МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное

учреждение высшего образования

«Ижевский государственный технический университет

имени М.Т. Калашникова»

(ФГБОУ ВО «ИжГТУ имени М.Т. Калашникова»)

Факультет «Информатика и вычислительная техника»

Кафедра «Программное обеспечение»

Направление подготовки 09.03.04 Программная инженерия

ОТЧЕТ ПО ПРАКТИКЕ

|  |  |  |
| --- | --- | --- |
|  | | |
|  | *(наименование типа практики)* |  |
|  |  |  |
|  | | |
|  | *(полное наименование профильной организации)* |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Выполнил обучающийся |  | / |  | / |
|  | *(подпись)* |  | *(ФИО, курс, номер группы)* |  |

Дата сдачи отчета: «    »                     202  г.

Дата аттестации «    »                       202  г.

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Руководитель практики от  ИЖГТУ имени М.Т. Калашникова |  | / |  | / |
|  | (*подпись*) |  | *(И.О. Фамилия, должность, ученая степень)* |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Заведующий кафедрой |  | / |  | / |
|  | *(подпись)* |  | *(И.О. Фамилия, должность, ученая степень)* |  |

# РАЗРАБОТКА ЗАДАЧИ

## Описание постановки задачи

### Характеристика задачи

Для решения поставленных задач необходимо разработать:

1. Модуль для управления личным кабинетом
2. Модуль для выполнения задач администратора сервиса
3. Модуль для управления игровой информацией
4. Модуль для интеграции с платформами крупных игровых дистрибьюторов через публичное API

Для ведения библиотеки игр, написания отзывов и оценивания игр на сервисе пользователю необходимо иметь учетную запись. Для организации всех процессов, связанных с взаимодействием пользователя со своим профилем и элементами системы, требующими наличие учетной записи, необходим инструмент, который обеспечит регистрацию, авторизацию и управление профилем пользователя.

Для игр с платформ, не предоставляющих публичное API, и игр, выпускаемых вне платформ, необходимо реализовать возможность ручного добавления игр в базу. Также для администратора необходимо создать функционал для ручного запуска и настройки процессов интеграции с внешними системами. Для предотвращения добавления запрещенного контента в отзывы к играми, необходим инструмент для их модерирования.

Необходимо разработать функционал для всех действий пользователя на сервисе, связанных с взаимодействием с данными об играх, а именно:

1. Поиск игр по критериям
2. Написание отзывов
3. Оценивание игр
4. Открытие страниц игр
5. Добавление, удаление, изменение статуса игры в библиотеке пользователя

Функционал должен представлять собой прослойку между действиями пользователя и обращениями к базе данных игровой информации, роль которого заключается в обработке действия пользователя, получения параметров запроса и валидации этих параметров, для дальнейшего преобразования полученной от пользователя информации и создания подходящего запроса к БД.

Модуль интеграции

### Входная информация

Входными данными для всех модулей являются таблицы баз данных с игровой информацией и учетными записями пользователей. Ниже приведен список таблиц и их состав столбцов.

Таблица Games:

1. Name – наименование
2. Description – описание
3. Logo – логотип
4. Genre – жанр игры
5. AgeRating – возрастное ограничение
6. Publisher – издатель (платформа)
7. Developer – разработчик
8. ReleaseDate – дата выхода
9. Rating – средний пользовательский рейтинг игры на сервисе

Таблица Catalogs:

1. Name – наименование
2. Description – описание

Таблица GameSummaries:

1. GameName – наименование игры
2. Rate – Оценка пользователя
3. GameId – идентификатор игры
4. UserId – идентификатор пользователя

Таблица Reviews:

1. GameId - идентификатор игры
2. UserId – идентификатор пользователя
3. Text – текст отзыва
4. Date – дата отзыва
5. IsApproved – признак допуска отзыва до публикации администратором

Таблица AspNetRoles:

1. RoleId – идентификатор роли пользователя
2. Name – наименоваие

Таблица AspNetUsers:

1. UserId – идентификатор пользователя
2. UserName – никнейм пользователя
3. PasswordHash – хэш пароля
4. Login – логин пользователя
5. FullName – имя пользователя
6. BirthDate – дата рождения пользователя

Также входными данными являются шаблоны html-страниц, которые будут возвращаться пользователю с подставленными в них данными.

