должно понравится и другим. Такой подход способен спрогнозировать элементы, потенциально интересные пользователю [7]. Метод collaborative filtering базируется на сборе и анализе огромного количества информации о поведении пользователей, их оценках, предпочтениях, и предсказании пересекающихся точек интереса этих пользователей. Основным преимуществом данного подхода является то, что он не полагается на машинный анализ содержимого контента и следовательно с помощью такого подхода можно рекомендовать контент со сложным содержанием, либо структурой, например кинофильмы.

При построении модели поведения пользователя существует два способа сбора данных: явный и неявный.

Примеры явного сбора данных:

- Попросить пользователя оценить контент
- Предложить воспользоваться поиском
- Предложить пользователю оценить элементы от более привлекательных к менее

- Показать пользователю два элемента и предложить выбрать тот, который нравится больше
- Предложить пользователю составить список элементов, которые ему нравятся

Примеры неявного сбора данных:

- Следить за товарами, которые просматривает пользователь в онлайн магазине
- Анализ количества просмотров элемента пользователем
- Получение списка элементов, которые пользователь открыл\просмотрел на своем компьютере
- Анализ профиля пользователя из социальных сетей

Подход collaborative filtering часто сталкивается с тремя основными проблемами: холодный старт, масштабируемость и разрежённость [8].

- Холодный старт, проблема заключающаяся в том, что для полезных рекомендаций требуется большой массив данных о предпочтениях пользователя, который не доступен на начальном этапе.
- В среде, где используются рекомендательные системы, как правило миллионы пользователей и продуктов. Таким образом требуется огромное количество вычислительных мощностей.
- Проблема разрежённости сводиться к тому, что только активные пользователи ставят оценки, таким образом даже самые продаваемые товары могут иметь низкое количество оценок, либо наоборот может иметь место быть «накрутка» не популярных товаров для попадания их в рекомендации.

Content-based filtering подход использующий серии дискретных характеристик элемента, для рекомендации похожих элементов пользователю [9]. Способ полагается на то, что пользователю будут интересны те элементы, которые похожи своими характеристиками на другие. Проще говоря, данный подход основывается на предположении, что пользователю будут полезны рекомендации похожие на те элементы, которые он предпочел ранее.

Система создает content-based профиль пользователя основанный на векторе весов характеристик элементов. Веса означают значимость каждой характеристики для пользователя и могут быть вычислены на основе индивидуальных предпочтений с помощью различных техник.

В простой реализации данного подхода, как правило, используется метод средних векторов оценок, в более сложных используются методы машинного обучения, байесовский классификатор, кластерный анализ, дерево принятия решений, искусственная нейронная сеть, для того, чтобы оценить какие элементы могут понравиться пользователю [10].

Гибридные рекомендательные системы – системы использующие комбинацию нескольких подходов выдачи рекомендаций [11]. Такие системы имеют крайне высокое качество выдачи прогнозов, однако значительно сложнее в реализации и требуют намного более значительных вычислительных ресурсов. Последние исследования показывают, что комбинирование collaborative filtering и content-based filtering могут быть более эффективны в некоторых случаях. Гибридный подход может быть реализован несколькими способами:

1) подготовить прогнозы collaborative filtering и content-based filtering по отдельности, затем скомбинировать результат,

2) добавление возможностей content-based filtering в collaborative filtering (и наоборот),

3) унифицировать подходы в одной модели.

Некоторые опыты подтверждают, что качество рекомендаций гибридного подхода значительно выше чистых collaborative filtering и content-based filtering. Такой подход также может использоваться для решений общих проблем рекомендательных систем, таких как холодный старт и разрежённость.

Гибридная система комбинирует подходы для получения синергии между ними. Можно выделить 7 основных подходов к гибридным системам:

- Весовой. Оценки различный рекомендательных компонентов объединены числено.
- Переключение: система выбирает между рекомендациями только одну.
- Смешанный подход – предоставляет рекомендации разных систем.
- Комбинация характеристик. Характеристики из разных источников комбинируются и выдается одна рекомендация
- Усиление характеристик. Одна техника рекомендаций используется для вычисления характеристик, являющихся входной частью данных для следующей техники.
- Каскад. Рекомендациям дается строгий приоритет, выдаются рекомендации на основе приоритета.

Мета-уровень. Метод в котором одна система производит одну модель, которая служит входной моделью для другой системы.

Каждый подход имеет свои проблемы, особенно распространена проблема «холодного старта», в той или иной мере справедливая для рекомендательных систем [12].

Рекомендательные системы являются альтернативой поисковым алгоритмам, т.к. помогают пользователям найти те элементы, которые сложно найти с помощью поиска.

Рекомендательные системы имеют широкий диапазон применения. В данный момент широкое распространение получили сайты, специализирующиеся на продаже товаров в интернете, музыкальные сервисы, видео-сервисы, поисковые системы, сервисы-агрегаторы новостей и развлекательного контента и так далее. Пример типичного применения content-based filtering с сайта amazon.com представлен на рисунке 2.

Рисунок 2 – пример content-based filtering с сайта amazon.com

Пример типичного применения collabarative filtering с сайта amazon.com представлен на рисунке 3.

Рисунок 3 – пример collabarative filtering с сайта amazon.com

1.3 Концепция применения рекомендательных систем в целях улучшения отклика приложений

Современные мобильные устройства сильно различаются по производительности особенно это заметно среди устройств работающих на операционных системах Android и Windows. Такие устройства представлены во всех ценовых сегментах. Рассмотрим устройства под управлением операционной системы Android, так как такие устройства являются самыми распространёнными не только в России, но и по всему миру. У самых доступных устройств, как правило низкопроизводительные комплектующие, кроме того, обилие моделей и разброс в версиях операционной системы вызывает явление дефрагментации, что также плохо сказывается на производительности. Если на флагманских аппаратах, все работает достаточно быстро, то на доступных устройствах как правило проявляются задержки в работе и частые «зависания». Для избежания этого, разработчику требуется оптимизировать приложение под конкретное устройство, что

является довольно затруднительным из-за обилия устройств и их модификаций.

Рассмотрим возможность применения рекомендательной системы в целях повышения производительности работы мобильного приложения. Идея заключается в том, чтобы обучить рекомендательную систему предугадывать дальнейшие действия пользователя во время простоя и на основании рекомендаций подгружать данные в фоне. Если система сработала верно, то пользователь откроет уже подгруженные данные, а в свою очередь рекомендательная система обучится. Если система не «угадала» действие пользователя, то она также обучится. Такой подход может значительно увеличить скорость отклика приложения, при условии, что рекомендательная система обучена.

1.4 Обзор альтернативных подходов повышения производительности приложений

Существует несколько основных подходов, для повышения отклика приложений на операционной системе Android.

Одним из способов является разработка с помощью набора инструментов NDK. Это означает, что разработка ведется на языке программирования C и C++. Такой способ сильно усложняет разработку, часто требует оптимизации под конкретные устройства. Часто бывает неоправданным.

Кроме этого разработчики прибегают к использованию многопоточности. Такой подход позволяет задействовать все ядра процессора устройства, уменьшить простои, распараллелить задачи.

Оптимизация ресурсов также является методом улучшения отклика приложения. Например, вместо обычной верстки можно использовать программную.

Самой последней и перспективной разработкой в плане повышения отклика работы не только приложений, но и устройства в целом является

применение обучаемого механизма предсказания ветвлений [18]. Предсказание ветвлений позволяет сократить время простоя конвейера, за счёт предварительной загрузки и исполнения инструкций, которые должны выполниться после выполнения инструкции условного перехода. Прогнозирование ветвлений критически важно, потому что позволяет оптимально использовать вычислительные ресурсы процессора. Однако, такой подход используется только в последних флагманских устройствах фирмы Samsung, массового распространения такая технология пока что не получила.

Применение даже всех этих методов не гарантирует значительного повышения производительности.

2. Постановка эксперимента

В рамках научно-исследовательской работы мною был поставлен эксперимент, в ходе которого выяснилось, что применение рекомендательных систем в мобильных приложениях с целью повышения отклика работает. Рассмотрим поставленный эксперимент более подробно.

Для реализации был использован сервис рекомендаций Recombee, который составляет список рекомендаций на основании информации, поступающей из клиентского приложения. Было разработано простое мобильное приложение для операционной системы Android, которое получало список новостей с сайта Hackernews.com, выводило этот список, параллельно создавая свой список рекомендаций из 5 новостей на основании данных от сервиса Recombee. Список рекомендованных новостей кэшировался. Если пользователь открывал рекомендованную для него новость, то такая новость загружалась из кэша, если пользователь открывал новость не из списка рекомендаций, то новость просто загружалась из сети. Все просмотры новостей фиксировались и отправлялись в сервис Recombee. Таким образом, система обучалась и улучшала рекомендации. Помимо этого, время затраченное на открытие новости логировалось , пример представлен на рисунке 4.

Рисунок 4 – логирование времени открытия

На рисунке 6 представлен пример открытия двух новостей не из списка рекомендаций, и последней из списка. Как видно, последняя новость открылась всего лишь за 1 секунду.

Эксперименты проводились на двух смартфонах под управлением Android. Первый смартфон это Samsung Galaxy S6 G920F, 3Gb оперативной памяти 64-bit dual-channel 1552 MHz LPDDR4 (24.88 GB/s), процессор Cortex-

A57 четырехъядерный 2.1 ГГц+ Cortex-A53 четырехъядерный 1.5 ГГц (ARM big.LITTLE with GTS), Android 7. Вторым аппаратом для испытаний послужил Samsung Galaxy S2 I9100, 1 GB оперативной памяти, двухъядерный процессор ARM Cortex-A9 Samsung на базе чипа Exynos 4210 с тактовой частотой 1.2 ГГц, построенный на однокристальной системе, Android 4.2.2.. Было произведено 10 случайных открытий новостей на этих устройствах. Результат представлен в таблице 1.

Таблица 1.

Samsung Galaxy S2				
№	**Начало загрузки**	**Конец загрузки**	**Разница**	**Рекомендация**
1	11:13:26.070	11:13:28.280	2,21	нет
2	11:13:40.075	11:13:44.790	4,72	нет
3	11:13:56.275	11:13:56.805	0,53	да
4	11:14:04.215	11:14:05.510	1,30	да
5	11:43:14.215	11:43:19.970	5,76	нет
6	11:49:27.740	11:49:29.650	1,91	нет
7	11:50:05.625	11:50:07.285	1,66	да
8	11:51:16.975	11:51:19.415	2,44	нет
9	11:51:39.960	11:51:42.375	2,42	нет
10	11:52:16.025	11:52:18.060	2,04	нет
Samsung Galaxy S6				
1	13:10:27.050	13:10:27.970	0,92	да
2	13:10:30.075	13:10:32.090	2,02	нет
3	13:10:21.045	13:10:22.005	0,96	да
4	13:11:03.210	13:11:05.320	2,11	нет
5	13:12:13.005	13:12:15.610	3,61	нет
6	13:12:26.540	13:12:28.070	1,53	нет

7	13:13:04.635	13:13:07.285	2,65	нет
8	13:13:14.075	13:13:15.255	1,18	да
9	13:13:36.060	13:13:38.050	1,99	нет
10	13:14:15.020	13:14:15.990	0,97	да

Как видно из таблицы время загрузки предварительно рекомендованных элементов по сравнению с не рекомендованными сократилось. Особенно это заметно на слабом устройстве, что подтверждает начальное утверждение, что применение рекомендательных систем в целях повышения производительности приложения работает. Поэтому, было принято решение ставить эксперименты и дальше. Для дальнейших опытов было выбрано менее производительное устройства – Samsung Galaxy S2, так как на нем более ярко выражен результат применения рекомендательной системы в целях улучшения отклика. На устройстве Samsung Galaxy S2 было запущенно 100 случайных открываний различный новостей в случайное время. Результат эксперимента представлен в приложении 2. В следствии 100 прогонов, было открыто 30 рекомендованных новостей из кэша и 70 загружено из сети. В результате было выяснено, что среднее время открытия новости из кэша составляет 1 секунду, 161 миллисекунд, а загрузка из сети в среднем составляет 5 секунд и 554 миллисекунда, что значительно дольше. По результатам поставленных опытов можно сделать вывод о том, что применение рекомендательных систем в мобильных приложениях в целях улучшения отклика работает и дает значительный прирост открытия контента.

3. Постановка задачи

В ходе исследования возможности повышения отклика мобильных приложений было выявлено, что рекомендательные системы применимы в таких целях. Однако, в данный момент не существует такого сервиса или подхода, который предоставлял бы возможность использования рекомендательных систем в целях улучшения отклика мобильных приложений, это определяет актуальность данной разработки. Кроме того, имеющиеся способы являются обобщёнными и не являются кроссплатформенными. На основании этого было принято решение разработки сервиса, который можно легко интегрировать в приложение, разрабатываемое под любую платформу.

На основании вышесказанного, была выведена цель данной работы, а именно: разработка масштабируемого решения для улучшения отклика приложения, с использованием рекомендательных систем.

