УДК 004.04

РЕАЛИЗАЦИЯ СИСТЕМЫ УПРАВЛЕНИЯ МНОГОПОЛЬЗОВАТЕЛЬСКОЙ ОНЛАЙН-ИГРОЙ

С.Д. ПАШКЕВИЧ
(Представлено: канд. техн. наук, доц. А.Ф. ОСЬКИН)

Предлагается подход к реализации системы управления многопользовательской онлайн-игрой, как распределенного веб-приложения, взаимодействие компонентов которого осуществляется согласно принципам архитектурного стиля REST. Описываются функциональные возможности системы управления и ее структурные компоненты. Приводится диаграмма компонентов веб-приложения.

Введение. Несмотря на популярность игр, ориентированных на компьютеры, мобильные многопользовательские онлайн-игры в последние годы набирают все большую популярность. Рост аппаратных характеристик мобильных устройств за последние пять лет расширил возможности разработчиков игр, позволяя создавать намного более крупные и амбициозные проекты [1]. Зачастую онлайн игры требуют наличие специальной функциональности, которая выносится в отдельные приложения, позволяющие осуществлять администрирование игрового процесса.

Системы управления предназначены не только для осуществления контроля за игровым процессом, но и для получения статистической информации, необходимой для анализа данных, а также изучения интересов игроков.

Администратор приложения получает возможность самостоятельно определять права пользователей, а также редактировать, формировать либо изменять структуру отображаемой информации, осуществлять техническую поддержку пользователей и рассылку новостей. Пользователь, в свою очередь, получает возможность редактировать свою личную информацию, следить за своей игровой статистикой и статистикой других игроков, обратиться за помощью к администраторам игры и так далее. Перечисленные возможности значительно облегчают поддержку и дальнейшее развитие игры в целом, а также обуславливают необходимость в наличии системы управления, так как дополнительная функциональность для администрирования не относится к игровому процессу напрямую и ее наличие не всегда удобно в рамках игрового приложения, особенно мобильного.

Игровой мир представляет собой галактику, которая состоит из множества секторов, которые содержат определенное количество планет. На планетах игрок может брать квесты, покупать/продавать товары, улучшать оборудование и производить починку космического корабля, с помощью которого осуществляется перемещение по игровому миру. Каждому игроку предоставляется персонаж, имеющий следующие характеристики: точность пилота, маневренность пилота, знание техники, навык торговли, лидерство, обаяние пилота. После выполнения заданий, либо участия в сражениях, игроку начисляется опыт, который может быть использован для улучшения характеристик своего персонажа. Галактику населяют три расы: зеленые, красные и синие. Персонаж каждой расы обладает характерными для нее свойствами. Игроку также предлагается возможность выбора расы.

Система управления многопользовательской онлайн-игрой. Функциональность, предоставляемая системой управления, обуславливается описанием игры. Страница *регистрации и авторизации* является входной точкой в приложение, позволяет пользователю зарегистрироваться, либо авторизироваться через популярные социальные сети: Facebook, Google+, Twitter. После успешного прохождения процедуры авторизации/регистрации пользователю будут присвоены специальные права доступа, которые могут ограничивать либо предоставлять доступ к различным ресурсам системы управления, например, таким как *админпанель*, также пользователю станет доступна страница *профиль пользователя*.

Страница *профиль пользователя* отображает раздел *игровой статистики*, раздел *личной информации*, раздел *управления игровым процессом* и *меню*, которое предоставляет возможность перемещаться по страницам системы управления, доступным пользователю.

Раздел *личной информации* предназначен для отображения информации о пользователе: имя, e-mail, возраст, пол.

Раздел *игровой статистики* предназначен для отображения информации об игровом прогрессе пользователя: точность пилота, маневренность пилота, знание техники, навык торговли, лидерство, *обаяние пилота*. Кроме этого игровая статистика включает в себя отображение сведений о космическом корабле. Данные, отображающиеся в разделах *личной информации* и *игровой статистики*, принимаются с приложения-сервер при загрузке страницы и динамически обновляются.

Раздел *управления игровым процессом* предоставляет возможность управлять функциональностью, которая обеспечивает взаимодействие с игровым процессом, таким образом пользователю предоставлены следующие возможности:

– возможность модифицировать компоненты космического корабля, такие как: *корпус*, *двигатель*, *топливный бак*, *радар*, *сканер*, *ремонтный дроид*, *генератор защитного поля*, *вооружение*;

– возможность изменить расу, за которую он играет, среди доступных: *зеленые, красные* и *синие*;

– возможность просматривать информацию о планетах, которые доступны пользователю, а также выбрать определенную планету в качестве базы.