### Выходная информация

Выходными данными каждой операции являются html-страницы для пользователя и код выполнения http-запроса для системы. Также, в случае экспорта списков пользователя выходной информацией будет являться файл в формате JSON, составленный по определенному шаблону, хранящий в себе информацию о всех играх в библиотеке пользователя.

## Описание модуля «Личный кабинет»

### Назначение и характеристика

Модуль предназначен для работы с учетными записями пользователей: получения информации о профиле пользователя (никнейм, фото, дата рождения, имя и т.д.). Также модуль занимается обработкой библиотек игр пользователей: добавление, удаление и изменения статуса игр в библиотеке. Библиотека представляет собой список, разделенный по категориям.

### Используемые данные

Алгоритм использует данные учетной записи, а также роль пользователя.

Также для получения библиотеки пользователя используется таблица базы данных GameSummaries.

### Результаты решения

В случае регистрации результатом является создание новой учетной записи в базе данных.

В случае авторизации результатом является установление контекста текущего пользователя.

В случае получения профиля результатом является html-страница профиля пользователя и его библиотека игр.

Также в любом из действий результатом может является сообщение об ошибке в случае некорректных введенных учетных данных.

### Алгоритм решения

Регистрация:

1. Получение логина и пароля для новой учетной записи
2. Создание новой учетной записи с ролью «Пользователь»
3. Авторизация в новую учетную запись

Авторизация:

1. Получение логина и пароля для новой учетной записи
2. Попытка найти пользователя с такими же логином и паролем
3. В случае успеха - установка контекста текущего пользователя, получение информации о профиле и библиотеке игр пользователя
4. В случае неудачи - сообщение об ошибке

Схема алгоритма регистрации и авторизации представлена на рисунке 2.1.

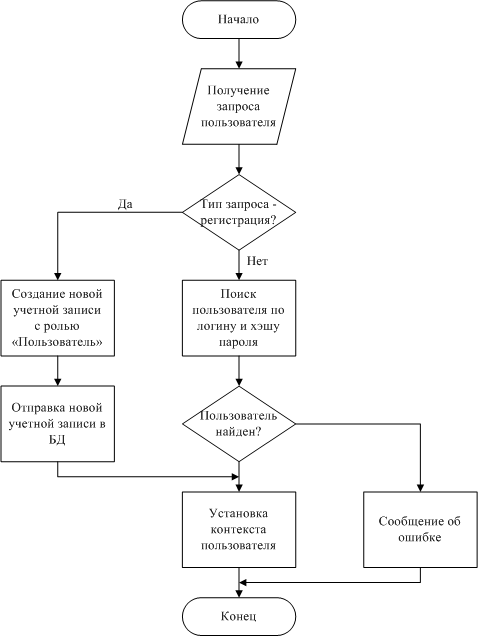


Рис 2.1. Схема алгоритма авторизации и регистрации

## Описание модуля администрирования

### Назначение и характеристика

Модуль предназначен для ручного взаимодействия с модулем управления данными об играх, запуска и настройки автоматических процессов заполнения базы, и модерирования отзывов пользователей.

### Используемые данные

Добавление игры в базу – структурированная информация об игре (название, описание, дата выхода и т.д.) в виде формы на специальной html-странице.

Запуск и настройка процессов интеграции – строка в формате cron.

Модерирование отзыва – текст отзыва.

### Результат решения

Добавление игры в базу – новая строка в таблице Games в БД игровых данных.

Запуск и настройка процессов интеграции – выполнение внеочередного сеанса интеграции или измененное расписание запуска.

Модерирование отзыва – появление отзыва в общем доступе или удаление из БД.

### Алгоритм решения

Добавление игры в базу:

1. Заполнение формы на html-странице
2. Валидация полученных данных
3. Добавление игры в таблицу БД

Запуск процесса интеграции:

1. Выбор необходимого процесса
2. Запуск процесса

Настройка процесса интеграции:

1. Выбор необходимого процесса
2. Изменение конфигурационных данных процесса
3. Сохранение изменений

Модерирование отзыва:

1. Открытие текста отзыва
2. Принятие решения о публикации или удалении отзыва

Алгоритм представлен на рисунке 2.2.