Для достижения поставленной цели был определен ряд задач:

- Выбор архитектуры приложения
- Проектирование системы
- Разработка бэкенд-части
- Разработка тестового мобильного приложения
- Разработка пакета классов, позволяющих использовать рекомендательную систему в целях предиктивной подгрузки данных

4. Проектирование системы

4.1 Выбор архитектуры

Так как в данной работе будет рассматриваться разработка приложения для операционной системы Android, то и архитектурные решения будут рассмотрены, соответственно, типичные для разработки под такую операционную систему.

Современными архитектурными подходами в разработке приложений для Android являются: MVC, MVP, MVVM.

Model-View-Controller – шаблон проектирования, разделяющий данные приложения, пользовательский интерфейс, и управляющую логику на три отдельных компонента: модель-представление-контроллер [13]. Идея заключается в том, что каждый из компонентов может модифицироваться практически независимо от других.

Плюсы:

- Возможность одновременной(параллельной) разработки
- Группировка по функционалу
- Легкость модификации
- Множество представлений одной модели

Минусы:

- Сложная навигация по проекту из-за уровней абстракции и декомпозиции MVC
- Декомпозиция должна быть согласована со всей командой разработки

Model-View-Presenter – шаблон проектирования, произвольный от MVC. Данный подход был разработан для облегчения автоматического тестирования и улучшения разделения ответственности в презентационной логике [14]. В данном подходе подразумевается, что Presenter берет на себя функционал посредника и отвечает за управление событиями пользовательского интерфейса. Обычно экземпляр Представления создаёт

экземпляр Presenter’а, передавая ему ссылку на себя. При этом Presenter работает с Представлением в абстрактном виде, через его интерфейс. Когда вызывается событие Представления, оно вызывает конкретный метод Presenter’а, не имеющего ни параметров, ни возвращаемого значения. Presenter получает необходимые для работы метода данные о состоянии пользовательского интерфейса через интерфейс Представления и через него же передаёт в Представление данные из Модели и другие результаты своей работы. Кроме этого, данный подход является рекомендуемым командой разработчиков Google.

Model-View-ViewModel - используется для разделения модели и её представления, что необходимо для изменения их отдельно друг от друга [15]. Например, разработчик задает логику работы с данными, а дизайнер соответственно работает с пользовательским интерфейсом. MVVM удобно использовать вместо классического MVC и ему подобных в тех случаях, когда в платформе, на которой ведётся разработка, присутствует «связывание данных». В шаблонах проектирования MVC/MVP изменения в пользовательском интерфейсе не влияют непосредственно на Модель, а предварительно идут через Контроллер (англ. Controller) или Presenter. В таких технологиях как WPF и Silverlight есть концепция «связывания данных», позволяющая связывать данные с визуальными элементами в обе стороны. Следовательно, при использовании этого приема применение модели MVC становится крайне неудобным из-за того, что привязка данных к представлению напрямую не укладывается в концепцию MVC/MVP. Формально, в Android есть возможность связывания данных.

Рассмотрим применение вышеуказанных архитектур в мобильном приложении.

MVC – является неприспособленным подходом для Android разработки, так как Android SDK имеет в себе Activity и Fragment и для реализации такой архитектуры потребуется множество вспомогательных функций, кроме этого нужно будет решать проблемы жизненного цикла Activity и Fragment.

MVVM – не подходит для разработки приложений для Android, хотя эта платформа и имеет возможность связывания данных, однако жизненные циклы Activity и Fragment несут в этот подход множество дополнительного кода.

MVP – является наиболее подходящим вариантом, если к реализации подойти с использованием контрактов. Контракт – это интерфейс класса, который описывает функционал View и Presenter. Таким образом можно контролировать жизненный цикл, повышается читабельность кода, тестировать такое приложение с помощью автотестов значительно проще.

4.2 Проектирование системы

Для реализации требуется спроектировать работу системы. Внутренними элементами системы является мобильное приложение и бэкенд-приложение. Внешними элементами являются сервис recombee и newsapi.org. Схема взаимодействия элементов системы представлена на рисунке 5.

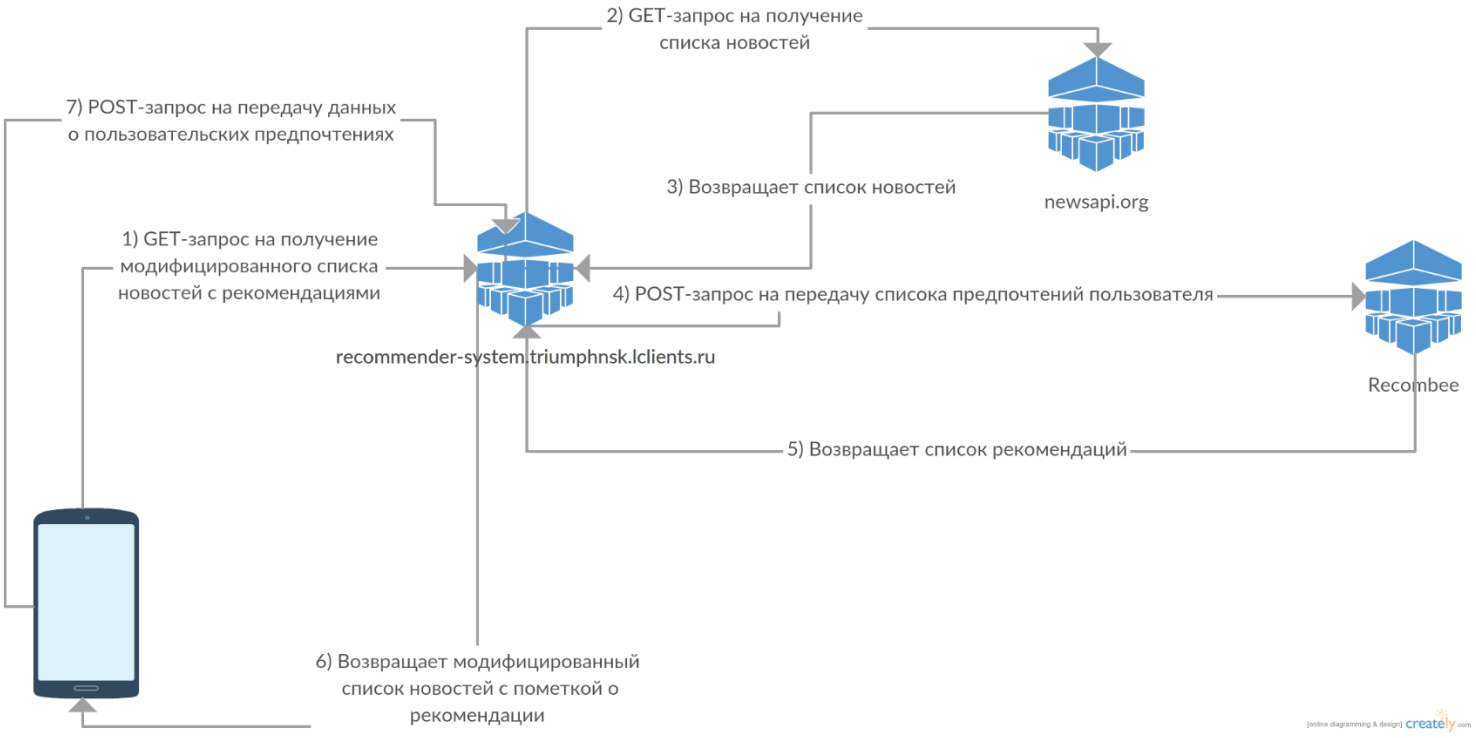


Рисунок 5. Схема взаимодействия элементов системы

На данной схеме изображено взаимодействие элементов системы. Рассмотрим его подробнее. При запуске мобильного приложения, происходит GET-запрос к API recommender-system.triumphnsk.lclients.ru. В свою очередь,

recommender-system.triumphnsk.lclients.ru производит GET-запрос к сервису newsapi.org и в ответ получает JSON-файл, содержащий в себе список новостей (приложение 3). Далее, recommender-system.triumphnsk.lclients.ru передает список новостей, полученный ранее и данные о пользовательских предпочтениях, если таковые имеются, в сервис recombee. Если рекомендаций нет, то к первой пятерке новостей добавляется пометка о том, что они рекомендованы, иначе к списку новостей добавляются пометки о рекомендациях на основании данных сервиса recombee. После этого в мобильное приложение в ответ на первый запрос отправляется модифицированный список новостей, содержащий в себе пометки о том, рекомендована новость или нет (приложение 5). В свою очередь в мобильном приложении выводится список полученных новостей, а после этого новости, у которых стоит пометка о том, что они рекомендованы, начинают кэшироваться. Затем, если пользователь открыл новость и она находится в кэше, то она подгружается из памяти устройства, иначе, новость подгружается из сети. После того, как новость прогрузилась, с мобильного приложения осуществляется POST-запрос к recommender-system.triumphnsk.lclients.ru, в котором содержатся данные о том, какой пользователь, в какой момент времени, какую открыл новость. В свою очередь recommender-system.triumphnsk.lclients.ru сохраняет эти предпочтения и использует для дальнейших рекомендаций.

Кроме этого, были разработаны блок-схемы отображающие работу внутренних элементов системы. На рисунке 6 представлена блок-схема работы мобильного приложения.

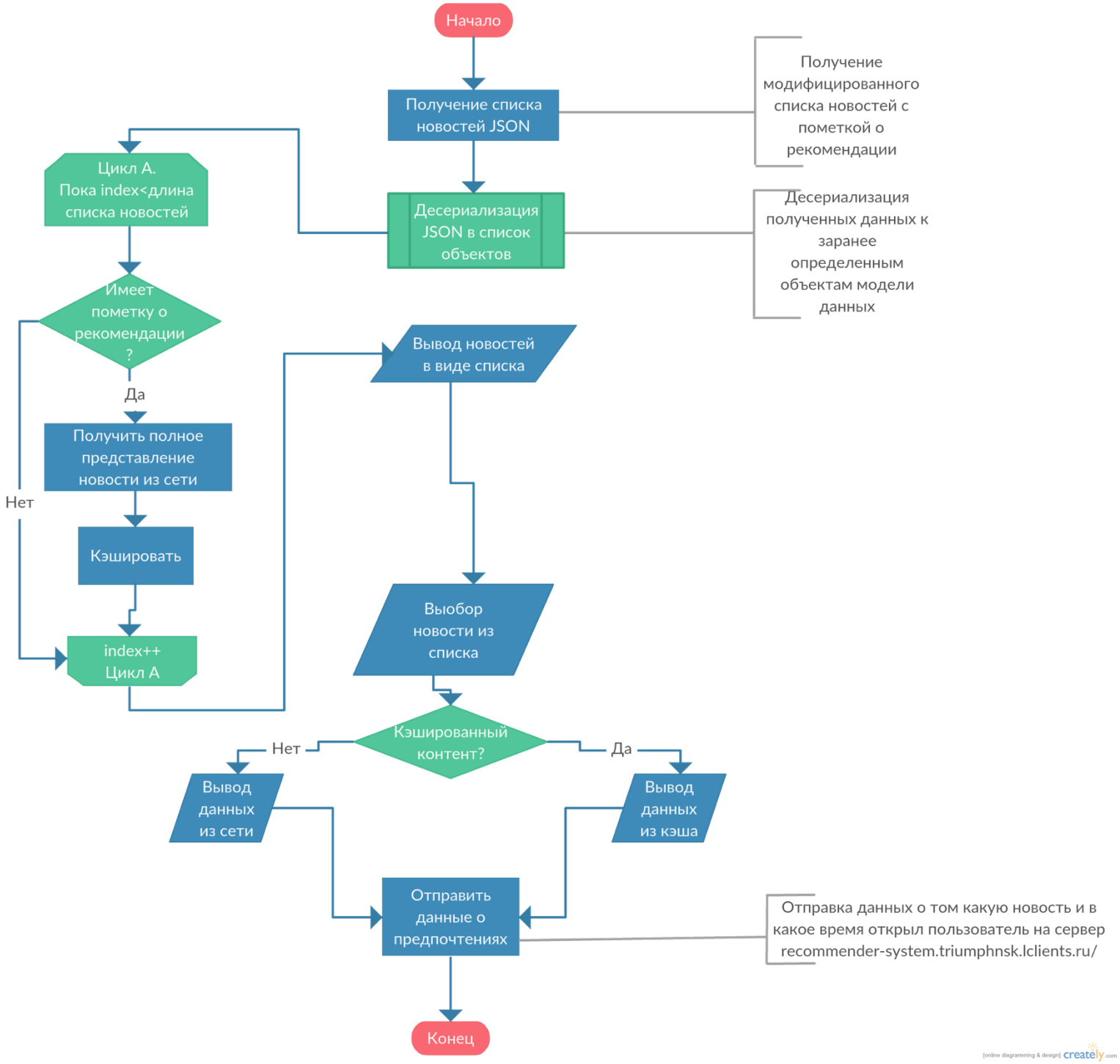


Рисунок 6. Блок-схема работы мобильного приложения

На представленной выше блок-схеме отображена работа основного алгоритма мобильного приложения. Данный алгоритм начинает работу с момента, когда мобильное приложение запускается. После запуска осуществляется GET-запрос к приложению, расположенному на сервере recommender-system.triumphnsk.lclients.ru. Этот запрос возвращает список новостей с пометкой о рекомендации в виде JSON (приложение 5). Далее

происходит операция десериализации полученных данных и привидения их к заранее определенной модели. После этого полученные объекты выводятся в виде списка новостей в пользовательский интерфейс. Когда объекты выведены, приложение начинает операцию кэширования данных. Эта операция заключатся в том, что новости, имеющие пометку о том, что они рекомендованы загружаются по ссылке и сохраняются в кэш. Далее пользователь выбирает интересующую его новость из списка и проверяется условие является ли эта новость закэшированной, если это так, то новость загружается из кэша, в противном случае, новость загружается из сети.

