Федеральное государственное образовательное бюджетное

учреждение высшего образования

**«Финансовый университет**

**при Правительстве Российской Федерации»**

**(Финансовый университет)**

Департамент анализа данных и машинного обучения

Пояснительная записка к курсовой работе по дисциплине

«Современные технологии программирования»

на тему:

**«Разработка информационной системы для кинотеатра с использованием Java, Spring и технологий мобильной разработки»**

Выполнил:

Студент группы ПИ19-4

Козловский Алексей Дмитриевич

Научный руководитель:

профессор, доктор эконом. наук

Демин Игорь Святославович

**Москва, 2021**

ОГЛАВЛЕНИЕ

I. ВВЕДЕНИЕ

В крупных городах России и мира давно уже не встретишь кинотеатра с билетной кассой. Всё потому, что большинство сетей кинотеатров и компаний по продаже билетов не только на киносеансы, но и на другие мероприятия автоматизировали свои задачи благодаря информационно-справочным системам.

К примеру, сети кинотеатров «Карро» и «Кино Окко» давно используют в своей работе информационно-справочные стенды для удобной покупки билетов (см. прил. 1), а также сейчас пользуются популярностью такие сервисы как «Яндекс Афиша», «Афиша.ру», «Кинопоиск Афиша» и другие, которые являются информационным сервисом-посредником между кинотеатром и пользователем и позволяют последним с легкостью покупать билеты на сеансы с любого устройства или браузера.

Ставится вопрос: как работают такте сервисы изнутри? Какие решения используются для их реализации? Какова актуальность внедрения таких информационно-справочных систем в кинотеатры?

Цель данной курсовой работы – проанализировать предметную область автоматизации кинотеатров, опираясь на проведенный анализ составить информационную модель задачи автоматизации и продемонстрировать возможный вариант её решения на примере серверного приложения на языке Java и с использованием фреймворка Spring, а также клиентского мобильного приложения под платформу Android OS, написанного на языке Java и с использованием современных технологий программирования.

Выполнение мною курсовой работы поспособствует приобретению и усовершенствованию навыков анализа предметной области, построения информационной модели задачи, проектирования реляционных баз данных, а также программирования на языке Java и разработки клиент-серверных приложений с технологией REST.

II. ПОСТАНОВКА ЗАДАЧИ

В рамках курсовой работы поставлена задача продемонстрировать возможное клиент-серверное решение по автоматизации работы кинотеатров с использованием библиотеки Spring Boot для серверной части и также Android SDK и библиотек RxJava2, Retrofit2, Picasso для клиента.

Разработанное серверное приложение должно обрабатывать запросы пользователей (клиентов), хранить и обрабатывать данные с помощью СУБД (системой управления базами данных), в данном случае MySQL.

Разработанное клиентское приложение должно иметь интуитивный и эргономичный дизайн экранов для удобного взаимодействия с пользователем, четкую и продуманную логику взаимодействия с серверной частью, а также быть оптимальным по затратности ресурсов и времени исполнения кода.

Разработанная система должна удовлетворять всем установленным требованиям, указанным в техническом задании, а именно:

1. В курсовом проекте должна быть разработана информационная модель предметной области, представленная в виде пользовательских классов и таблиц БД. Взаимодействие с БД должно быть реализовано при помощи ORM.
2. Должно быть разработано несколько форм пользовательского интерфейса для клиента.
3. Разработчик самостоятельно определяет интерфейс программы и ее функциональность, однако для получения максимальной оценки приложение в обязательном порядке независимо от предметной области, указанной в задании, должно выполнять следующие операции:

* Отображать в таблице данные предметной области.
* Для информационной модели, основанной на БД, таблицы должны быть предварительно заполнены записями.
* Реализовать добавление в БД нового объекта, удаление объекта, редактирование объекта.
* Реализовать фильтрацию записей БД, удовлетворяющих введенному пользователем сложному критерию.
* Реализовать сортировку записей.
* Обновлять изменения источника данных в базе данных.
* После сохранения данных при запуске программы загрузить данные из БД.
* Создать пункт меню «Об авторе».

1. Программа не должна завершаться аварийно: сообщения о  
   некорректном вводе данных, противоречивых или недопустимых значениях данных, при отсутствии данных по функциональному запросу пользователя и других нештатных ситуациях отображать в окнах сообщений.
2. Программа должна иметь содержательные комментарии, которые  
   могут генерировать автоматически составляемую документацию при помощи инструмента Javadoc.

III. ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

Предметной областью является автоматизация продажи билетов в кинотеатре. И хотя клиентские части у многих сервисов разные (чаще всего это веб-сайты), серверная часть зачастую схожа. Она хранит все необходимые данные, такие как: фильмы в прокате, сеансы на них, чеки клиентов, их данные для доступа в аккаунт в хешированном виде, их персональную информацию, избранные фильмы и меню кинобара.

