CS 411W Lab II

Product Specification

GameEye Team Yellow

18 October 2020

Table of Contents

1 Introduction

If the average gamer today wants to keep track of games that interest them, they face a number of challenges. To begin with there are just too many games released every year, thousands across the many platforms which exist. While some simpler games may have relatively short development periods, games like Skyrim or other complex games can take years to develop. During this time, gamers can easily forget about a game, especially if new information dwindles. Not to mention that all developers are not the same; smaller and more independent developers may not have the funds to keep gamers interested as their development progress. With the growth of the internet, there are now a multitude of sources to get information. From dedicated gaming news sites like The Verge to social media platforms like Twitter, there are too many places to look. To conquer these challenges, their needs to be a method for aggregating and organizing the vast amount of news that is published.

1.1 Purpose

The purpose of GameEye is to make it easier for gamers to stay up to date for their favorite games by delivering relevant news directly to them. GameEye shall allow each user to have their own personal watchlist consisting of games they have chosen to follow. GameEye shall gather information from multiple sources at regularly scheduled intervals. GameEye collects six different categories of data which include: news articles, tweets, reddit posts, images, and videos. To extract and classify the data as well as assign impact scores, GameEye utilizes machine learning. To keep users in tune, GameEye shall use notifications to alert them when new information is found.

GameEye looks to target active and casual gamers who play or are interested in multiple games. By curating their watchlist to their liking, the user shall be notified when relevant and important gaming information is released. GameEye does not intended to replace the sources from which it gathers content, it only serves as an aggregator. Content is scraped regularly every day but is not done real-time. Users can always navigate to the original source of information, as a link shall be provided. To preserve functionality, users will be limited to a certain number of games on their watchlist.

1.2 Scope

GameEye is a smart content and news aggregator that uses machine learning to deliver relevant content to its users. The users decide what games they would like to receive information on by way of their personal watchlist. For a subset of the information, new articles and tweets, impact scores will be given. GameEye uses machine learning to decide how to score each tweet or article. Machine learning shall also be used to ensure only important and relevant information is extracted from the articles and tweets. GameEye shall track both released and unreleased games, notify users whenever new information arises. All in all, GameEye shall provide a single source of information about the games that interest the user.

Due to time constraints and other limitations, the GameEye prototype will differ from the real-world product. The prototype shall only scrape news articles, and the tweets will be left alone for now. There will be no support for content archiving and there will be a limited amount of games available for the users watchlist.

1.3 Definitions, Acronyms and Abbreviations

1. **Aggregator:** A platform or service that collects and centralizes data.
2. **Angular Framework:** Platform for building mobile and desktop applications.
3. **API:** Application Programming Interface; a set of functions and procedures allowing the creation of applications that access the features or data of an operating system, application, or other services.
4. **Amazon Web Services (AWS):** Amazon subsidiary that provides on-demand cloud computing services and APIs.
5. **CSS:** Cascading Style Sheets; used to stylize webpages.
6. **Guest:** Initial role for users who have not created an account on GameEye.
7. **Hitlist:** List of highly watched video games by users.
8. **HTML:** HyperText Markup Language; used as markup for documents meant to be displayed in a web browser.
9. **Impact Score:** A score from 1 to 3 on the impact some news has on a game and its players. It is computed using machine learning.
10. **Indie Games:** Games developed by individuals or smaller teams of people without the financial support of larger game publishers.
11. **Integrated Development Environment (IDE):** A software application that includes a set of programming tools, notably a code editor and a debugger interface, to facilitate programming.
12. **IntelliJ IDEA:** IDE developed by JetBrains to write Java applications and will be used in the backend development of GameEye.

13. **Internet Games Database (IGDB):** Database of known video games; accessed through a REST API to populate GameEye's database.

14. **JavaScript:** Object-oriented programming language used to create dynamic behavior on webpages.

15. **JSoup Library:** Java library for working with real-world HTML.

16. **JUnit Java Framework:** A testing framework for Java.

17. **Keras:** An open-source neural-network library written in Python.

18. **MongoDB:** A cross-platform document-oriented database program.

19. **Noise Filtering:** Removal of news articles and other content that is irrelevant or unimportant to the user.

20. **OIDC Authentication:** Open-ID Connect (OIDC) is an authentication protocol based on the OAuth 2.0 family of specifications.

21. **Progressive Web Application (PWA):** A type of application software delivered through the web which is built using common web technologies including HTML, CSS, and JavaScript; PWAs are an evolution of traditional web applications and can be used to a certain degree while offline.

22. **Python:** An interpreted, high-level, general-purpose programming language.

23. **Representational State Transfer (REST):** A software architectural style used in creating web services.

24. **RSS Feed:** Really Simple Syndication (RSS) is a web feed that allows users and applications to access updates to websites in a standardized, computer-readable format.

25. **Scikit-learn:** Open-source machine learning library for the Python programming language.

26. **SpaCy:** Open-source library for advanced natural language processing.

27. **Spring Framework:** Application framework and inversion of control container for the Java platform.

28. **Tester:** GameEye prototype users who will provide feedback on their experience with the application.