На рисунке 7 представлен основной алгоритм работы серверного приложения.

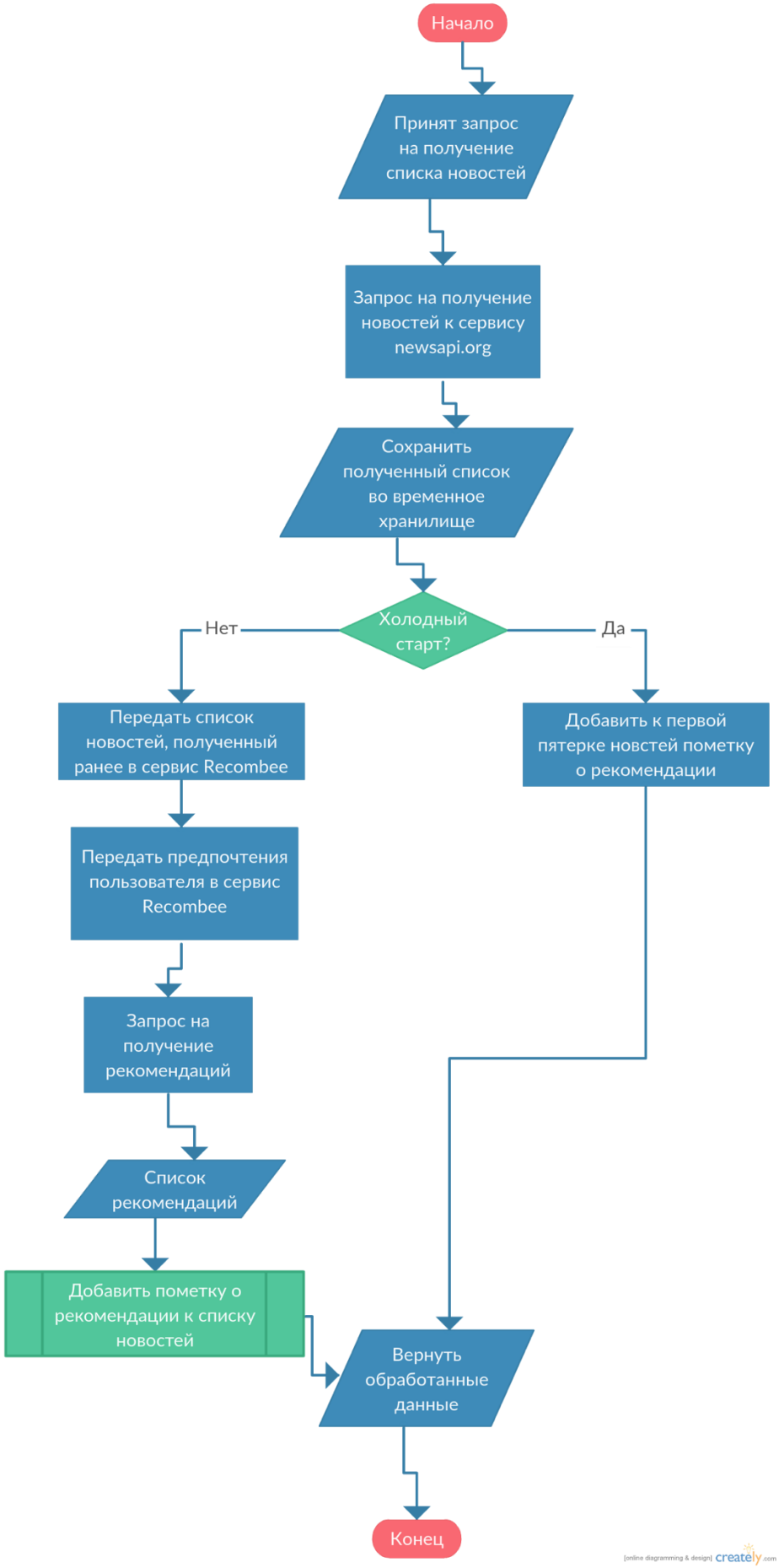


Рисунок 7. Алгоритм возврата списка новостей с рекомендациями

На представленной выше блок-схеме изображен алгоритм возврата списка новостей с рекомендациями. Данный алгоритм запускается в момент получения GET-запроса на получение списка новостей с мобильного приложения. После получения обращения, приложение делает запрос к сервису newsapi.org и в ответ получает JSON, содержащий список новостей (приложение 3). После этого JSON сохраняется во временное хранилище. Далее определяется есть ли данные о пользовательских предпочтениях, если нет, то считается, что произошла ситуация холодного старта и к первой пятерке новостей добавляется пометка того, что они рекомендованы, в ином случае, осуществляется работа с сервисом рекомендаций Recombee. Работа с сервисом рекомендаций осуществляется путем передачи списка новостей, а затем, и передачей данных о накопленных предпочтениях пользователя. Далее осуществляется запрос на получение рекомендаций отправленных до этого, в ответ сервис рекомендаций отправляет список рекомендованных новостей. Далее в списке новостей от сервиса newsapi.org добавляются пометки о том рекомендована новость или нет, путем сравнения со списком рекомендаций от сервиса recombee.

Для проектирования работы мобильного приложения была выбрана диаграмма Sequence Diagram. На рисунке 8 представлена диаграмма последовательностей работы системы, при получении списка новостей и передачи данных во фрагмент отображения списка новостей.

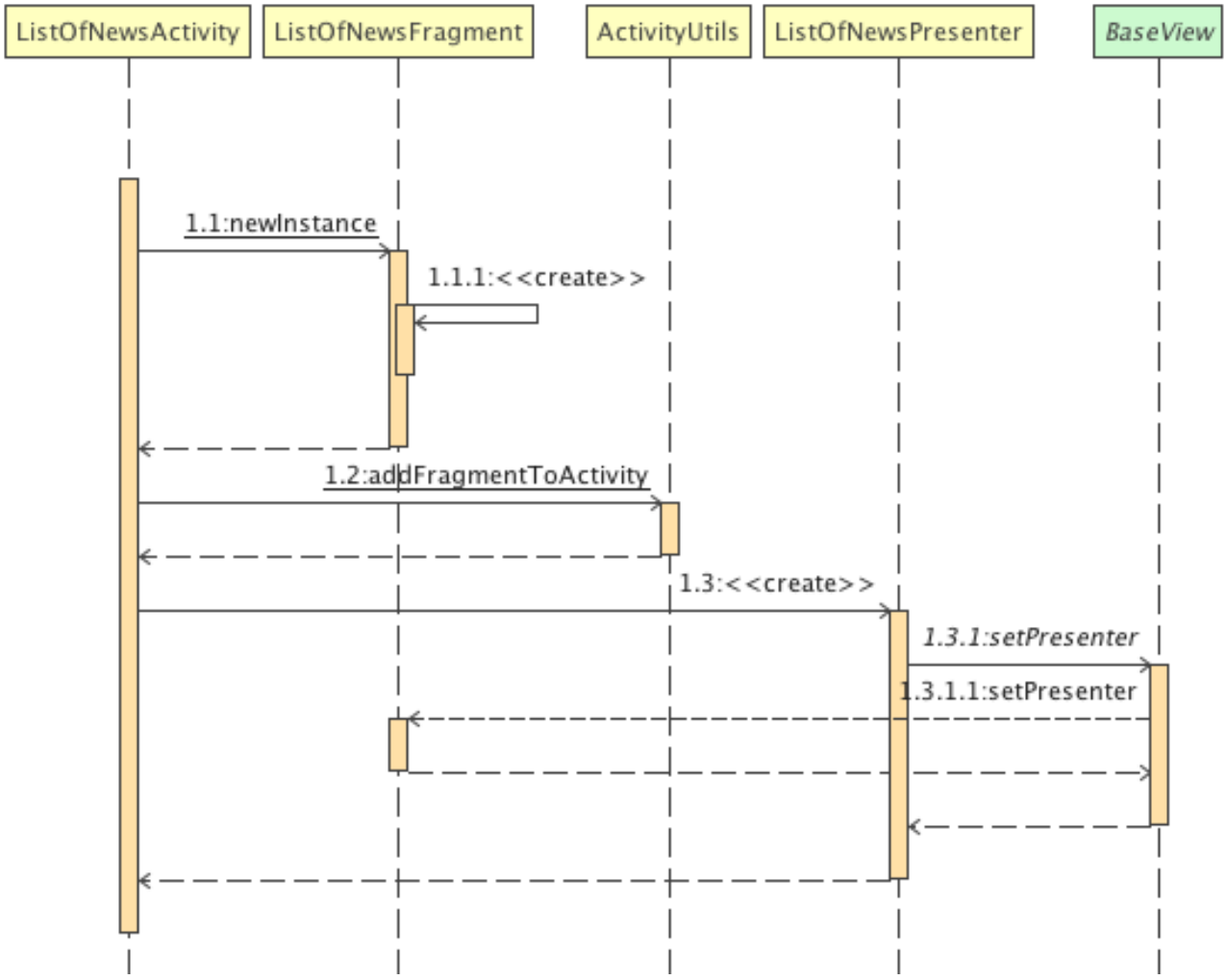


Рисунок 8. Диаграмма последовательностей

Когда данные получены, они передаются во фрагмент, жизненным циклом которого управляет родительское активити, а его методами управляет презентор. Диаграмма последовательностей фрагмента, отображающего список новостей представлена на рисунке 9.

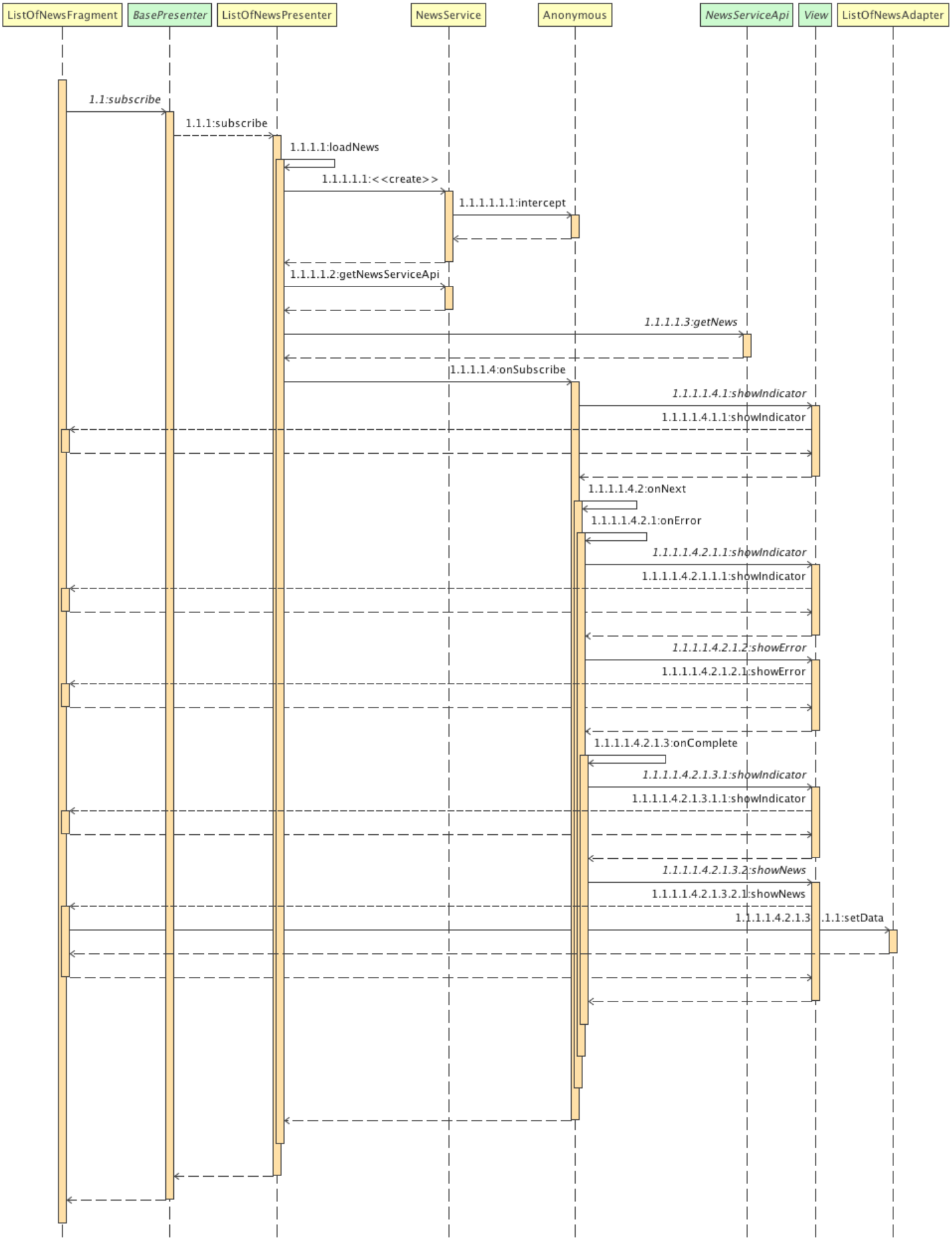


Рисунок 9. Диаграмма последовательностей

На данной диаграмме изображена последовательность получения данных от интерфейсов систем. Получение данных происходит в фоновом потоке с применением приемов реактивного программировнаия. После того, как данные получены и переданы в представление, представленное на рисунке 12, исчезает индикатор загрузки и появляется список контента.