В свою очередь клиент делает HTTP-запросы к серверу в синхронном или асинхронном режиме. Полученные с сервера данные он отображает на веб-странице или экране пользователя.

В этой курсовой работе в качестве клиента выступает мобильное приложение, так как смартфоны наиболее удобные и популярные устройства (75.5% населения планеты имеют мобильные устройства, из них – 62.2% предпочитают их настольным компьютерам)

IV. АКТУАЛЬНОСТЬ АВТОМАТИЗАЦИИ

Актуальность автоматизации обслуживания клиентов в кинотеатрах обуславливается потребностью тех к быстрому и удобному получению предоставляемых услуг, таких как билет на определенный сеанс или заказ из кинобара, избегая очереди на кассе, и оплатить заказ посредством удобного способа оплаты, например банковской картой или платежным сервисом (как пример с помощью Google Play, Apple Pay, Qiwi и прочих).

Также автоматизация обслуживания поспособствует управляющему персоналу кинотеатра быстро и легко обмениваться информацией с пользователем, тем самым сэкономив его время и свои деньги на оплату труда билетных кассиров; и помимо всего прочего иметь наглядную и точную отчетность за конкретный период, статистику посещаемости и проч.

V. АЛГОРИТМИЧЕСКИЕ РЕШЕНИЯ

Сервер

Серверное приложение курсового проекта реализовано с помощью сборщика проектов Gradle и библиотеки Spring Boot. Оно обрабатывает запросы клиента параллельно по следующему алгоритму.

* Служебный класс DispatcherServlet после получения HTTP-запроса от клиента обращается к интерфейсу HandlerMapping, который сравнив URL-адрес запроса с существующими маппингами контроллеров, определяет, какой из них ответственный за этот запрос, после чего, делегирует запрос ему.
* Контроллер подписан на события HandlerMapping и ждёт, пока ему делегируют запрос, после чего он обращается к нужному сервису за получением данных и возвращает ответ.
* Сервис получает запрос от контроллера и обращаются за данными к нужному репозиторию. После чего возвращает контроллеру ответ.
* Репозиторий получает запрос от сервиса и обращается за данными уже к сущностям, то есть напрямую к базе данных. После получения данных он возвращает ответ сервису.

Более наглядно алгоритм работы сервера продемонстрирован на рисунке 3.

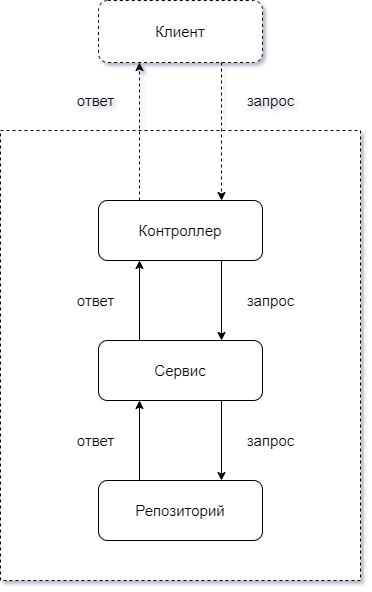


Рис. 3 Иерархическая архитектура серверного приложения

Хранение данных на сервере реализовано при помощи СУБД MySQL. Каждая сущность представляется в виде реляционной таблицы и имеет обязательный искусственный **уникальный идентификатор**, с помощью которого можно однозначно идентифицировать объект сущности.

Также сущности, в зависимости от своего типа, имеют архитектуру, необходимую для хранения определенной информации.