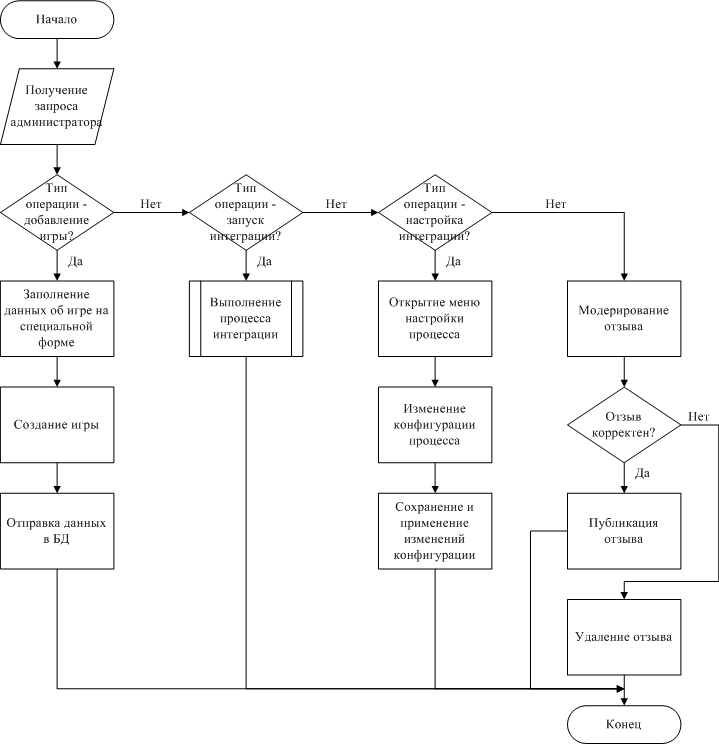


Рис. 2.2. Схема алгоритма работы модуля администрирования

## Описание модуля управления данными об играх

### Назначение и характеристика

Модуль управления данными об играх предназначен для взаимодействия пользователя с базой игр – поиск игр по критериям, оценивание игр и написание отзывов к ним.

Основной задачей модуля является создание подготовленных запросов к базе данных игровой информации на основании полученных пользовательских запросов.

Также важной задачей является обеспечении безопасности баз данных – пользователь не должен взаимодействовать с БД напрямую. Данный модуль выступает промежуточным звеном, которые должно принять запрос от пользователя, обработать его, и вернуть пользователю доступную запрошенную информацию из базы.

### Используемые данные

Основная используемая информация в модуле – это get-параметры http-запроса пользователя и таблицы базы данных.

Список используемых таблиц:

1. Games – таблица с основной информацией о каждой игре
2. Catalogs – таблица с видами возможных каталогов (статусов), по которым игры группируются в библиотеке
3. GameSummaries – таблица, хранящая информацию каждом добавлении игры в библиотеку всеми пользователями
4. GameComments – таблица с отзывами об играх
5. AspNetUsers – таблица с учетными данными пользователей

Взаимодействие с таблицами происходит не прямыми запросами к БД, а через интерфейс библиотек Microsoft.EntityFrameworkCore 3.1.8 (для работы с игровой информацией) и Microsoft.AspNetCore.Identity.EntityFrameworkCore 3.1.8 (для работы с контекстом пользователя).

Для возврата результатов в виде html-страниц используется технология Microsoft.AspNetCore.Mvc 3.1.0.

Варианты действий, обрабатываемых модулем, и их параметры представлены в таблице 2.1.

Таблица 2.1. Варианты входной информации

|  |  |
| --- | --- |
| Действие | Параметры действия |
| Поиск игр | Критерии поиска (название, жанр, дата выхода и т.д.) |
| Открытие страницы игры | Идентификатор игры |
| Добавление игр в библиотеку | Идентификатор игры, идентификатор каталога в библиотеке |
| Изменение статуса игры в библиотеке | Идентификатор игры, идентификатор каталога в библиотеке |
| Удаление игры из библиотеки пользователя | Идентификатор строки в библиотеке |
| Оценивание игры пользователем | Идентификатор игры, значение оценки (целое число) |
| Написание отзыва к игре | Идентификатор игры, текст отзыва |

Помимо указанных параметров, для каждого действия входной информацией является наличие учетной записи у текущего пользователя, т.к. для авторизованного и неавторизованного пользователя действие может возвращать разный результат.

### Результаты решения

Результатом обработки модулем действий пользователя является информация из базы данных в виде html-страницы в интернет-браузере, данные в которой формируются по технологии представлений Razor, используемой в ASP .NET Core.