Пользователь через интерфейс мобильного приложения просматривает список новостей, получаемый из открытого API, рисунок 10.

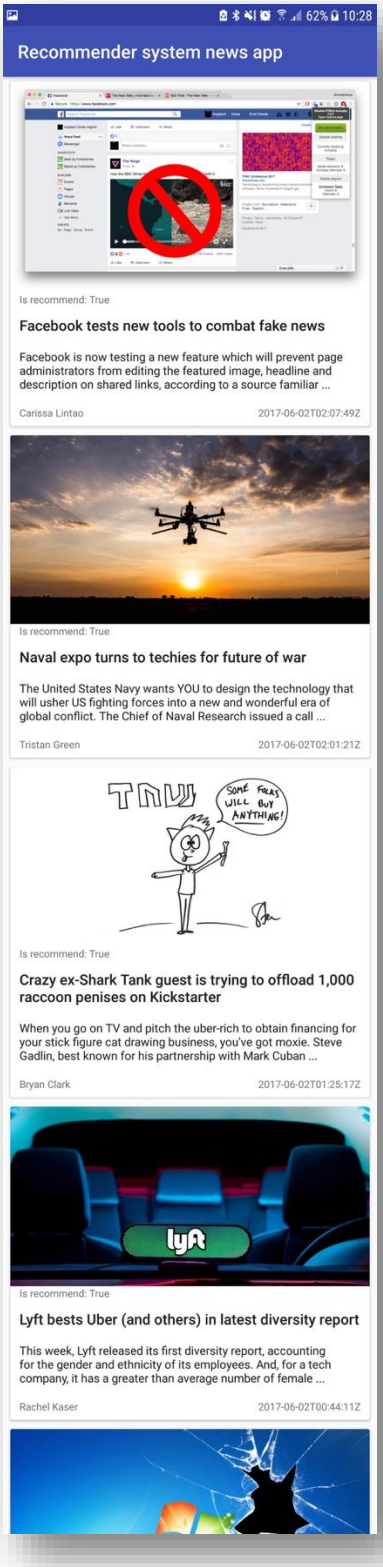


Рисунок 10. Просмотр списка новостей

Просмотр списка контролируется презентором. Презентор делает запрос к API recommender-system.triumphnsk.lclients.ru и получает список новостей с пометкам о рекомендации. После отображения списка, презентор кэширует

новости помеченные как рекомендуемые. И отображает весь список пользователю. Пользователь открывает заинтересовавшую его новость, вызывая у презентора метод получения детального представления новости. В этот момент запускается новое активити содержащее в себе контейнер для отображения новости. Диаграмма последовательностей представлена на рисунке 11.

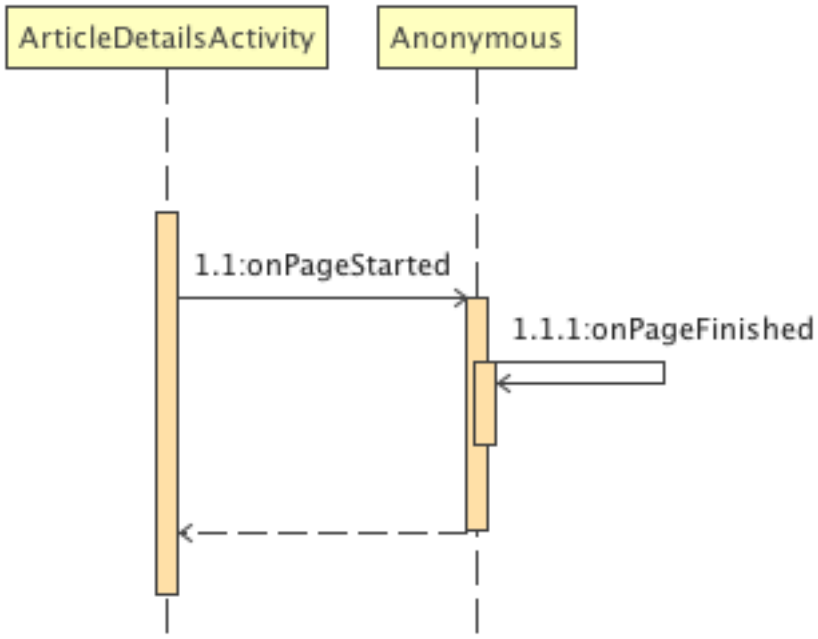


Рисунок 11. Диаграмма последовательностей

Если новость закэширована, то она подгружается из кэша, если ее нет в кэше, то новость подгружается по адресу, указанному как поле объекта новости. При этом, презентор в любом случае отправляет данные о том, какая новость заинтересовала пользователя, тем самым обучая систему. Время загрузки кэшируемой новости и новости из сети логируется. Пример открытой новости представлен на рисунке 12.

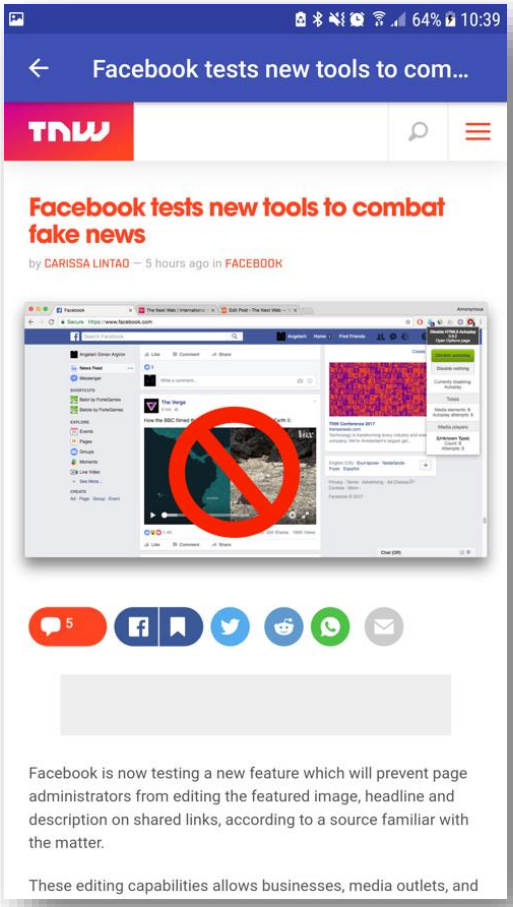


Рисунок 12. Детальное представление новости

Пример логирования времени открытия представлен на рисунке 13.

```
06-02 10:44:44.843 30732-30732/alexander.recommendersystemnewsapp I/Start_Loading: Fri Jun 02 10:44:44 GMT+05:00 2017
06-02 10:44:45.851 30732-30732/alexander.recommendersystemnewsapp I/End_Loading: Fri Jun 02 10:44:45 GMT+05:00 2017
06-02 10:44:46.843 30732-30732/alexander.recommendersystemnewsapp I/End_Loading: Fri Jun 02 10:44:46 GMT+05:00 2017
06-02 10:45:08.968 30732-30732/alexander.recommendersystemnewsapp I/Start_Loading: Fri Jun 02 10:45:08 GMT+05:00 2017
06-02 10:45:09.718 30732-30732/alexander.recommendersystemnewsapp I/End_Loading: Fri Jun 02 10:45:09 GMT+05:00 2017
06-02 10:45:13.747 30732-30732/alexander.recommendersystemnewsapp I/End_Loading: Fri Jun 02 10:45:13 GMT+05:00 2017
```

Рисунок 13. Логирование времени загрузки

Для реализации мобильного приложения, следуя практикам MVP, была разработана диаграмма классов, представленная на рисунке 14.

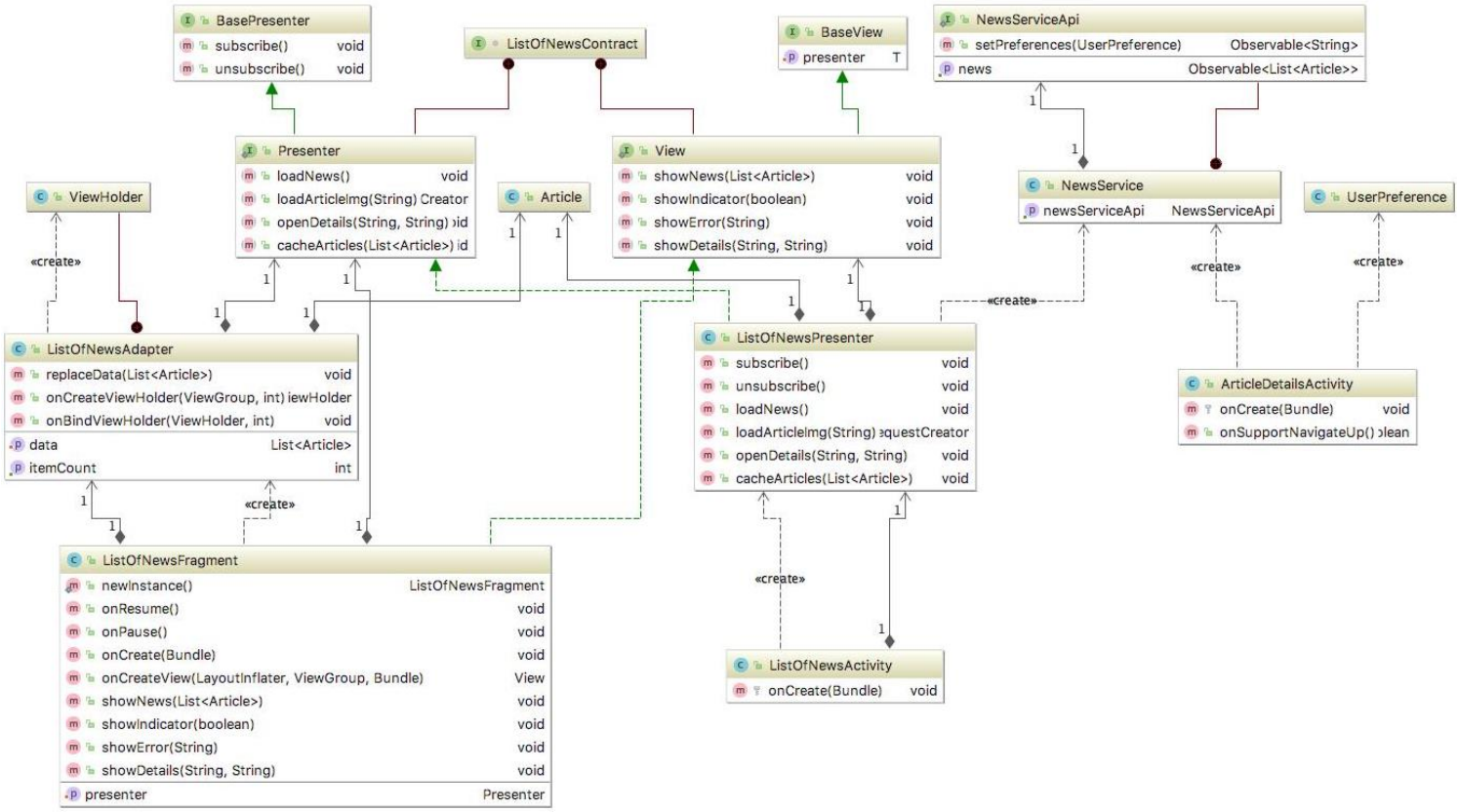


Рисунок 14. Диаграмма классов

4.3 Реализация

Для реализации этой задачи был использован сервис Recombee, который составляет список рекомендаций. Это условно бесплатный сервис, который предоставляет рекомендации через RESTful API. Для реализации требуется запустить сервер, на котором описывается логика подключения к Recombee. В роли сервера служит виртуальный хостинг с поддержкой python по адресу http://recommender-system.triumphnsk.lclients.ru/.

Было настроено окружение, и создан проект, разработанный на языке python с применением фреймворка Django.

Для создания проекта требуется подключится к виртуальному хостингу через SSH-соединение и установить менеджер пакетов pip с помощью команды:

```
python get-pip.py
```

Далее было развернуто и активировано виртуальное окружение, с помощью команд:

```
sudo pip install virtualenv // установка
mkvirtualenv venv // создание окружения
source bin/activate // активация окружения
```

Далее требуется установить фреймворк Django, создать проект, создать приложение и создать суперпользователя, выполнив команды:

```
sudo pip install django //установка
python django-admin.py startproject recommender-system //создание проекта
python manage.py startapp recommender // создание приложения
python manage.py createsuperuser // создание суперпользователя
```

После этого были произведены основные настройки проекта, представленные в приложении 7.

Затем были разработаны модели данных: NewsData, Article, UserPreferences.