29. **Web Scraping:** Automated extraction of data from websites.

30. **WebStorm:** IDE developed by JetBrains for writing JavaScript and web-related code.

1.4 References

1. Anderton, K. (2019, June 26). The Business of Video Games: Market Share for Gaming Platforms in 2019. *Forbes.* [Infographic]. Retrieved from:https://www.forbes.com/sites/kevinanderton/2019/06/26/the-business-of-video-games-market-share-for-gaming-platforms-in-2019-infographic/#c1793427b254

2. Angelopoulos, A., Cook, J., Diasanta, C., Epps, J., Hund, B., Lewis, B., Wilson, A. (2020). Lab 1 – GameEye Product Description.

3. Dietz, J. (2011, June 23). *30 games that emerged from development hell.*Metacritic. Retrieved from: https://www.metacritic.com/feature/games-that-shed-vaporware-status

4. Gough, C. (2020, June 25*). Number of games released on Steam worldwidefrom 2004 to 2019*. Statista. Retrieved from:https://www.statista.com/statistics/552623/number-games-released-steam/

5. Gough, C. (2020, September 22). *Number of active gamers worldwide from 2015 - 2023*. Statista. Retrieved from:https://www.statista.com/statistics/748044/number-video-gamers-world/

6. Gough, C. (2020, September 28). *Number of available gaming apps at Google Play from 1st quarter 2015 to 2nd quarter 2020*. Statista. Retrieved from:https://www.statista.com/statistics/780229/number-of-available-gaming-apps-in-the-google-play-store-quarter/

7. Humphries, M. (2019, September 18). Twitch Acquires Gaming Database Website IGDB. *PCMag*. Retrieved from: https://www.pcmag.com/news/twitch-acquires-gaming-database-website-igdb

8. Rose, M. (2014, May 15). *How the surge of Steam releases will affect game developers*. Gamasutra. Retrieved from:https://www.gamasutra.com/view/news/217583/How_the_surge_of_Steam_releases_will_affect_game_developers.php

9. Web, K. (2019, October 1). *The $120 billion gaming industry is going through more change than it ever has before, and everyone is trying to cash in*. Business Insider. Retrieved from: https://www.businessinsider.com/video-game-industry-120-billion-future-innovation-2019-9

1.5 Overview

This product specification details the software and hardware components, features, and external interfaces of the GameEye prototype. The information provided in the remaining sections in this document provide a detailed description of: (a) the software and hardware used for implementing the GameEye prototype, (b) the features that the GameEye prototype will provide including any limitations, and (c) the interfaces that GameEye will provide to other software.

2 General Description

2.1 Prototype Architecture Description

The frontend of GameEye is a cross-platform single page web application built with Angular. It provides the user interface and communicates with the primary backend through the REST API. It also communicates with firebase services for user authentication.

The primary backend is built with Spring Boot. It implements the REST API for interaction with the frontend and controls the web scraping. It communicates with the main database for game data, user and settings information. The main database is provided by MongoDB. The machine learning backend allows for the extraction and categorization of data during the web scraping process and is built with TensorFlow Serving. It also communicates with the primary backend with through a REST API. ElasticSearch database will store game titles and allow GameEye to implement autocompletion in the search bar which users will use to add games to their watchlist. IGDB is a gaming database containing thousands of games and is shall be the initial source of the titles GameEye uses. Firebase services are also used in the backend for user authentication and to provide push-notifications for the user.

2.2 Prototype Functional Description

The frontend is responsible for providing the user interface to communicate with the backend. The prototype should provide a page where users can create a GameEye account by using their email and providing a password. This page should also allow the user to create an account by using a Google or Microsoft account. When an account is created, the user shall be provided a page which displays the games currently in their watchlist. From the watchlist page, user shall be able to access another page which allows them to add games to their watchlist. There shall be a page where the users can see the information resources for a specific game; this page shall also be accessible from the watchlist page. The resources page shall allow the user to access a page news articles that GameEye has gathered are displayed. To allow the user a more customizable experience users shall be provided page where they can change settings and account information. Last but not least user shall be notified when relevant news articles have been scrapped.

The backend is responsible for the logic, operations, and data storage that provide functionality for the frontend. It should collect video game metadata from IGDB. MongoDB shall be used to store the bulk of GameEye's data such as store video game, user, and setting information. It should use ElasticSearch shall be used for advancing game title searching and providing autocompletion for the frontend. It should utilize web scraping routinely to scour video game news sites for articles. It shall use machine learning to analyzed scraped articles. Articles which are not relevant or were previously scraped shall be discarded. Scraped articles shall be assigned impact scores also through the use of machine learning. It shall push the notifications that the user receives upon the presence of newly scraped articles. In order to communicate with the front end, it should implement the REST API. To prevent unauthorized access to the

backend, it should use role-based endpoint authorization. Finally, it should provide adequate tools for developers and testers.

2.3 Software Interfaces (Paragraphs)

a. MongoDB
b. ElasticSearch
c. IGDB
d. Firebase Authentication
e. Firebase Cloud Messaging