Варианты выходной информации действий, обрабатываемых модулем представлены в таблице 2.2.

Таблица 2.2. Варианты выходной информации

|  |  |
| --- | --- |
| Действие | Выходная информация |
| Поиск игр | Если найдена всего 1 игра, то html-страница с информацией об игре.  Если найдено больше одной игры, то html-страница со списком ссылок на игры, удовлетворяющих критериям запроса |
| Открытие страницы игры | Html-страница игры |
| Добавление игр в библиотеку | Признак успешности выполненного запроса и обновленная html-страница игры |
| Изменение статуса игры в библиотеке | Признак успешности выполненного запроса и обновленная html-страница игры |
| Удаление игры из библиотеки пользователя | Признак успешности выполненного запроса |
| Оценивание игры пользователем | Результат запроса к БД, обновленная html-страница игры |
| Написание отзыва | Код результата выполнения |

Также, помимо указанной выше выходной информации, каждое действие может вернуть сообщение об ошибке в случае некорректности данных или недоступности БД.

### Алгоритм решения

Общий алгоритм работы модуля:

1. Получение запроса пользователя
2. Вычисление действия пользователя
3. Обработка параметров запроса
4. Подготовка и отправка запроса в БД
5. Получение и обработка ответа от БД
6. Преобразование информации от БД в html-страницу по шаблону

Схема работы модуля представлена на рисунке 2.3.



Рис 2.3. Схема общего алгоритма работы модуля

Алгоритм поиска игр по критериям:

1. Получение поискового запроса пользователя с критериями
2. Составление поисковой команды на основе полученных критериев
3. Получение ответа от БД
4. Если критериям соответствует только 1 игра, то возврат html-страницы игры
5. Если критериям соответствует больше 1 игры, то возврат html-страницы со списком ссылок на страницы игр, удовлетворяющим критериям поиска
6. Если в базе не нашлось игр по данным критериям, то возврат сообщения об этом.

Схема алгоритма поиска игр представлен на рисунке 2.4.

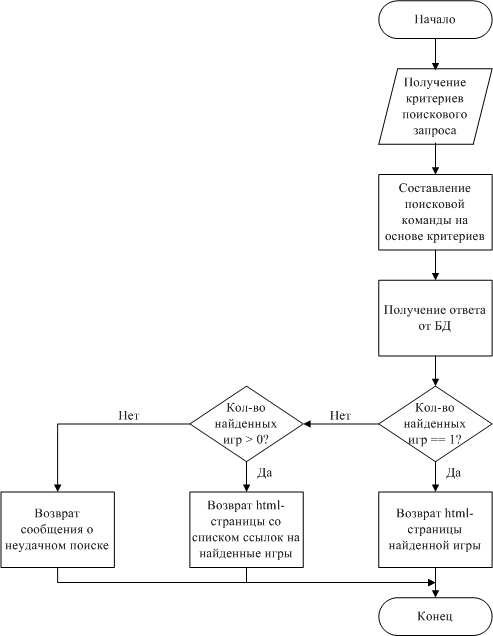


Рис 2.4. Схема алгоритма поиска игр по критериям

Подробный алгоритм фильтрации списка игр по критериям представлен на рисунке 2.5.