Модель NewsData:

```
class NewsData(models.Model):
    status = models.CharField(max_length=100)
    source = models.CharField(max_length=100)
    sortBy = models.CharField(max_length=10)
    articles = models.ForeignKey(Article)
```

Поля модели соответствуют полям корневого узла JSON (приложение 3), получаемого с сервиса newsapi.org.

Модель Article:

```
class Article(models.Model):
    author = models.CharField(max_length=255)
    title = models.CharField(max_length=255)
    description = models.TextField()
    url = models.URLField()
    urlToImage = models.URLField()
    publishedAt = models.CharField(max_length=100)
```

Поля данной модели соответствуют полям элемента массива “articles”.

Модель UserPreferences:

```
class UserPreferences(models.Model):
```

```
user = models.User
models.ForeignKey(Article)
```

Данная модель представляет собой данные о предпочтениях пользователя, содержит в себе поле соответствующее текущему пользователю и ссылку на модель “Article”.

После создания модель была выполнена миграция с помощью команды: python manage.py migrate \\ в базе данных создались соответствующие таблицы

Далее был подключен сервис Recombee. Для подключения Recombee требуется добавить его в окружение с помощью команды в терминале:

```
$ pip install recombee-api-client
```

Затем были созданы представления которые получают список новостей от сервиса newsapi.org, получают список рекомендаций, модифицируют полученный список новостей, отправляют запросы к сервису recombee на добавление пользовательских предпочтений и списка новостей. С полным кодом основного представления можно ознакомиться в приложении 8.

Далее для этих представлений были описаны URL-адреса для доступа к ним.

```
urlpatterns = [
 url(r'^get_news$', views. get_news, name='get_news'),
 url(r'^set_preferences$', views. set_preferences, name='set_preferences'),
 url(r'^admin/', admin.site.urls),]
```

При обращении по вышеуказанным адресам вызывается соответствующий метод представления и возвращает результат.

Далее было реализовано мобильное приложение. Диаграмма классов представлена в приложении 1.

При создании проекта были добавлены следующие зависимости (таблица 2):

Таблица 2. Зависимости мобильного приложения

Зависимость	**Назначение\основной функционал**

com.android.support:appcompat-v7:25.3.1	Библиотека поддержки
com.android.support:design:25.3.1	Добавляет возможность использовать элементы Google Material Design
com.android.support:cardview-v7:25.3.1	Добавляет возможность использовать элемент «карточка» в интерфейсе пользователя
com.android.support:recyclerview-v7:25.3.1	Позволяет выводить повторяющиеся элементы верстки, например в виде списка или таблицы
io.reactivex.rxjava2:rxjava:2.1.0	Добавляет возможность использования подхода реактивного программирования на Java
io.reactivex.rxjava2:rxandroid:2.0.1	Вспомогательная библиотека для rxjava2
com.jakewharton.retrofit:retrofit2-rxjava2-adapter:1.0.0	Адаптер, позволяющий использовать Retrofit в реактивном стиле
com.squareup.retrofit2:retrofit:2.3.0	Type-Safe клиент
com.squareup.retrofit2:converter-gson:2.3.0	Библиотека автоматически конвертирующая JSON в объект Java
com.squareup.picasso:picasso:2.5.2	Вывод изображений

После этого разработан класс NewsService, который отвечает за взаимодействие между мобильным приложением и бэкенд приложением.

```
public class NewsService {
    private static final String URL = "http://recommender-system.triumphnsk.lclients.ru/";
    public NewsService(){
        OkHttpClient client = new OkHttpClient.Builder()
                .addInterceptor(new Interceptor() {
                    @Override
```

```
                public Response intercept(@NonNull Chain chain) throws IOException {
                    Request original = chain.request();
                    Request request = original.newBuilder()
                            .cacheControl(new
CacheControl.Builder().noCache().build()).build();
                    return chain.proceed(request);
                }}).build();
        Retrofit retrofit = new Retrofit.Builder()
.baseUrl(URL).addCallAdapterFactory(RxJava2CallAdapterFactory.create()).addConver
terFactory(GsonConverterFactory.create()).client(client).build();
        mNewsServiceApi = retrofit.create(NewsServiceApi.class);
    }
    private NewsServiceApi mNewsServiceApi;
    public NewsServiceApi getNewsServiceApi() {
        return this.mNewsServiceApi;
    }
    public interface NewsServiceApi {
        @GET("get_news")
        Observable<List<Article>> getNews();
        @POST("set_preferences")
        Observable<String> setPreferences(@Body UserPreference preference);
    }}
```

После этого был создан класс CacheProvider, который отвечает за кэширование новостей. Содержит в себе один статический метод cacheCurrentArticle.

```
public static void cacheCurrentArticle(String Url) {
    URL url = null;
    try {
        url = new URL(Url);
    } catch (MalformedURLException e) {
        e.printStackTrace();
    }
    URLConnection connection = new URLConnection(url) {
        @Override
        public void connect() throws IOException {
```

```
            try {
                File httpCacheDir = new File(context.getCacheDir(), "http");
                long httpCacheSize = 10 * 1024 * 1024; // 10 MiB
                HttpResponseCache.install(httpCacheDir, httpCacheSize);
            } catch (IOException e) {
                Log.i(TAG, "HTTP response cache installation failed:" + e);
            }
        }
    };
    try {
        connection.connect();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

Данные сохраняются в директорию, которая также используется WebView отображающее детальное представление новости. Это значит, что при открытии новости в WebView автоматически принимается решение от куда загружать данные: из кэша или из сети.

Далее, для реализации мобильного приложения были представлены JSON-файлы (приложение 3, приложение 5) в виде POJO (приложение 4).

Следуя практикам MVP был описан контракт представления и презентора.

```
interface ListOfNewsContract {
    interface View extends BaseView<Presenter> {
        void showNews(List<Article> articles);
        void showIndicator(boolean isShow);
        void showError(String message);
        void showDetails(String articleURL, String title);
    }
    interface Presenter extends BasePresenter {
        void loadNews();
        RequestCreator loadArticleImg(String url);
        void openDetails(String articleURL, String title);
```

```
        void cacheArticles(List<Article> articles);
    }
}
```

После этого был описан реализован презентор, который имплементирует интерфейс презентора из контракта. Рассмотрим основной метод.

```
@Override
public void loadNews() {
        NewsService newsService = new NewsService();
        newsService.getNewsServiceApi().getNews()
                .subscribeOn(Schedulers.newThread())
                .observeOn(AndroidSchedulers.mainThread())
                .subscribe(new Observer<List<Article>>() {
                @Override
                public void onSubscribe(@NonNull Disposable d) {
                        mCompositeDisposable.add(d);
                        mView.showIndicator(true);
                    }
                    @Override
                    public void onNext(@NonNull List<Article> articles) {
                        if (!mArticles.isEmpty()) {
                            mArticles.clear();
                        }
                        mArticles.addAll(articles);
                                        List<Article> articlesForCache = new ArrayList();
                        for (Article article : mArticles) {
                            if (article.isRecommend){
                                articlesForCache.add(article);
                            }
                        }
                        cacheArticles(articlesForCache);

                    }

                    @Override
```

```
            public void onError(@NonNull Throwable e) {
                mView.showIndicator(false);
                mView.showError(e.getMessage());
            }

            @Override
            public void onComplete() {
                mView.showIndicator(false);
                mView.showNews(mArticles);
            }
        });
    }
```

Получение списка новостей проходит в потоке, отличном от главного. Для получения списка новостей, был использован интерфейс получения новостей, который «излучает» объекты Article (приложение 6). Данный процесс сконфигурирован таким образом, что получение данных происходит в фоновом процессе, а полученный результат «излучается» в основной поток. В данном случае подписка на излучателей происходит здесь же. Когда данные получены, они сохраняются в лист, содержащий объекты класс Article, также происходит операция кеширования новостей, у которых поле isRecommend равно true. По завершении операции сохраненный список объектов Article передается в представление, которое отвечает за вывод информации в пользовательский интерфейс.

При открытии детального представления новости, запускается активити ArticleDetailsActivity, которое содержит в себе WebView на основе движка Chrome. Данное WebView в автоматическом режиме проверяет данные в кэше и при наличии таковых, загружает контент из памяти устройства, в противном случае осуществляется переход по ссылке. Рассмотрим основной метод, который срабатывает когда данные загружены и требуется передать данные о предпочтениях пользователя.