Раздел *меню* позволяет перейти на страницы: *адminпанель* (при наличии прав доступа пользователя), *галактика, игроки, техническая поддержка*.

Страница *адminпанель* позволяет: управлять учетными записями пользователей, отвечать на вопросы пользователей из раздела технической поддержки, управлять игровым сервером.

Страница *галактика* отображает список доступных звездных систем, пользователь может открыть любую из них. После выбора звездной системы будет отображен список планет, пользователь может открыть подробную информацию о планете, кроме этого будет выведен список игроков, которые находятся в этой системе, либо на планете в данный момент.

Страница *игроки* выводит список игроков в зависимости от выбранного фильтра: зарегистрированные, сейчас в игре. Выбрав игрока из списка можно ознакомиться с его игровой статистикой.

Страница *техническая поддержка* позволяет осуществлять обратную связь пользователей с администраторами.

Веб-клиент должен предоставлять следующие возможности, соответствующие концепции:

– динамическое обновление данных, которые будут изменяться при взаимодействии игроков друг с другом в рамках игрового процесса: статистика, текущий уровень игрока, урон во время сражения, количество побед и поражений, информация о пользователях, взаимодействующих друг с другом;

– осуществлять управление игровым сервером и игровым процессом;

– обмен данными с игровым сервером с помощью WebSocket, для обновления информации, которая отображается на страницах;

– взаимодействие с приложением-сервер с помощью HTTP запросов.

Структурные компоненты системы управления. Наличие веб-браузера на всех современных компьютерах и мобильных устройствах обуславливает разработку веб-клиента, с помощью которого будет происходить взаимодействие пользователя и администратора с сервером и игровой средой.

Для реализации клиентской части веб-приложения выбран компонентно-ориентированный архитектурный стиль, позволяющий декомпозировать приложение на более маленькие логические составляющие, которые могут быть ориентированы на работу с определенной частью REST API. Благодаря этому, любой компонент может быть заменен с минимальными изменениями в остальных частях системы, так как взаимосвязанность между компонентами приложения минимальна, что ускоряет разработку и дальнейшее сопровождение приложения.

Динамическое обновление данных осуществляется с помощью технологии WebSocket, которая позволяет подключиться к игровому серверу и соответственно принимать необходимые данные.

Наличие пересекающихся возможностей у веб-клиента и мобильного игрового приложения, таких как регистрация, авторизация, управление профилем пользователя и так далее, обусловливают разработку приложения-сервер, которое будет представлять из себя REST API приложение и наличие централизованной базы данных. Архитектура REST требует использование большинства возможностей протокола HTTP [2]. REST-приложение обладает рядом преимуществ в сравнении с JSON-RPC, приложение-клиент не зависит от имен и аргументов процедур, имеет стандартные коды состояния, обладает поддержкой кеширования, также позволяет направлять запрос от приложения-клиента путем добавления дополнительной информации в URL, либо заголовок HTTP запроса [3].

Для ограничения доступа сторонних приложений, приложение-сервер позволяет производить аутентификацию на время выполнения запроса с помощью RBAC. За контролем доступа приложения-клиента к REST API отвечает Cross-origin resource sharing (CORS) – технология современных браузеров, которая позволяет предоставлять веб-странице доступ к ресурсам другого домена [4].

Исходя из концепции системы управления, приложение сервер должно позволять взаимодействовать с базой данных посредствам специальных запросов к определенным узлам. Каждый узел должен соответствовать объекту предметной области, например: пользователь, настройки, сообщения, космический корабль, галактика, компонент корабля, статистика. Запрос отправленный от клиента будет обработан контроллером, который также соответствует объекту предметной области, данный контроллер расположен в компоненте REST Controller Component, который взаимодействует с Model Component. Каждый контроллер должен предоставлять функциональность по считыванию, редактированию, удалению и добавлению новых записей.