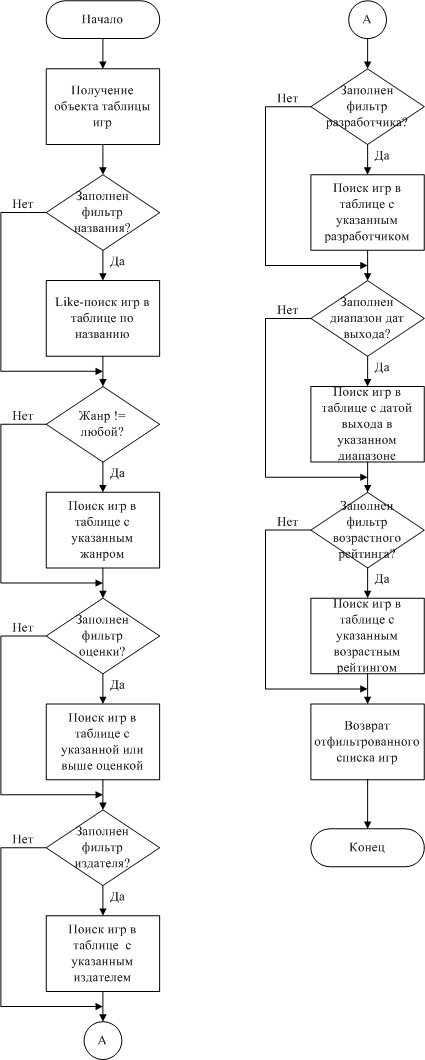


Рис 2.5. Схема алгоритма фильтрации списка игр по критериям

Схема алгоритм добавления, удаления и изменения статуса игры в библиотеке пользователя представлена на рисунке 2.6.

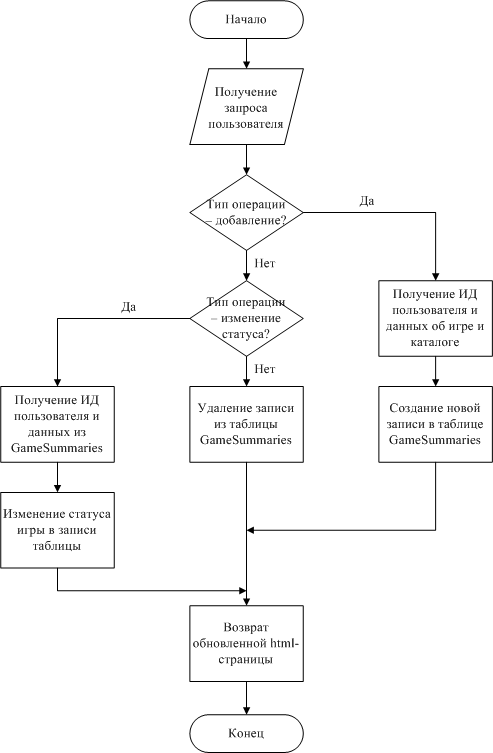


Рис 2.6. Схема алгоритма добавления, удаления и изменения статуса игры в библиотеке пользователя

## Описание модуля интеграции с платформами крупных игровых дистрибьюторов

### Назначение и характеристика

Назначение данного модуля заключается в регулярном обновлении базы игр за счет настроенной интеграции с крупными игровыми платформами, т.к. основная цель разрабатываемой системы – это объединить данные с разных платформ. На данный момент настроена интеграция с платформой Steam. Модуль состоит из 2 подсистем:

1. Подсистема интеграции, задача которой получать обновления данных через API, обрабатывать полученные данные и заносить в базу данных;
2. Подсистема управления запусками процессов, задача которой заключается в периодическом запуске методов подсистемы интеграции по расписанию, заданному в формате cron в конфигурационном файле. В качестве основы подсистемы используется фреймворк Quartz 3.3.2.

### Используемые данные

Для интеграции с каждой платформой будет использоваться отдельный блок настроек в конфигурационном файле сервиса appsettings.json. В нем будут задаваться основные параметры интеграции, такие как расписание запуска в формате cron, ограничение на количество записей за один сеанс связи, уникальный идентификатор системы, путь до лог-файла с результатами интеграции и прочие, уникальные для платформы, данные (например, уникальный ключ доступа).

Для отслеживания истории интеграции и избегания дублирования данных используется таблица базы данных IntegrationInfos, каждая запись которой представляет краткую информацию о полученной извне игры.

Состав столбцов таблицы:

ExternalSystemDescriptor – уникальный идентификатор платформы, с которой настроена интеграция,

ExternalGameId – уникальный идентификатор игры на внешней платформе,

InternalGameId – уникальный идентификатор игры в разрабатываемом сервисе,

Date – дата и время занесения информации об игре.

Во время обработки полученной через API информации подсистема интеграции проверит наличие в базе данных сервиса полученной игры во избежание дублирования.

Также используемыми данными можно считать адреса API сервисов, для которых не существует библиотек-оберток на технологии .Net Core. В этом случае адреса API также заносятся в конфигурационный файл.

### Результаты решения

Результатом работы модуля являются:

1. Обновление таблицы Games в БД сервиса;
2. Обновление таблицы IntegrationInfos в БД сервиса;
3. Лог-файлы для каждой настроенной интеграции, в которые заносится информация о ходе выполнения процесса и возникающих ошибках.