```
@Override
public void onPageFinished(WebView view, String url) {
```

```
            super.onPageFinished(view, url);
            mProgressBar.setVisibility(View.GONE);
            mWebView.setVisibility(View.VISIBLE);
            Date date = new Date();
            Log.i("End_Loading", date.toString());
            NewsService newsService = new NewsService();
newsService.getNewsServiceApi().setPreferences(new
UserPreference(url)).observeOn(Schedulers.newThread()).subscribe(new
Observer<String>() {@Override
        public void onSubscribe(@NonNull Disposable d) {
        mCompositeDisposable.add(d);}
        @Override
        public void onNext(@NonNull String s) {
        Log.i("Set_user_pref: ", s);}
        @Override
        public void onError(@NonNull Throwable e) {
Log.e(ArticleDetailsActivity.class.getName(), e.toString());}
        @Override
        public void onComplete() {
        mCompositeDisposable.clear();
        }});}
```

В данном методе скрывается индикатор загрузки, и появляется страничка, содержащая детальное представление новости. Затем логируется время завершения загрузки. После этого отправляются данные о пользовательских предпочтениях. В данном случае данные отправляются в фоновом процессе, по завершению операции поток уничтожается.

В ходе разработки была описана логика взаимодействия с сервером, созданы интерфейсы, описана логика кэширования рекомендаций, созданы представления отображения списка новостей, детальное представление новости. Рекомендации ориентированы на детальное представление новости, т.е. список новостей отображается в том порядке, в котором он отображается на сайте, а детальное отображение новости уже кэшируется в фоне.

Для реализации мобильного приложения была разработана следующая иерархия пакетов и классов, представленная на рисунке 15.

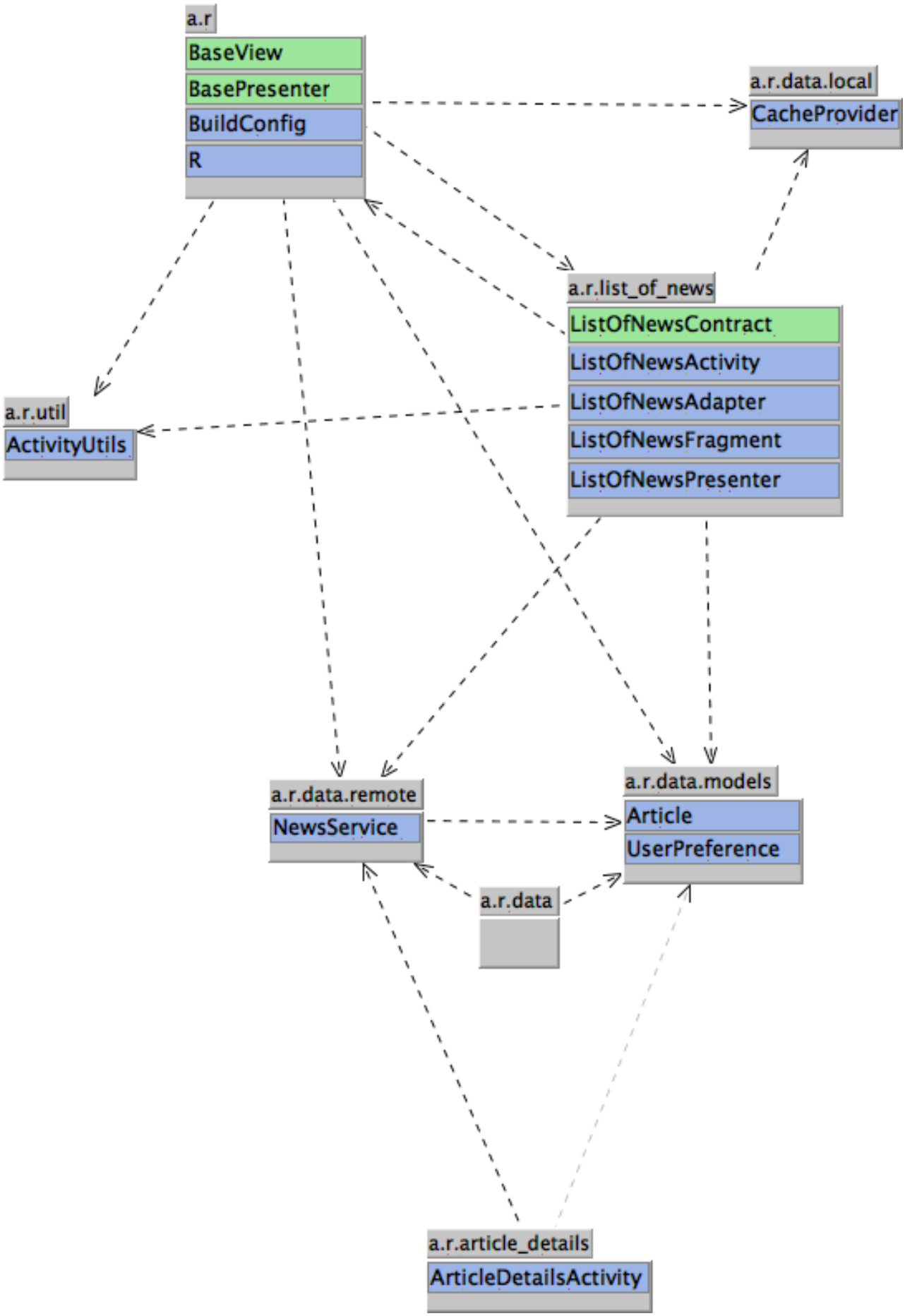


Рисунок 15. Иерархия пакетов и классов

Рассмотрим их подробнее. В пакете article_details находится активити, отображающее детальное представление новости. В пакете data.models находятся модели данных. В пакете data.remote находится класс, описывающий сетевое взаимодействие с API. В пакете data.local находится класс, отвечающий за кэширование данных. Пакет list_of_news содержит в себе класс активити, который является контейнером для фрагмента, который отображает список новостей, с помощью адаптера, кроме того, в этом пакете содержится контракт для представления и презентора, также в пакете находится и сам презетор. В пакете utils содержится класс, являющийся утилитой для добавления фрагмента в контейнер. В корне приложения находятся 2 интерфейса классов, описывающих базовое представление и презентор.

4.4 Внедрение разработки

Данная разработка зарекомендовала себя с положительной стороны. Было принято решение о внедрении в мобильное приложение Origami. Данное приложение входит в часть системы Origami. Это кроссплатформенная система для создания электронных версий журналов. Клиентами данной платформы являются международные организации Save the children, энергетическая компания Eltek и другие. Журнал может быть трех видов: Origami View, где есть платные каналы с журналами, Origami Kiosk для магазинов и просто standalone приложения журналов. Просмотр журналов осуществляется в том числе и на мобильных устройствах, пример представлен на рисунке 16.

Рисунок 16. Пример журнала в Origami

Из-за сложной верстки, обилия контента и сложной логики взаимодействия с сетью, на некоторых устройствах выявляются проблемы со скоростью отклика приложения. Для решения данной проблемы было принято решение внедрения разработанной системы. Для внедрения системы, нужно всего лишь добавить классы, описывающие взаимодействие с контентом и рекомендательной системой.

Заключение

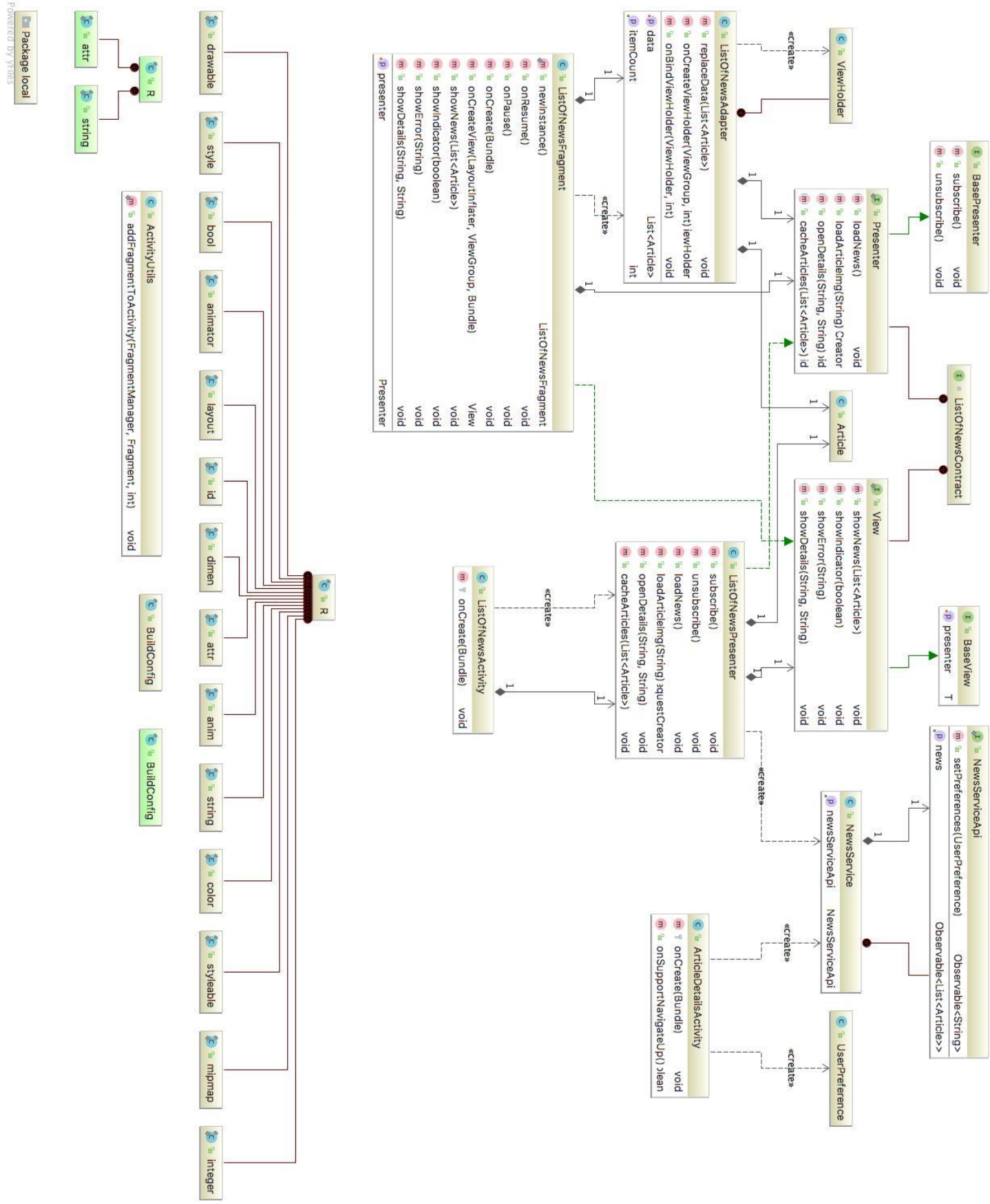
В ходе работы была выведена концепция нового применения рекомендательных систем в мобильных приложениях, в целях улучшения отклика, разработано бэкенд-приложение, разработан пакет классов, реализующий возможность такого применения рекомендательных систем, разработано тестовое приложение, в которое встроено, разрабатываемое решение.

Была выбрана и построена современная архитектура приложения, спроектирована система, разработано мобильное приложение, разработан бэкенд, сконфигурирована рекомендательная система, разработан пакет классов, позволяющих использовать рекомендательную систему в целях предективной подгрузки данных.

Данная разработка внедряется в мобильное приложение с широкой аудиторией по всему миру.

На основании этого, в дальнейшем будет развиваться тема применения рекомендательных систем в целях улучшения отклика. Планируется создание сервиса, который можно будет внедрить в любое мобильное приложение под операционную систему Android, который будет собирать статистику и на основании этого делать рекомендации, которые будут призваны увеличить скорость работы мобильных приложений.

Приложение 1. Диаграмма классов мобильного приложения



Приложение 2. Результаты эксперимента

№ п.п.	Время начала	Время окончания	Разница	Рекомендованный контент
1	0:48:38.48	0:48:39.48	0:00:01.01	ДА
2	0:52:54.52	0:52:58.52	0:00:04.04	НЕТ
3	1:09:32.9	1:09:34.9	0:00:02.02	ДА
4	1:49:42.49	1:49:44.49	0:00:02.02	ДА
5	2:17:35.17	2:17:41.17	0:00:05.05	НЕТ
6	2:28:40.28	2:28:47.28	0:00:07.07	НЕТ
7	2:28:46.28	2:28:49.28	0:00:03.03	НЕТ
8	2:57:09.57	2:57:12.57	0:00:03.03	НЕТ
9	3:09:47.9	3:09:51.9	0:00:04.04	НЕТ
10	3:19:48.19	3:19:49.19	0:00:02.02	ДА
11	3:58:22.58	3:58:28.58	0:00:07.07	НЕТ
12	4:02:06.2	4:02:10.2	0:00:04.04	НЕТ
13	4:04:12.4	4:04:14.4	0:00:02.02	ДА
14	4:23:15.23	4:23:23.23	0:00:08.08	НЕТ
15	4:30:14.30	4:30:16.30	0:00:02.02	ДА
16	4:33:58.33	4:33:59.33	0:00:01.01	ДА
17	4:36:13.36	4:36:21.36	0:00:08.08	НЕТ
18	4:47:27.47	4:47:36.47	0:00:09.09	НЕТ
19	4:55:17.55	4:55:20.55	0:00:03.03	НЕТ
20	5:10:53.10	5:11:00.11	0:00:07.07	НЕТ
21	5:15:59.15	5:16:06.16	0:00:07.07	НЕТ
22	5:25:07.25	5:25:09.25	0:00:02.02	ДА
23	5:36:08.36	5:36:14.36	0:00:05.05	НЕТ
24	5:46:30.46	5:46:37.46	0:00:07.07	НЕТ
25	5:56:43.56	5:56:47.56	0:00:03.03	НЕТ
26	6:01:04.1	6:01:05.1	0:00:01.01	ДА
27	6:06:51.6	6:06:53.6	0:00:01.01	ДА
28	6:11:21.11	6:11:23.11	0:00:03.03	НЕТ
29	6:32:31.32	6:32:39.32	0:00:08.08	НЕТ
30	6:35:04.35	6:35:10.35	0:00:06.06	НЕТ
31	7:01:42.1	7:01:46.1	0:00:04.04	НЕТ
32	7:15:42.15	7:15:43.15	0:00:01.01	ДА
33	7:28:57.28	7:29:05.29	0:00:08.08	НЕТ
34	7:33:53.33	7:33:57.33	0:00:04.04	НЕТ
35	7:40:28.40	7:40:35.40	0:00:07.07	НЕТ
36	7:48:29.48	7:48:33.48	0:00:04.04	НЕТ
37	8:13:26.13	8:13:32.13	0:00:06.06	НЕТ
38	8:24:12.24	8:24:15.24	0:00:03.03	НЕТ

39	8:28:36.28	8:28:38.28	0:00:02.02	ДА
40	8:47:59.47	8:48:01.48	0:00:02.02	ДА
41	8:59:24.59	8:59:27.59	0:00:04.04	НЕТ
42	9:02:34.2	9:02:36.2	0:00:02.02	ДА
43	9:26:00.26	9:26:09.26	0:00:09.09	НЕТ
44	9:34:37.34	9:34:45.34	0:00:08.08	НЕТ
45	9:41:14.41	9:41:16.41	0:00:02.02	ДА
46	10:05:22.5	10:05:29.5	0:00:07.07	НЕТ
47	10:06:54.6	10:06:56.6	0:00:02.02	ДА
48	10:32:18.32	10:32:19.32	0:00:01.01	ДА
49	10:34:10.34	10:34:18.34	0:00:08.08	НЕТ
50	11:04:55.4	11:04:56.4	0:00:01.01	ДА
51	11:19:19.19	11:19:21.19	0:00:02.02	ДА
52	11:59:12.59	11:59:19.59	0:00:07.07	НЕТ
53	12:00:43.0	12:00:49.0	0:00:06.06	НЕТ
54	12:15:36.15	12:15:39.15	0:00:03.03	НЕТ
55	12:47:31.47	12:47:39.47	0:00:09.09	НЕТ
56	13:02:38.2	13:02:42.2	0:00:04.04	НЕТ
57	13:52:20.52	13:52:22.52	0:00:02.02	ДА
58	14:27:43.27	14:27:52.27	0:00:09.09	НЕТ
59	14:31:24.31	14:31:32.31	0:00:08.08	НЕТ
60	14:31:49.31	14:31:57.31	0:00:07.07	НЕТ
61	14:42:13.42	14:42:17.42	0:00:04.04	НЕТ
62	14:57:01.57	14:57:08.57	0:00:07.07	НЕТ
63	15:06:09.6	15:06:16.6	0:00:07.07	НЕТ
64	15:31:26.31	15:31:31.31	0:00:06.06	НЕТ
65	16:47:26.47	16:47:30.47	0:00:04.04	НЕТ
66	16:50:37.50	16:50:43.50	0:00:07.07	НЕТ
67	17:04:19.4	17:04:21.4	0:00:01.01	ДА
68	17:06:13.6	17:06:17.6	0:00:04.04	НЕТ
69	17:08:43.8	17:08:44.8	0:00:01.01	ДА
70	17:11:36.11	17:11:37.11	0:00:01.01	ДА
71	17:17:12.17	17:17:16.17	0:00:03.03	НЕТ
72	17:35:21.35	17:35:23.35	0:00:02.02	ДА
73	17:59:54.59	18:00:01.0	0:00:07.07	НЕТ
74	18:24:49.24	18:24:54.24	0:00:05.05	НЕТ
75	18:28:56.28	18:29:03.29	0:00:07.07	НЕТ
76	18:48:15.48	18:48:22.48	0:00:07.07	НЕТ
77	18:56:04.56	18:56:12.56	0:00:08.08	НЕТ
78	18:57:44.57	18:57:51.57	0:00:06.06	НЕТ
79	19:06:51.6	19:06:55.6	0:00:04.04	НЕТ
80	19:14:59.14	19:15:01.15	0:00:02.02	ДА
81	19:17:50.17	19:17:54.17	0:00:04.04	НЕТ
82	19:28:05.28	19:28:09.28	0:00:04.04	НЕТ

83	19:28:20.28	19:28:26.28	0:00:06.06	НЕТ
84	19:50:16.50	19:50:20.50	0:00:04.04	НЕТ
85	20:10:28.10	20:10:33.10	0:00:04.04	НЕТ
86	20:24:15.24	20:24:18.24	0:00:03.03	НЕТ
87	20:45:16.45	20:45:19.45	0:00:03.03	НЕТ
88	21:07:45.7	21:07:49.7	0:00:04.04	НЕТ
89	21:20:16.20	21:20:24.20	0:00:07.07	НЕТ
90	21:24:41.24	21:24:42.24	0:00:01.01	ДА
91	21:24:46.24	21:24:47.24	0:00:01.01	ДА
92	21:25:03.25	21:25:06.25	0:00:03.03	НЕТ
93	21:26:02.26	21:26:04.26	0:00:02.02	ДА
94	21:37:18.37	21:37:21.37	0:00:03.03	НЕТ
95	21:47:00.47	21:47:02.47	0:00:02.02	ДА
96	22:16:03.16	22:16:06.16	0:00:03.03	НЕТ
97	23:22:14.22	23:22:22.22	0:00:08.08	НЕТ
98	23:40:59.40	23:41:01.41	0:00:02.02	ДА
99	23:42:43.42	23:42:50.42	0:00:07.07	НЕТ
100	23:56:46.56	23:56:49.56	0:00:03.03	НЕТ

Приложение 3. Пример JSON файла