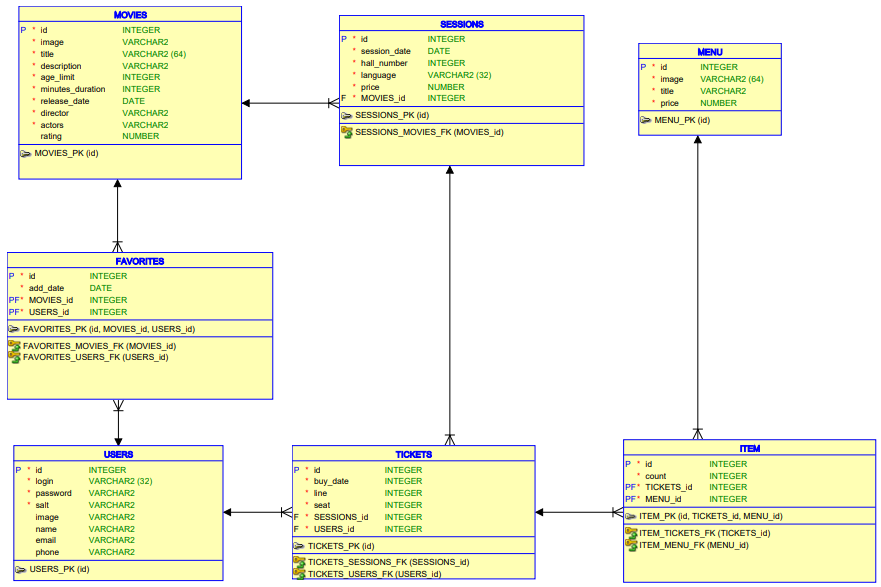


Рис. 1 Схема разработанных сущностей.

К примеру, сущность «Фильмы» содержит информацию о фильмах, которые идут или когда-либо шли в прокате:

* название фильма – столбец title
* описание фильма – столбец description
* ссылка на постер фильма – столбец image
* возрастное ограничение – столбец age\_limit
* продолжительность (в мин.) – столбец minutes\_duration
* дата релиза – столбец release\_date
* режиссер(ы) – столбец director
* актеры – столбец actors
* рейтинг фильма – столбец rating

Сущность «Сеансы» имеет связь «многие к одному» с сущностью «Фильмы», а также хранит информацию о сеансах, такую как:

* дата сеанса – столбец session\_date
* название зала сеанса – столбец hall
* язык сеанса – столбец language
* цена билета на сеанс – столбец price
* внешний ключ фильма – столбец movie\_id

Сущность «Пользователи» хранит данные пользователя, такие как:

* псевдоним – столбец login
* пароль в захэшированном виде – столбец password
* соль (комбинация из 12 случайных символов для невозможности декодирования пароля пользователя) – столбец salt
* изображение профиля пользователя – столбец image
* полное имя пользователя – столбец name
* эл. почта пользователя – столбец email
* номер телефона пользователя – столбец phone

Сущность «Избранное» является решением связи «многие ко многим» для сущностей «Фильмы» и «Пользователи». Она хранит информацию об избранных фильмах пользователя, такую как:

* дата добавления в избранное – столбец add\_date
* внешний ключ фильма – столбец movie\_id
* внешний ключ пользователя – столбец user\_id

Сущность «Меню» хранит информацию о товарах в кинобаре:

* изображения товара – столбец image
* название товара – столбец title
* стоимость товара – столбец price

Сущность «Билеты» хранит данные о заказах пользователя:

* дата приобретения билета – столбец buy\_date
* выбранный ряд – столбец line
* выбранное место – столбец seat
* внешний ключ сеанса – столбец session\_id
* внешний ключ пользователя – столбец user\_id

Сущность «Элементы» является решением связи «многие ко многим» для сущностей «Билеты» и «Меню» и хранит информацию о том, какие товары из кинобара были добавлены к приобретенному пользователем билету:

* количество товаров – столбец count
* внешний ключ билета – столбец ticket\_id
* внешний ключ товара – столбец menu\_id

Также на серверной стороне реализован алгоритм хеширования паролей при помощи алгоритма MD5 и метода «соления» паролей: к чистому паролю пользователя добавляется случайная комбинация из 20 байтов (соль) для предотвращения любой возможности декодирования пароля.

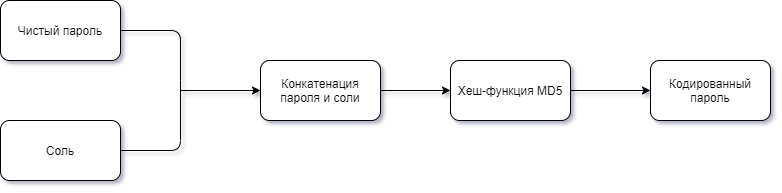


Рис 1. Алгоритм реализованного кодирования паролей пользователя

Клиентское приложение курсового проекта реализовано с помощью парадигмы асинхронного программирования и библиотеки RxJava2, что способствует более удобной и правильной обработке HTTP-запросов. В свою очередь, обработка HTTP-запросов реализована с помощью библиотеки Retrofit2, которая максимально удобна для разработчика тем, что позволяет хранить всю логику выполнения запросов в интерфейсе.

Приложение написано с использованием паттерна проектирования MVP (Model–View–Presenter. Суть заключается в том, чтобы создать абстракцию представления (того, что видит пользователь). Для этого необходимо выделить интерфейс представления с определенным набором свойств и методов. Presenter, в свою очередь, получает ссылку на реализацию интерфейса, подписывается на события представления и по запросу изменяет модель. Вся работа с моделью выполняется в фоновых потоках, чтобы не нагружать главный поток пользовательского интерфейса (Main Thread).