Для пользователя информация станет доступной после обновления текущей страницы или переходе на другую.

### Алгоритм решения

Алгоритм запуска процессов интеграции по расписанию:

1. Получить все сервисы интеграции
2. Создать объект управления расписанием

Для каждого сервиса:

* 1. Создать «работу»
  2. Получить расписание запуска сервиса из конфигурационного файла
  3. Создать триггер запуска на основе расписания
  4. Зарегистрировать в объекте управления расписанием работу с ее триггером

Схема работы алгоритма запуска процессов интеграции представлена на рисунке 2.7.



Рис. 2.7. Схема работы алгоритма запуска процессов интеграции

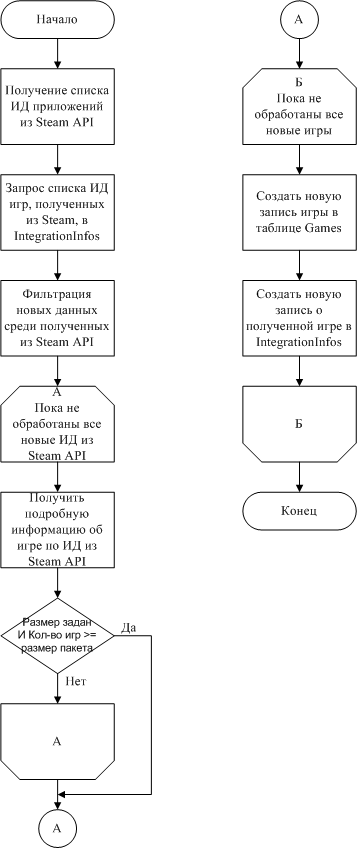


Рис 2.8. Схема работы процесса интеграции со Steam

Общий алгоритм процесса интеграции:

1. Получение данных их API
2. Фильтрация новых данных из полученных с помощью IntegrationInfos
3. Занесение новых данных в таблицу Games
4. Создание записи о добавленной игре в IntegrationInfos

В случае возникновения ошибок в процессе получения данных, информация о них будет занесена в лог.

Схема алгоритма интеграции со Steam представлена на рисунке 2.8.

## Описание контрольного примера

### Назначение

Контрольный пример служит для проверки работоспособности системы. В контрольном примере предусмотрена проверка следующих функциональных возможностей системы:

1. Регистрация и авторизация пользователей на сервисе
2. Поиск игр по критериям
3. Добавление, изменение и удаление игры из библиотеки пользователя
4. Написание отзывов к играм
5. Загрузка информации об играх с внешних игровых платформ

### Исходные данные

Исходными данными для контрольного примера являются таблицы БД:

1. AspNetUsers – таблица с данными о пользователях
2. Games – таблица с данными об играх
3. Reviews – таблица с отзывами
4. GameSummaries – таблица с историей добавлений игр в библиотеки
5. IntegrationInfos – таблица с информацией об играх, полученных с внешних платформ

Для запуска процесса интеграции с внешней платформой используются настройки из конфигурационного файла сервиса.

Пример настройки для интеграции со Steam:

"IntegrationSettings": {

"SteamIntegrationSettings": {

"CronSchedule": "0 29 \* ? \* \* \*",

"MaxPacketSize": 100,

"SteamWebApiKey": "<SteamWebApiKey>"

}

### Результаты расчета

При входе в систему регистрируем нового пользователя (рис. 2.9) или авторизуемся в существующей учетной записи (рис. 2.10) для полного доступа к функционалу сервиса.

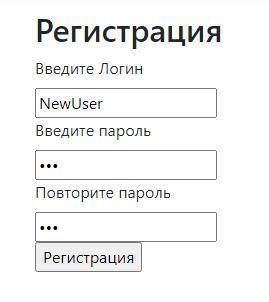


Рис. 2.9. Форма регистрации

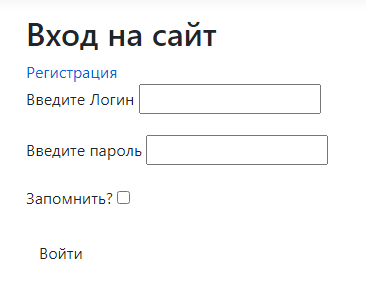


Рис 2.10. Форма авторизации

После авторизации пользователю становится доступен его личный кабинет с данными профиля и библиотекой (рис 2.11).