```
{
   "status": "ok",
   "source": "google-news",
   "sortBy": "top",
   "articles": [{
      "author": "Julie Hirschfeld Davis",
      "title": "Trump’s Cabinet, With a Prod, Extols the ‘Blessing’ of Serving Him",
      "description": "President Trump transformed a routine meeting into a mood-boosting display of support. Vice President Mike Pence called his role “the greatest privilege of my life.”",
      "url": "https://www.nytimes.com/2017/06/12/us/politics/trump-boasts-of-record-setting-pace-of-activity.html",
      "urlToImage": "https://static01.nyt.com/images/2017/06/13/us/13dc-trump/13dc-trump-facebookJumbo-v2.jpg",
      "publishedAt": "2017-06-13T01:36:48Z"
   }, {
      "author": "Michael D. Shear and Maggie Haberman",
      "title": "Friend Says Trump Is Considering Firing Mueller as Special Counsel",
      "description": "Conservative allies of the president have started attacking the credibility of Robert S. Mueller III, the special counsel in the Russia investigation.",
      "url": "https://www.nytimes.com/2017/06/12/us/politics/robert-mueller-trump.html",
      "urlToImage": "https://static01.nyt.com/images/2017/06/13/us/13dc-mueller-1/13dc-mueller-1-facebookJumbo.jpg",
      "publishedAt": "2017-06-13T02:48:22Z"
   }, {
      "author": "https://www.facebook.com/NBCNews",
      "title": "D.C., Maryland Officials have hit President Trump with a lawsuit",
      "description": "Washington D.C. and Maryland officials say Trump may be violating a constitutional clause barring presidents from cashing in on the presidency",
      "url": "http://www.nbcnews.com/news/us-news/d-c-maryland-officials-hit-president-trump-lawsuit-n771011",
      "urlToImage": "https://media4.s-nbcnews.com/j/newscms/2017_24/2034911/170612-maryland-ag-brian-dc-ag-karl-racine-se-118p_7e65f46e074d1f738a571dc107f0c2db.nbcnews-fp-1200-800.jpg",
      "publishedAt": "2017-06-12T00:00:00Z"
   }, {
```

"author": null,
"title": "The Latest: Hawaii official: Ruling shows US system works",
"description": "The Latest on a U.S. appeals court keeping President Donald Trump’s travel ban blocked (all times local):",
"url": "https://www.washingtonpost.com/national/religion/the-latest-hawaii-official-expects-travel-ban-appeal/2017/06/12/f4af83f0-4fcc-11e7-b74e-0d2785d3083d_story.html",
"urlToImage": "https://img.washingtonpost.com/rf/image_1484w/2010-2019/Wires/Online/2017-06-13/AP/Images/Trump_99816.jpg-0119c.jpg",
"publishedAt": "2017-06-12T14:33:00Z"
}, {
"author": "Jeremy Diamond, CNN",
"title": "Post-Orlando Trump proposals not yet reality",
"description": "The day after the Pulse nightclub in Orlando became the site of the deadliest terrorist attack since 9/11, Trump produced the latest iteration of his Muslim ban",
"url": "http://www.cnn.com/2017/06/12/politics/trump-pulse-orlando-attack/index.html",
"urlToImage": "http://i2.cdn.cnn.com/cnnnext/dam/assets/170601052736-01-donald-trump-0531-super-tease.jpg",
"publishedAt": "2017-06-12T19:59:19Z"
}, {
"author": "Julia Edwards Ainsley",
"title": "Attorney General Sessions to talk publicly to U.S. Senate panel",
"description": "U.S. Attorney General Jeff Sessions' public testimony before a Senate panel on Tuesday sets up another potentially dramatic hearing on possible ties between President Donald Trump's campaign and Russian meddling in the 2016 presidential race.",
"url": "http://www.reuters.com/article/us-usa-trump-russia-sessions-idUSKBN1920QO",
"urlToImage": "https://s3.reutersmedia.net/resources/r/?m=02&d=20170613&t=2&i=1188721809&w=&fh=545px&fw=&ll=&pl=&sq=&r=LYNXMPED5B11Q",
"publishedAt": "2017-06-13T01:19:00Z"
}, {
"author": "https://www.facebook.com/annafifield",
"title": "Dennis Rodman is on his way to North Korea. Was he sent by Trump?",
"description": "The controversial basketballer hopes to meet Kim Jong Un again in Pyongyang",
"url": "https://www.washingtonpost.com/world/dennis-rodman-is-on-his-way-to-north-korea-was-he-sent-by-trump/2017/06/12/5c9c19cc-4fd9-11e7-a973-3dae94ed3eb7_story.html",

"urlToImage": "https://img.washingtonpost.com/rf/image_1484w/2010-2019/WashingtonPost/2017/06/13/Foreign/Images/2017-06-12T234646Z_1_LYNXMPED5B1UP-ORVPE_RTROPTP_3_PEOPLE-US-USA-NORTHKOREA-RODMAN.jpg",
"publishedAt": "2017-06-12T15:32:00Z"
}, {
"author": "Neil MacFarquhar and Ivan Nechepurenko",
"title": "Across Russia, Protesters Heed Navalny’s Anti-Kremlin Rallying Cry",
"description": "Aleksei A. Navalny, a leading Putin critic, was detained outside his home in Moscow as an extraordinary wave of antigovernment protests swept across Russia.",
"url": "https://www.nytimes.com/2017/06/12/world/europe/russia-aleksei-navalny-kremlin-protests.html",
"urlToImage": "https://static01.nyt.com/images/2017/06/12/us/12rusprotest/12rusprotest-facebookJumbo-v2.jpg",
"publishedAt": "2017-06-13T00:31:10Z"
}, {
"author": null,
"title": "Sean Spicer: Donald Trump To Reveal If He Taped James Comey When He’s Good And Ready",
"description": "“Does President Trump have audio recordings of his conversations and meetings with the former FBI Director James Comey?” a reporter asked White House Press Secretary Sean Spicer at Monday’s briefing, after Labor Secretary Alexander Acosta’s briefing show-and-tell on Trump’s apprenticeship program. “The",
"url": "https://www.yahoo.com/tv/sean-spicer-donald-trump-reveal-190903083.html",
"urlToImage": "https://s.yimg.com/uu/api/res/1.2/.DDcCT1j1wf65Bp6Jro15A--/aD02Njk7dz0xNjAwO3NtPTE7YXBwaWQ9eXRhY2h5b24-/http://media.zenfs.com/en-us/homerun/deadline.com/f407bcab7818c0ea5abba5192ee415b4",
"publishedAt": "2017-06-12T19:09:03Z"
}, {
"author": "BBC News",
"title": "Theresa May to hold talks with DUP leaders on government deal",
"description": "Theresa May is to meet DUP leader Arlene Foster to try to seal an agreement to enable her party to govern.",
"url": "http://www.bbc.com/news/election-2017-40255958",
"urlToImage": "https://ichef-1.bbci.co.uk/news/1024/cpsprodpb/543D/production/_96456512_fosterandmay.jpg",

```
        "publishedAt": "2017-06-13T02:10:23Z"
    }]
}
```

Приложение 4. POJO

Соответствует одному элементу массива новостей из JSON

```
public class Article {
    @SerializedName("author")
    public String author;
    @SerializedName("title")
    public String title;
    @SerializedName("description")
    public String description;
    @SerializedName("url")
    public String url;
    @SerializedName("urlToImage")
    public String urlToImage;
    @SerializedName("publishedAt")
    public String publishedAt;
    @SerializedName("isRecommend")
    public Boolean isRecommend;
}
```

Модель используется для отправки пользовательских предпочтений

```
public class UserPreference {
    @SerializedName("user_id")
    private String userId;
    @SerializedName("article_url")
    private String articleUrl;

    UserPreference(String articleUrl){
        this.userId = "1";
        this.articleUrl = articleUrl;
    }
}
```

Приложение 5. Модифицированный JSON

"articles": [{"author": "Julie Hirschfeld Davis",
"title": "Trump’s Cabinet, With a Prod, Extols the ‘Blessing’ of Serving Him",
"description": "President Trump transformed a routine meeting into a mood-boosting display of support. Vice President Mike Pence called his role “the greatest privilege of my life.”",
"url": "https://www.nytimes.com/2017/06/12/us/politics/trump-boasts-of-record-setting-pace-of-activity.html",
"urlToImage": "https://static01.nyt.com/images/2017/06/13/us/13dc-trump/13dc-trump-facebookJumbo-v2.jpg",
"publishedAt": "2017-06-13T01:36:48Z"
"isRecommend": True
}, {
"author": "Michael D. Shear and Maggie Haberman",
"title": "Friend Says Trump Is Considering Firing Mueller as Special Counsel",
"description": "Conservative allies of the president have started attacking the credibility of Robert S. Mueller III, the special counsel in the Russia investigation.",
"url": "https://www.nytimes.com/2017/06/12/us/politics/robert-mueller-trump.html",
"urlToImage": "https://static01.nyt.com/images/2017/06/13/us/13dc-mueller-1/13dc-mueller-1-facebookJumbo.jpg",
"publishedAt": "2017-06-13T02:48:22Z"
"isRecommend": True
}, {
"author": "https://www.facebook.com/NBCNews",
"title": "D.C., Maryland Officials have hit President Trump with a lawsuit",
"description": "Washington D.C. and Maryland officials say Trump may be violating a constitutional clause barring presidents from cashing in on the presidency",
"url": "http://www.nbcnews.com/news/us-news/d-c-maryland-officials-hit-president-trump-lawsuit-n771011",
"urlToImage": "https://media4.s-nbcnews.com/j/newscms/2017_24/2034911/170612-maryland-ag-brian-dc-ag-karl-racine-se-118p_7e65f46e074d1f738a571dc107f0c2db.nbcnews-fp-1200-800.jpg",
"publishedAt": "2017-06-12T00:00:00Z"
"isRecommend": True
}, {
"author": null,
"title": "The Latest: Hawaii official: Ruling shows US system works",

"description": "The Latest on a U.S. appeals court keeping President Donald Trump’s travel ban blocked (all times local):",
"url": "https://www.washingtonpost.com/national/religion/the-latest-hawaii-official-expects-travel-ban-appeal/2017/06/12/f4af83f0-4fcc-11e7-b74e-0d2785d3083d_story.html",
"urlToImage": "https://img.washingtonpost.com/rf/image_1484w/2010-2019/Wires/Online/2017-06-13/AP/Images/Trump_99816.jpg-0119c.jpg",
"publishedAt": "2017-06-12T14:33:00Z"
"isRecommend": True
}, {
"author": "Jeremy Diamond, CNN",
"title": "Post-Orlando Trump proposals not yet reality",
"description": "The day after the Pulse nightclub in Orlando became the site of the deadliest terrorist attack since 9/11, Trump produced the latest iteration of his Muslim ban",
"url": "http://www.cnn.com/2017/06/12/politics/trump-pulse-orlando-attack/index.html",
"urlToImage": "http://i2.cdn.cnn.com/cnnnext/dam/assets/170601052736-01-donald-trump-0531-super-tease.jpg",
"publishedAt": "2017-06-12T19:59:19Z"
"isRecommend": True
}, {
"author": "Julia Edwards Ainsley",
"title": "Attorney General Sessions to talk publicly to U.S. Senate panel",
"description": "U.S. Attorney General Jeff Sessions' public testimony before a Senate panel on Tuesday sets up another potentially dramatic hearing on possible ties between President Donald Trump's campaign and Russian meddling in the 2016 presidential race.",
"url": "http://www.reuters.com/article/us-usa-trump-russia-sessions-idUSKBN1920QO",
"urlToImage": "https://s3.reutersmedia.net/resources/r/?m=02&d=20170613&t=2&i=1188721809&w=&fh=545px&fw=&ll=&pl=&sq=&r=LYNXMPED5B11Q",
"publishedAt": "2017-06-13T01:19:00Z"
"isRecommend": False
}, {
"author": "https://www.facebook.com/annafifield",
"title": "Dennis Rodman is on his way to North Korea. Was he sent by Trump?",
"description": "The controversial basketballer hopes to meet Kim Jong Un again in Pyongyang",

"url": "https://www.washingtonpost.com/world/dennis-rodman-is-on-his-way-to-north-korea-was-he-sent-by-trump/2017/06/12/5c9c19cc-4fd9-11e7-a973-3dae94ed3eb7_story.html",
"urlToImage": "https://img.washingtonpost.com/rf/image_1484w/2010-2019/WashingtonPost/2017/06/13/Foreign/Images/2017-06-12T234646Z_1_LYNXMPED5B1UP-ORVPE_RTROPTP_3_PEOPLE-US-USA-NORTHKOREA-RODMAN.jpg",
"publishedAt": "2017-06-12T15:32:00Z"
"isRecommend": False
}, {
"author": "Neil MacFarquhar and Ivan Nechepurenko",
"title": "Across Russia, Protesters Heed Navalny's Anti-Kremlin Rallying Cry",
"description": "Aleksei A. Navalny, a leading Putin critic, was detained outside his home in Moscow as an extraordinary wave of antigovernment protests swept across Russia.",
"url": "https://www.nytimes.com/2017/06/12/world/europe/russia-aleksei-navalny-kremlin-protests.html",
"urlToImage": "https://static01.nyt.com/images/2017/06/12/us/12rusprotest/12rusprotest-facebookJumbo-v2.jpg",
"publishedAt": "2017-06-13T00:31:10Z"
"isRecommend": False
}, {
"author": null,
"title": "Sean Spicer: Donald Trump To Reveal If He Taped James Comey When He's Good And Ready",
"description": "“Does President Trump have audio recordings of his conversations and meetings with the former FBI Director James Comey?” a reporter asked White House Press Secretary Sean Spicer at Monday's briefing, after Labor Secretary Alexander Acosta's briefing show-and-tell on Trump's apprenticeship program. “The",
"url": "https://www.yahoo.com/tv/sean-spicer-donald-trump-reveal-190903083.html",
"urlToImage": "https://s.yimg.com/uu/api/res/1.2/.DDcCT1j1wf65Bp6Jro15A--/aD02Njk7dz0xNjAwO3NtPTE7YXBwaWQ9eXRhY2h5b24-/http://media.zenfs.com/en-us/homerun/deadline.com/f407bcab7818c0ea5abba5192ee415b4",
"publishedAt": "2017-06-12T19:09:03Z"
"isRecommend": False
}, {
"author": "BBC News",
"title": "Theresa May to hold talks with DUP leaders on government deal",

"description": "Theresa May is to meet DUP leader Arlene Foster to try to seal an agreement to enable her party to govern.",
"url": "http://www.bbc.com/news/election-2017-40255958",
"urlToImage": "https://ichef-1.bbci.co.uk/news/1024/cpsprodpb/543D/production/_96456512_fosterandmay.jpg",
"publishedAt": "2017-06-13T02:10:23Z"
"isRecommend": False
}]

Приложение 6. Класс описывающий работу с сетью