Для работы с пользовательскими изображениями на стороне клиента используется библиотека Picasso, позволяющая загружать изображения с любого интернет-ресурса быстро и асинхронно.

VI. ОПИСАНИЕ ИНТЕРФЕЙСА ПРОГРАММЫ

В серверном приложении интерфейс отсутствует, так как представление данных и обработка событий пользователя не входит в его основные задачи.

Пользовательский интерфейс клиентского приложения эргономично спроектирован и представлен в нескольких основных типах, а именно:

1. Отдельные экраны (активности)

* Заглушечный экран
* Стартовый экран
* Главный экран с меню
* Экран регистрации
* Экран авторизации
* Экран подробной информации о фильме
* Экран настроек
* Экран выбора мест на сеанс
* Экран выбор дополнительных товаров к заказу

1. Фрагменты

* Фрагмент фильмов, идущих в кинотеатре
* Фрагмент истории билетов (чеков)
* Фрагмент пользовательского профиля

1. Диалоговые окна с сообщениями об ошибках

Заглушечный экран

Заглушечный экран отображается всегда при запуске приложения. За это отвечает специальный фильтр в манифесте приложения:

<intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
 <category android:name="android.intent.category.LAUNCHER" />  
</intent-filter>

На заглушечном экране отображен логотип приложения без каких-либо дополнительных элементов интерфейса. В то время, как пользователь видит заглушечный экран, происходит загрузка приложения (его ресурсов и классов) системой Android на протяжении примерно одной секунды. Это довольно распространенная практика разработки приложений, чтобы представить пользователю приятную глазу картинку вместо белого экрана (который система рисует по умолчанию во время загрузки ресурсов).

После загрузки ресурсов в классе заглушечного экрана проверяется факт авторизации пользователя. За это отвечает Boolean переменная «isAuthorized», которая хранится в локальном хранилище (на устройстве). Если пользователь не авторизован, запускается стартовый экран.



Рис. 1 Заглушечный экран

Стартовый экран

Стартовый экран отображается после заглушечного, если пользователь не авторизован. На нём запрашивается у пользователя зарегистрироваться или авторизоваться, без чего главный экран приложения не может быть открыт не при каких условиях. Также на стартовом экране снизу расположены кликабельные ссылки на страницы разработчика.