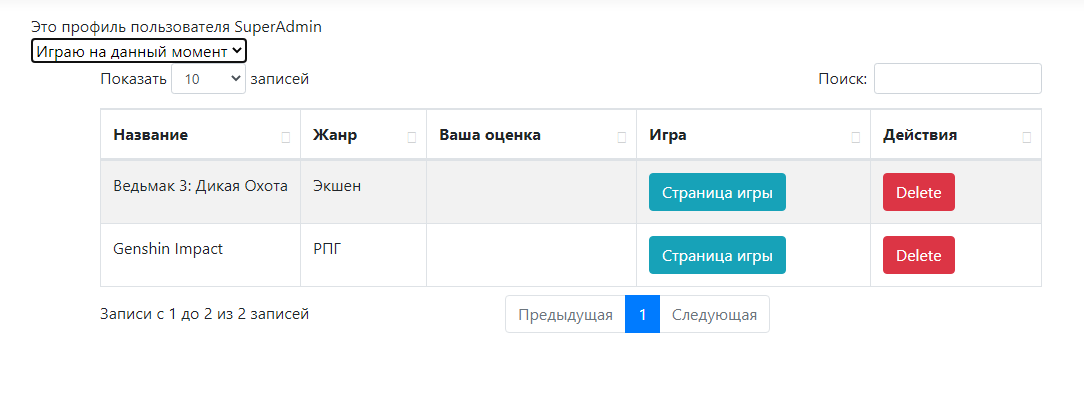


Рис 2.11. Страница личного кабинета пользователя

Для поиска игр по критериям необходимо перейти в меню «Расширенный поиск» на главной странице сервиса или воспользоваться строкой для быстрого поиска игры по наименованию (рис 2.12).

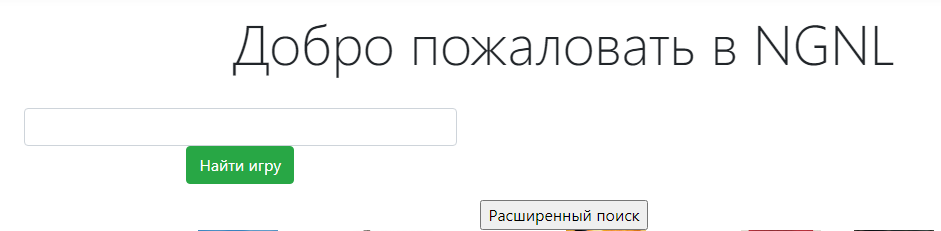


Рис 2.12. Главная страница

После перехода в меню «Расширенный поиск» откроется панель с фильтрами и общий список игр (рис 2.13).

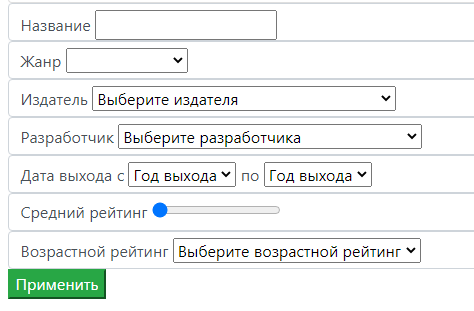


Рис 2.13. Панель фильтрации

После указания необходимых критериев и нажатия кнопки «Применить» в на странице появился отфильтрованный список игр (рис 2.14).