```
package alexander.recommendersystemnewsapp.data.remote;

import android.support.annotation.NonNull;

import

com.jakewharton.retrofit2.adapter.rxjava2.RxJava2CallAdapterFactory;

import java.io.IOException;

import java.util.List;

import alexander.recommendersystemnewsapp.data.models.Article;

import

alexander.recommendersystemnewsapp.data.models.UserPreference;

import io.reactivex.Observable;

import okhttp3.CacheControl;

import okhttp3.Interceptor;

import okhttp3.OkHttpClient;

import okhttp3.Request;

import okhttp3.Response;

import retrofit2.Retrofit;

import retrofit2.converter.gson.GsonConverterFactory;

import retrofit2.http.Body;

import retrofit2.http.GET;

import retrofit2.http.POST;

/**

 * Created by aleksandrkarpenko on 01.06.17.

 */
```

```
public class NewsService {
    private static final String URL = "http://recommender-
system.triumphnsk.lclients.ru/";
    public NewsService(){
        OkHttpClient client = new OkHttpClient.Builder()
                .addInterceptor(new Interceptor() {
                    @Override
                    public Response intercept(@NonNull Chain chain) throws
IOException {
                        Request original = chain.request();

                        Request request = original.newBuilder()
                                .cacheControl(new
CacheControl.Builder().noCache().build())
                                .build();
                        return chain.proceed(request);
                    }
                })
                .build();
        Retrofit retrofit = new Retrofit.Builder()
                .baseUrl(URL)
                .addCallAdapterFactory(RxJava2CallAdapterFactory.create())
                .addConverterFactory(GsonConverterFactory.create())
                .client(client)
                .build();
        mNewsServiceApi = retrofit.create(NewsServiceApi.class);
    }

    private NewsServiceApi mNewsServiceApi;
    public NewsServiceApi getNewsServiceApi() {
```

```
        return this.mNewsServiceApi;
    }
    public interface NewsServiceApi {
        @GET("get_news")
        Observable<List<Article>> getNews();
        @POST("set_preferences")
        Observable<String> setPreferences(@Body UserPreference
preference);
    }
}
```

Приложение 7. Настройки проекта

```
import os
BASE_DIR = os.path.dirname(os.path.dirname(__file__))
SECRET_KEY = 'XXXXX'
DEBUG = True
TEMPLATE_DEBUG = True
ALLOWED_HOSTS = ['recommender-system.triumphnsk.lclients.ru']
INSTALLED_APPS = (
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'app',
    'rest_framework',)
MIDDLEWARE_CLASSES =
('django.contrib.sessions.middleware.SessionMiddleware'
,'django.middleware.common.CommonMiddleware','django.middleware.csrf
.CsrfViewMiddleware','django.contrib.auth.middleware.AuthenticationMiddl
eware','django.contrib.messages.middleware.MessageMiddleware','django.
middleware.clickjacking.XFrameOptionsMiddleware',)
ROOT_URLCONF = 'app.urls'
TEMPLATES = [{'BACKEND':
'django.template.backends.django.DjangoTemplates',
'DIRS': [],
'APP_DIRS': True,
'OPTIONS': {'context_processors': [
'django.template.context_processors.debug',
'django.template.context_processors.request',
```

```
'django.contrib.auth.context_processors.auth',
'django.contrib.messages.context_processors.messages',],},},]
WSGI_APPLICATION = 'app.wsgi.application'
DATABASES = {'default': {'ENGINE': 'django.db.backends.mysql',
'NAME': 'XXXXX',
'USER': ' XXXXX ',
'PASSWORD': ' XXXXX ',
'HOST': ' XXXXX ',
'PORT': '3306',}}
DATABASE_ENGINE = 'mysql'
DATABASE_NAME = ' XXXXX '
DATABASE_USER = ' XXXXX '
DATABASE_PASSWORD = ' XXXXX '
DATABASE_HOST = 'mysql7.locum.ru'
DATABASE_PORT = '3306'
LANGUAGE_CODE = 'en-us'
TIME_ZONE = 'UTC'
USE_I18N = True
USE_L10N = True
USE_TZ = True
STATIC_URL = '/static/'
STATIC_ROOT = 'static'
```

Приложение 8. Класс представления

```
def get_news(request):
    if request.method == 'GET':
        r = requests.get('https://newsapi.org/v1/articles?source=the-wall-
street-journal&sortBy=top&apiKey=67b32ada0aba4dfa95f5769982c4cc85')
        json_object = r.json()
        serializer = NewsSerializer(data=json_object)
        if serializer.is_valid():
            news_data = serializer.data
            result = []
            user_preferences = get_object_or_404(UserPreferences,
user__pk=1)
            if not user_preferences:
                for i, article in news_data[0].articles:
          result.append(modify_current_article(article, i < 5))
                    i += 1
                return HttpResponse(json.dumps(result),
content_type='application/json')
            else:
                recommended_articles =
get_recommendations(user_preferences, news_data[0].articles)
                for r_article in recommended_articles:
                    for article in news_data[0].articles:
                        result.append(modify_current_article(article, r_article ==
article))
                return HttpResponse(json.dumps(result),
content_type='application/json')
        else:
            return HttpResponse(json_object)
    else:
```

```
        return HttpResponse('method must be GET')


def modify_current_article(article, is_recommend):
    result = [{
        'title': article.title,
        'description': article.description,
        'url': article.url,
        'urlToImage': article.urlToImage,
        'publishedAt': article.publishedAt,
        'isRecommend': is_recommend
    }]
    return result


def get_recommendations(user_preferences, articles):
    send_articles(articles)
    send_user_preferences(user_preferences)
    recommended = client.send(ItemBasedRecommendation(5,
target_user_id=user_preferences.user.id))
    result = NewsSerializer(data=recommended)
    return result


def send_user_preferences(user_preferences):
    client = RecombeeClient('imastudent',
'PbBaEVxx8ZOj0x3BhGtqfHyi8qQ8rm8rE1JBnSPoCnHwetzO3gjHer96YV
AIa14G')
    preferences = []
    for preference in user_preferences:
```

```
        item = AddDetailView(preference.user, preference,
cascade_create=True)
        preferences.append(item)
    br = Batch(preferences)
    client.send(br)


def send_articles(articles):
    client = RecombeeClient('imastudent',
'PbBaEVxx8ZOj0x3BhGtqfHyi8qQ8rm8rE1JBnSPoCnHwetzO3gjHer96YV
AIa14G')
    for article in articles:
        client.send(AddItem(article))


def set_preferences(request):
    if request.method == 'POST':
        user_id = request.POST.get('user_id')
        article_url = request.POST.get('article_url')
        user = get_object_or_404(User, pk=user_id)
        article = get_object_or_404(Article, url=article_url)
        if user:
            user_preference = UserPreferences(user=user, article=article)
            preferences = [user_preference]
            send_user_preferences(preferences)
            return HttpResponse('everything is ok')
        else:
            return HttpResponse('something going wrong')
    else:
        return HttpResponse('method must be POST')
```

СПИСОК ЛИТЕРАТУРЫ

1. Карпенко А.С., Глухих И.Н. Изучение рекомендательных систем. Разработка экспериментальной рекомендательной системы. – НИР, 2017г.
2. Карпенко А.С., Глухих И.Н. – Научно-исследовательская практика, 2017г.
3. Карпенко А.С., Глухих И.Н. – Отчет по производственной практике Общество с ограниченной ответственностью «КБ-Информ», 2017г.
4. Карпенко А.С., Глухих И.Н. Применение рекомендательных систем в мобильных приложениях. – НИР, 2017г.
5. Cisco Visual Network Index – Report, http://www.cisco.com/c/en/us/solutions/service-provider/visual-networking-index-vni/index.html.
6. Francesco Ricci and Lior Rokach and Bracha Shapira, Introduction to Recommender Systems Handbook, Recommender Systems Handbook, Springer, 2011. - 1-35 с.
7. Facebook, Pandora Lead Rise of Recommendation Engines - TIME". TIME.com. 27 May 2010. Retrieved 1 June 2015.
8. Buettner, Ricardo (2016). "Predicting user behavior in electronic markets based on personality-mining in large online social networks: A personality-based product recommender framework". Electronic Markets: The International Journal on Networked Business. Springer: 1–19. doi:10.1007/s12525-016-0228-z.
9. H. Chen, A. G. Ororbia II, C. L. Giles ExpertSeer: a Keyphrase Based Expert Recommender for Digital Libraries, in arXiv preprint 2015.
10. H. Chen, L. Gou, X. Zhang, C. Giles Collabseer: a search engine for collaboration discovery, in ACM/IEEE Joint Conference on Digital Libraries (JCDL) 2011.
11. Francesco Ricci and Lior Rokach and Bracha Shapira, Introduction to Recommender Systems Handbook, Recommender Systems Handbook, Springer, 2011. - 1-35 с.

12. Hosein Jafarkarimi; A.T.H. Sim and R. Saadatdoost A Naïve Recommendation Model for Large Databases, International Journal of Information and Education Technology, June 2012

13. The DCI Architecture: A New Vision of Object-Oriented Programming – Trygve Reenskaug and James Coplien – March 20, 2009.

14. "Model View Presenter." Jean-Paul Boodhoo, MSDN Magazine, August 2006

15. Martin Fowler (19 July 2004). "The Presentation Model Design Pattern"

16. Android [Электронный ресурс]. Режим доступа: http://ru-wiki.org/wiki/Android - Заглавие с экрана. – (Дата обращения: 01.06.2017).

17. iOS [Электронный ресурс]. Режим доступа: http://ru-wiki.org/wiki/ios - Заглавие с экрана. – (Дата обращения: 01.06.2017).

18. Предсказатель_переходов [Электронный ресурс]. Режим доступа: http:// ru-wiki.org/wiki/Предсказатель_переходов - Заглавие с экрана. – (Дата обращения: 01.06.2017).