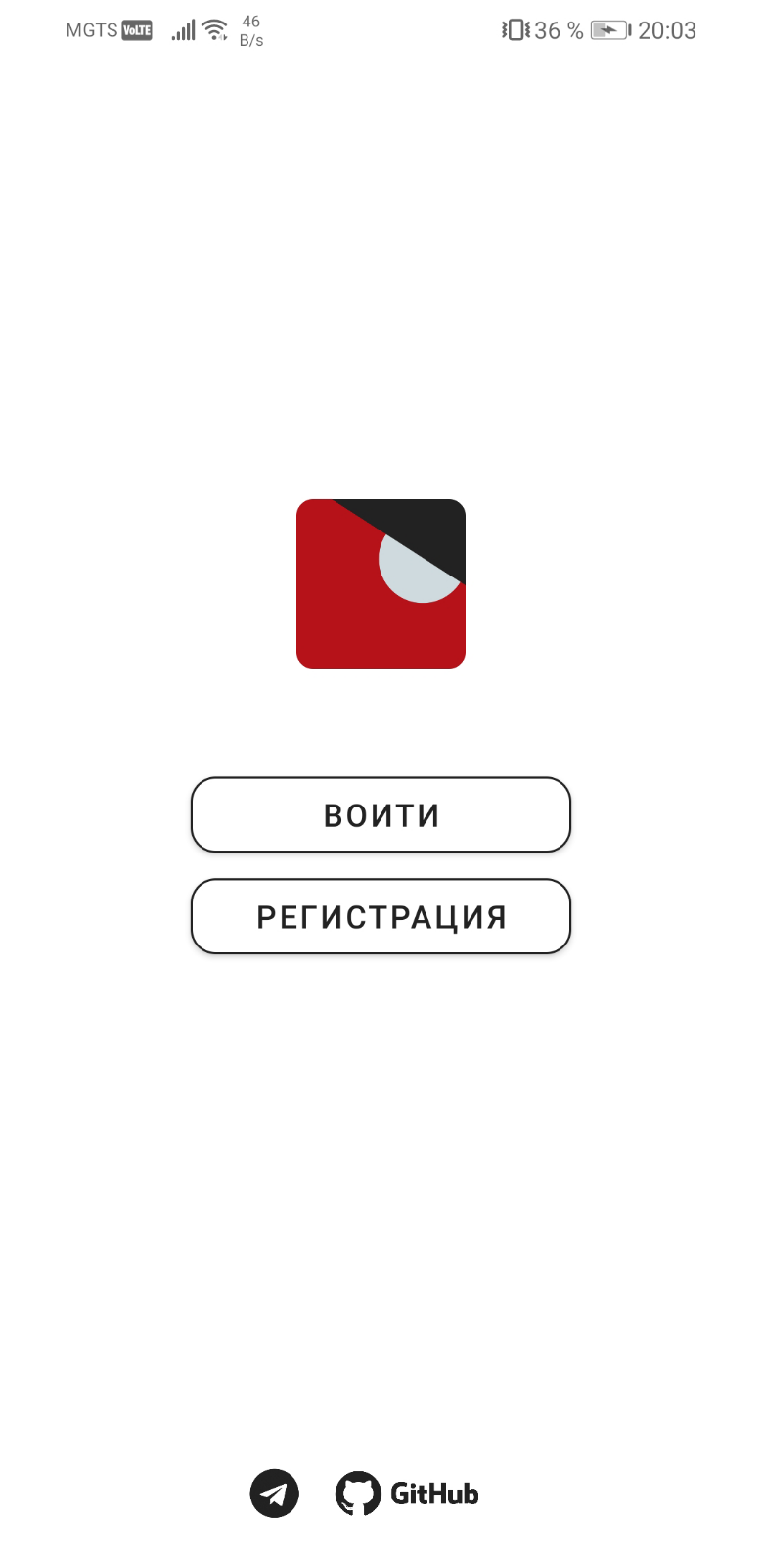


Рис. 3 Стартовый экран приложения

Главный экран с меню

Главный экран запускается после заглушечного, если пользователь авторизован в приложении или после экранов регистрации и авторизации, если он успешно зарегистрировался или авторизовался.

Главный экран является неким контейнером для фрагментов с фильмами, историей заказов и профилем пользователя, между которыми можно переключаться посредством нижнего меню. Также из главного экрана можно попасть в экран настроек, а слева сверху отображается название выбранного раздела или же фрагмента.

Фрагменты на главном экране сохраняют свое состояние, то есть не перерисовываются заново при выборе в меню, однако пользователь может обновить данные с сервера при помощи рефрешера.

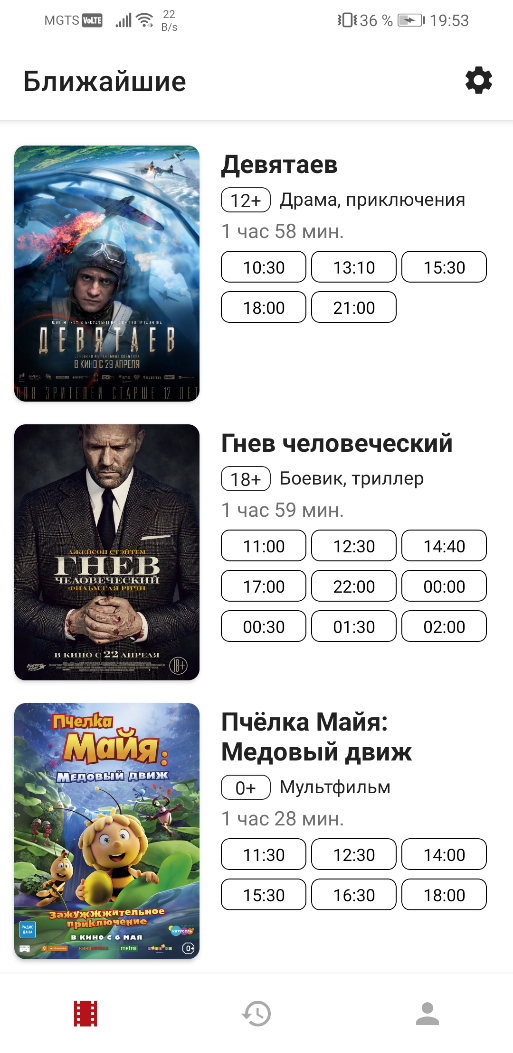


Рис. 3 Главный экран

Экран регистрации и авторизации

На экране регистрации у пользователя запрашивается электронная почта, псевдоним и пароль для создания нового аккаунта. Кнопка «Вперед» изначально выключена и недоступна для нажатия. Как только во всех полях ввода появляется хотя бы один символ (не считая пробела), она становится включенной и доступной для нажатия.

При нажатии на кнопку «Вперед» проверяется корректность ввода.

Требования к регистрационным данным:

* Эл. почта должна соответствовать определенном паттерну.
* Псевдоним не должен начинаться с цифры и может содержать только буквы латинского алфавита в любом регистре.
* Пароль должен содержать минимум 8 символов, включая хотя бы один символ верхнего регистра и один спец. символ (#, ?, % или &).