Приложение-сервер включает следующие структурные компоненты:

– Database Component – реализует взаимодействие с базой данных;

– Active Record Component – предоставляет интерфейс для преобразования переданных параметров в SQL, взаимодействует с компонентом базы данных;

– Model Component – представляет собой модель объекта реального мира, реализующий интерфейс Active Record;

– CORS Component – компонент, который отвечает за предоставление доступа к ресурсам REST API;

– Service Component – предоставляет интерфейс для обработки данных, которые принадлежат различным моделям;

– RBAC Component – предоставляет интерфейс, позволяющий производить авторизацию на время выполнения запроса и определяет права доступа;

– REST Controller Component – предоставляет интерфейс для взаимодействия с REST API.

Веб-клиент состоит из следующих структурных компонентов:

– Firebase SDK (library) – предоставляет API для работы с сервисами от Google, а также API для реализации регистрации и авторизации;

– Controller Component – осуществляет взаимодействие с View и Http компонентом, предоставляет данные для отображение на пользовательском интерфейсе;

– View Component – формирует пользовательский интерфейс;

– HTTP Component – осуществляет взаимодействие с REST API, отвечает за авторизацию и регистрацию пользователей.

На рисунке изображена диаграмма компонентов, которая показывает разбиение программной системы на структурные компоненты.

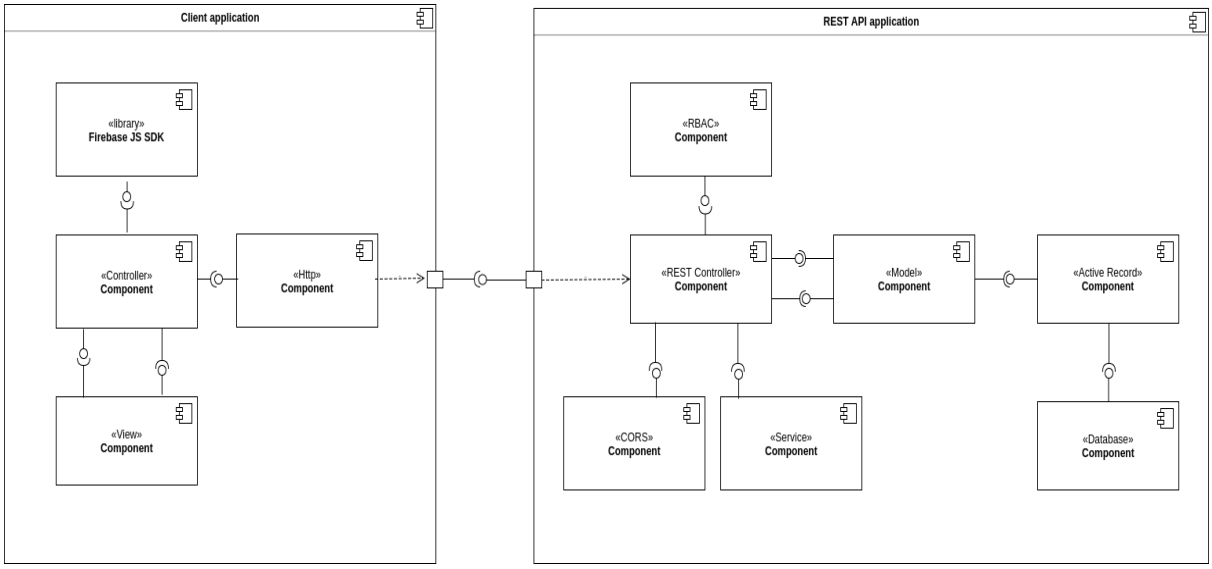

Рисунок. – Диаграмма компонентов веб-приложения

Заключение. Предложен подход к реализации системы управления многопользовательской онлайн игрой с использованием архитектурного стиля REST. Определены функциональные возможности системы управления и ее структурные компоненты.

ЛИТЕРАТУРА

1. Games-inn [Электронный ресурс] / Games-inn.ru © 2018. – Режим доступа: https://games-inn.ru/post/5-luchshih-igr-mmo-na-android-i-ios/. – Дата доступа: 15.09.2019.
2. Smartum [Электронный ресурс] / Smartum.pro © 2018. – Режим доступа: http://smartum.pro/ru/blog-ru/gid-dlya-nachinayuschih-po-rest-api/. – Дата доступа: 16.09.2019.
3. Qaru [Электронный ресурс] / Qaru © 2018. – Режим доступа: http://qaru.site/questions/2533/rest-vs-json-rpc. – Дата доступа: 17.09.2019.
4. Wikipedia [Электронный ресурс] / Wikimedia Foundation, Inc © 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Cross-origin_resource_sharing. – Дата доступа: 20.09.2019.