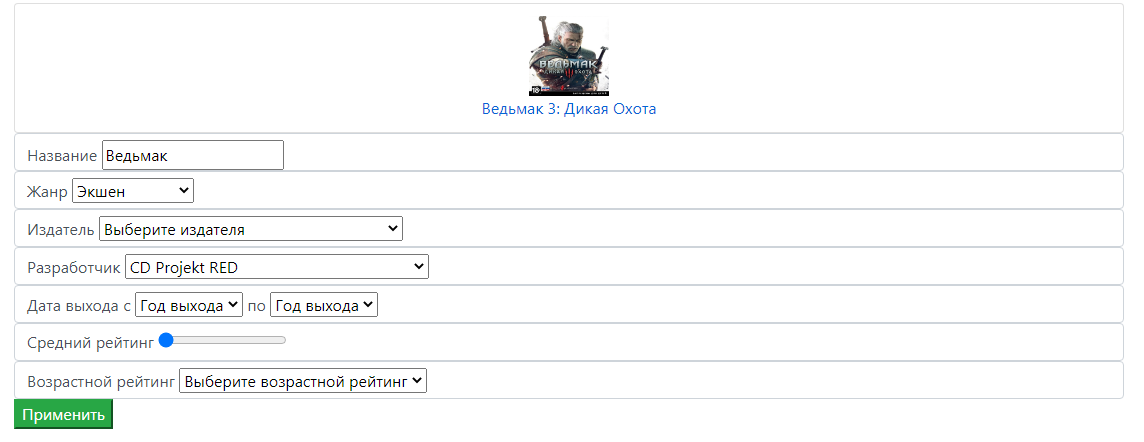


Рис 2.14. Список игр после применения поиска по критериям

Для перехода на страницу игры (рис 2.15) необходимо нажать по выбранному элементу в списке.



Рис 2.15. Страница игры

Помимо информации об игре на странице доступны следующие возможности:

1. Добавить игру в библиотеку. Для этого необходимо выбрать один из предложенных списков («Запланировано», «Играю на данный момент», «Пройдено», «Брошено») и нажать «Кнопку добавить в список». После этого игра появится в личном кабинете пользователя в указанном списке (рис. 2.11).
2. Если игра уже находится в библиотеке пользователя, то можно будет сменить список и нажать кнопку «Обновить».
3. Написать отзыв к игре.

После написания пользователем отзыва к игре, он попадает в специальный список администратора для дальнейшего модерирования (рис 2.16). Администратор может опубликовать отзыв, если в нем не содержится нежелательного контента или удалить отзыв нажатием на соответствующие кнопки в списке.

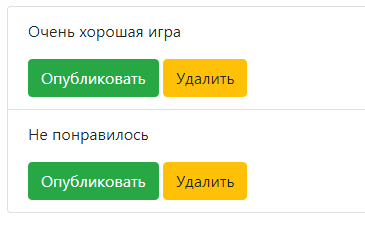


Рис 2.16. Меню модерирования отзывов

Во время сеанса интеграции идет обращении к API внешней системы и получение информации в виде структурированного текста (например, в формате json).

Пример получаемой информации об игре из Steam API:

{

"730": {

"data": {

"type": "game",

"name": "Counter-Strike: Global Offensive",

"required\_age": 0,

"is\_free": true,

"controller\_support": "full",

"detailed\_description": "<Описание игры>",

"header\_image": "<Ссылка на лого>",

"developers": ["Valve"],

"publishers": ["Valve"],

"platforms": {

"windows": true,

"mac": true,

"linux": true

},

"genres": [{

"id": "1",

"description": "Экшены"

}

],

"release\_date": {

"coming\_soon": false,

"date": "21 авг. 2012"

},

"background": "<Ссылка на фоновое изображение>",

}

}

}

После обработки полученной информации для каждой полученной игры создается запись в таблице Games (рис 2.17) и сервисная информация о сеансе интеграции в IntegrationInfos (рис 2.18).

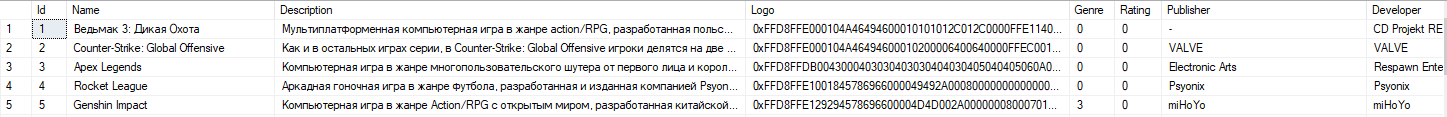


Рис 2.17. Информация из таблицы Games

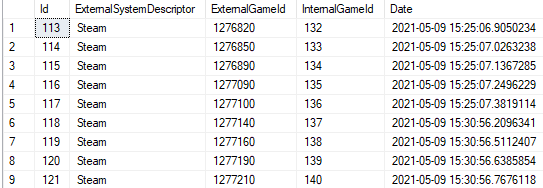


Рис 2.18. Информация из таблицы IntegrationInfos

### Результаты испытания сервиса-агрегатора актуальной информации с игровых платформ

В результате испытания системы можно сделать вывод, что данная программа выполняет поставленные перед ней задачи в полном объеме.