В случае некорректного ввода поле становится красным, что говорит об ошибке, и регистрация не продолжается. Также пользователь может выбрать изображение профиля, для чего требуется дать приложению разрешение на доступ к хранилищу и камере. В случае если он не выберет изображение, оно будет сгенерировано по шаблону фон + первый символ псевдонима.

Если же ввод корректен, появляется индикатор прогресса на время отправки клиентом данных на сервер, после чего запускается главный экран.

Экран авторизации служит для авторизации пользователя в системе. Экран авторизации запрашивает ввести логин или электронную почту и пароль. Пароль хешируется с помощью алгоритма MD5 и отправляется на сервер для сравнения с уже имеющимся паролем, сохраненным в базе данных. В случае совпадения паролей, возвращается сущность пользователя с указанным логином или эл. почтой, в другом случае – пользователь может увидеть у себя на экране ошибку о некорректности введенных данных.

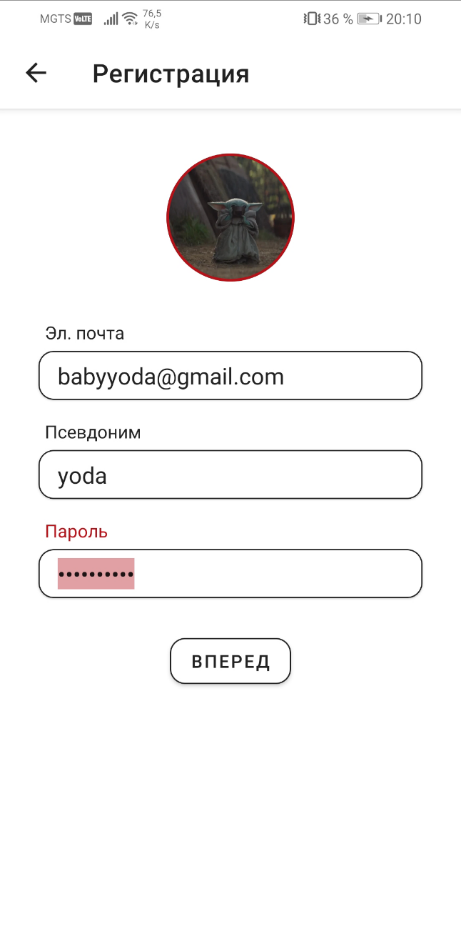
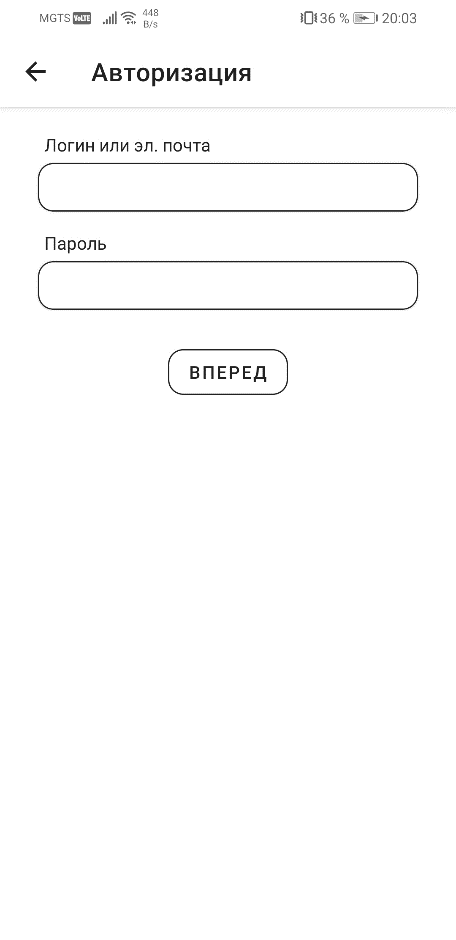
 

Рис. 5 Экран регистрации (слева)

Рис. 6 Экран авторизации (справа)

Экран подробной информации о фильме

Экран отображает подробную информацию о фильме, которая может быть интересна пользователю: возрастной рейтинг, категории, длительность, описание, режиссеров, актерский состав и ближайшие сеансы.

При нажатии на «звездочку» фильм будет добавлен в «Избранное».

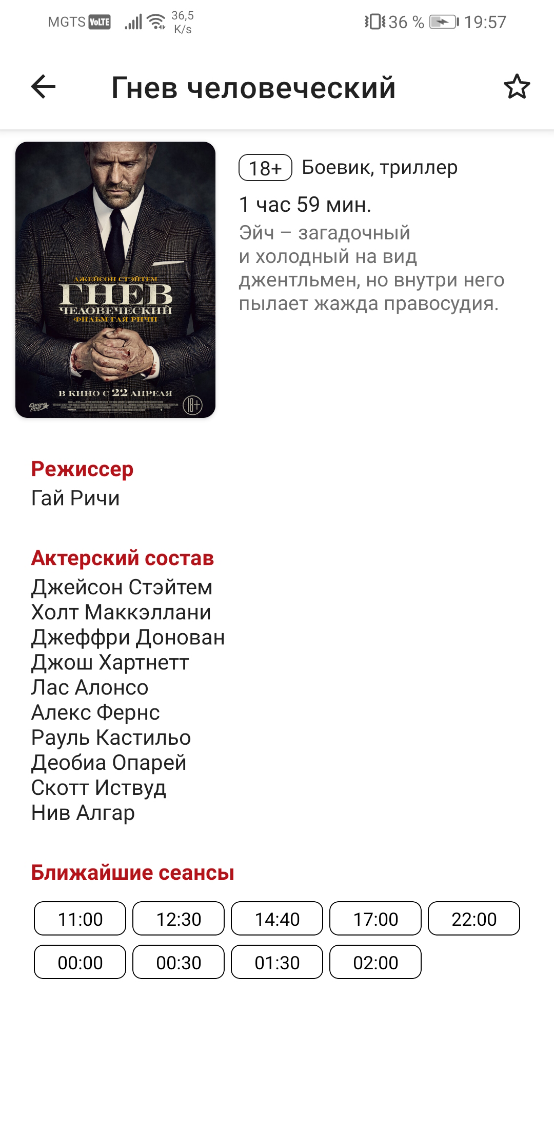


Рис. 7 Экран подробной информации о фильме

Экран настроек

Экран настроек позволяет пользователю изменить данные, указанные пре регистрации и указать дополнительные данные. При нажатии на элемент вызывается диалоговое окно (рисунок 9), в котором при нажатии на «Ок» выбранный параметр будет изменен. При нажатии на «Отмена» диалоговое окно будет закрыто без изменений параметра.

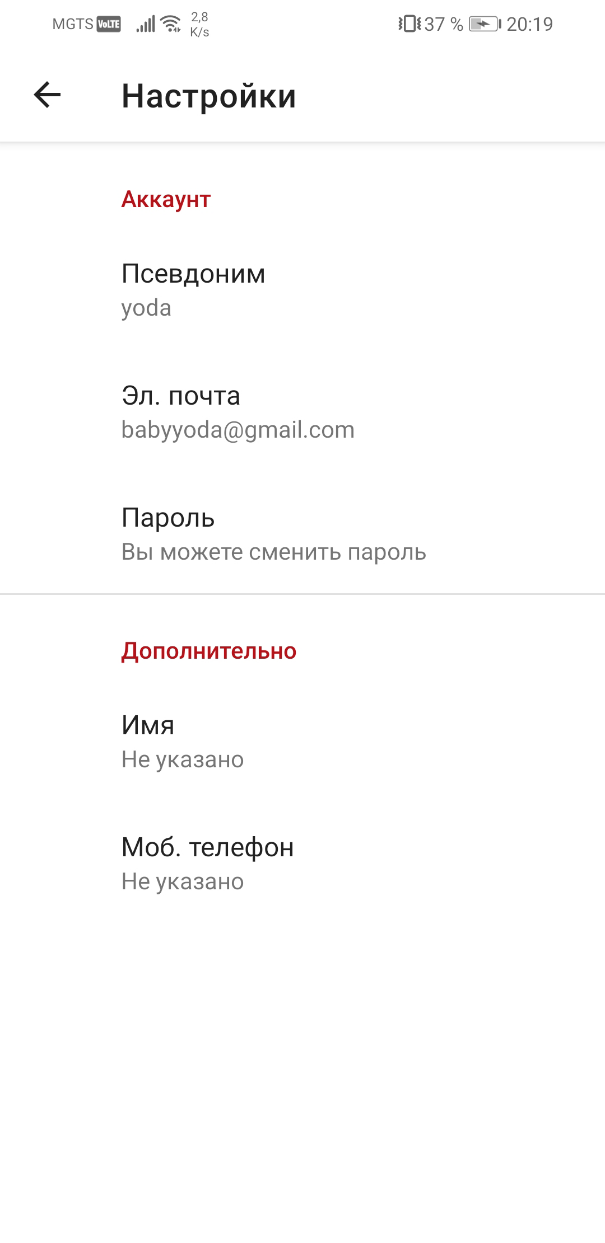
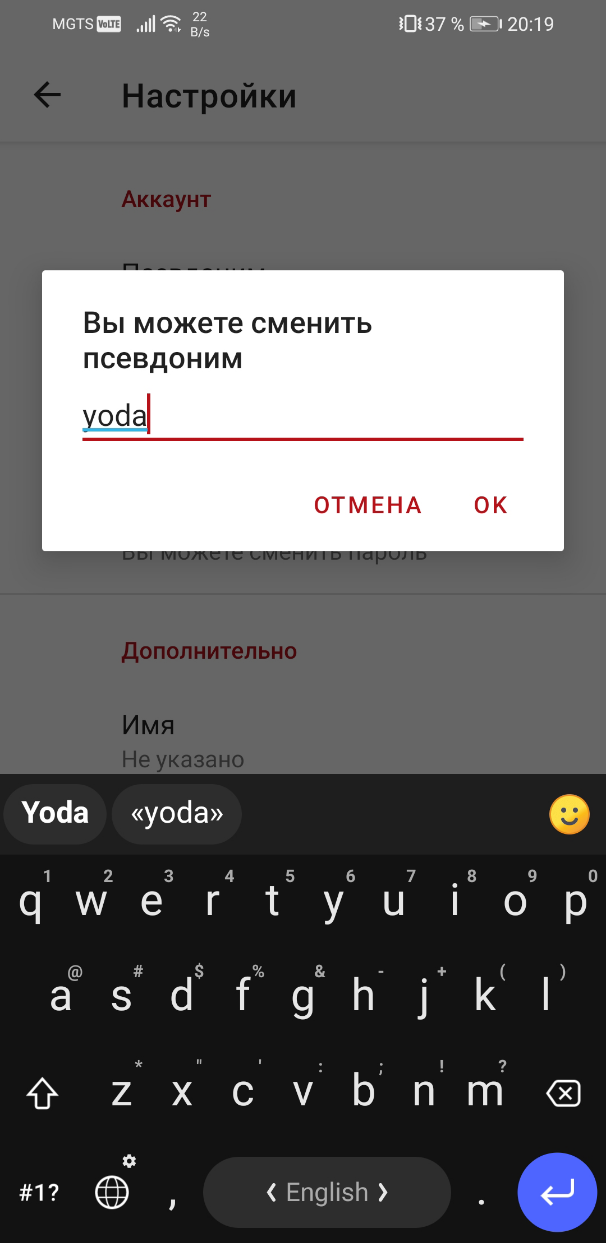
 

Рис. 8 Экран настроек (слева)

Рис. 9 Экран настроек после выбора элемента меню «Псевдоним» (справа)

VII. СОСТАВ ПРИЛОЖЕНИЯ

Приложение состоит из приложения (сервера), которое занимается хранением и обработкой информации из базы данных, базы данных (БД) и приложения, которое занимается отображением информации, полученной от сервера посредством запрос

а (клиента).

VIII. НАЗНАЧЕНИЕ И СОСТАВ КЛАССОВ ПРОГРАММЫ

IX. ЗАКЛЮЧЕНИЕ

X. СПИСОК ЛИТЕРАТУРЫ

1. Уоллс К. Spring в действии. – М.: ДМК Пресс, 2013. – 752 с.
2. Лонг Джош, Бастани Кеннет Java в облаке. Spring Boot, Spring Cloud, Cloud Foundry. – СПБ.: Питер, 2019. – 624 с.
3. Филлипс Билл, Стюарт Крис, Марсикано Кристин, Гарднер Брайан Android. Программирование для профессионалов. 4-е издание. — СПб.: Питер, 2021. — 704 с.
4. Кузнецов М. В., Симдянов И. В. Самоучитель MySQL 5. — СПб.: БХВ-Петербург, 2006. — 560 с.
5. Олег Докука, Игорь Лозинский Практика реактивного программирования в Spring 5. – М.: ДМК Пресс, 2019. – 508 с.
6. Козмина, Юлиана, Харроп, Роб, Шефер, Крис, Хо, Кларенс. Spring 5 для профессионалов.: Пер. с англ. - СПб.: ООО "Диалектика", 2019. - 1120 с.
7. Эккель Б. Философия Java. 4-е полное изд. – СПБ.: Питер, 2015. – 1168 с.
8. Мартин Роберт К. Чистый код. Создание анализ и рефакторинг. - СПб: Питер, 2019. - 464 с.
9. Прохоренок Н.А. JavaFX. - СПб: БХВ-Петербург, 2020. - 768 с.
10. Шилдт Г. Java. Полное руководство. - Киев: Диалектика, 2018. - 1488 с.

XI. ПРИЛОЖЕНИЕ

1. Информационные терминалы сети кинотеатров «КАРО Фильм»

