Lab 2 - GameEye Product Specification Outline

Angelos Angelopoulos

Old Dominion University

CS 411W: Professional Workforce Development II

Professor Janet Brunelle

October 18, 2020

Version 1

Table of Contents

List of Figures

List of Tables

1. Introduction

The video game industry has seen incredible growth in the past decades. Today, there are more than 2.47 billion gamers worldwide (Gough, 2019b), and the video game industry is projected to reach a value of $196 billion in 2022 (Webb, 2019), surpassing the film industry. Thousands of video games are released every year (Anderton, 2019). Just in 2018, more than 9000 games were released for personal computers on the Steam platform, and this number has been increasing in the past years (Gough, 2019a). By also considering console and mobile games, this number increases to tens of thousands (Gough, 2019c). With so many games releasing annually, a problem appears. Gamers may desire to play many upcoming games but staying up to date for all of them can be time-consuming. Simultaneously, video game developers spend millions on marketing as promoting a video game is crucial for driving sales. In particular, indie developers face an increased challenge in maintaining the public spotlight. Another issue is that information about video games is decentralized, making it more difficult to find news.

To solve these problems, a way to aggregate, categorize, and rank game information is needed. The solution is GameEye, a platform where users can search for and follow video games. Users will have a personal watchlist of video games. GameEye will keep track of followed games and notify users when new content is released online about them. The application will collect and categorize video game data from the internet, including news articles, tweets, Reddit posts, images, and YouTube videos. Using machine learning, news articles and

tweets will be classified depending on their content, and a multi-factor impact score will be computed and assigned based on how impactful the news is for a game and its users. GameEye aims to be the gamer's eye in the video game industry.

1.1. Purpose

GameEye is a product with the purpose to organize the vast amount of information in the video game industry. This is achieved by automating the discovery of new video game information in various formats such as news articles, tweets, Reddit posts, images, and videos. GameEye processes and analyzes discovered information using machine learning to reduce noise and deliver relevant information to users.

GameEye is a simple and intuitive solution. The user simply enters a few games they are interested in, and the application will keep track of them and notify the user of any new information. The application is being developed for casual and highly active gamers alike as both types of end-users benefit from it. Casual gamers who play a few games less frequently would value a plug-and-play solution where they can enter a few games that they are interested in and save time by getting notified about important news. Highly active gamers who frequently play and are interested in many games also benefit from the application as they will not have to manually keep track of so many games.

GameEye seeks to complement, not replace, current news sources and other media by aggregating and processing information from many different sources. Information from multiple media is scraped regularly every day and users are notified about relevant information. Links to navigate to the sources of information are always provided so that users can read the full information on the original website it was retrieved from. In this way, information sources such as video game news websites benefit from the increased traffic GameEye brings to them. The

application does not scrape data in real-time, nor allows users to have unlimited games in their watchlist, as both have high operational expenses.

1.2. Scope

GameEye is a smart content and news aggregator that uses machine learning to deliver relevant content to the user. It saves time by providing a central location for video game news and by notifying users about important developments. The goals of the application are to make the lives of video game players easier, as well as to help game developers maintain more attention to their games. GameEye aims to be the middleman between the gamer and the video game industry, performing tasks and analysis that the gamer would otherwise have to do manually.

The prototype will have a subset of the features of the real-world product. This subset is meant to demonstrate the applicability and benefits of the application. The following features are implemented in the prototype: (a) news article scraping, (b) machine learning impact scoring, (c) game title autocompletion, (d) user authentication and authorization, (e) user watchlists, (f) push notifications, (g) cross-platform support, (h) most-watched games list, and (i) game logos and article thumbnails. Furthermore, a limited amount of games will be supported.

[This space is intentionally left blank]

1.3. Definitions, Acronyms, and Abbreviations

Aggregator: A platform or service that collects and centralizes data.

Angular Framework: Platform for building mobile and desktop applications.

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.

CSS: Cascading Style Sheets; used to stylize webpages.

Guest: Initial role for users who have not created an account on GameEye.

Hitlist: List of highly watched video games by users.

HTML: HyperText Markup Language; used as markup for documents meant to be displayed in a web browser.

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.

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

Indie Games: Games developed by individuals or smaller teams of people without the financial support of larger game publishers.

IDE: Integrated Development Environment. A software application that includes a set of programming tools, notably a code editor and a debugger interface, to facilitate programming.

IntelliJ IDEA: IDE developed by JetBrains to write Java applications and will be used in the backend development of GameEye.

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

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

JUnit Framework: A testing framework for Java.

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

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

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

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

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.

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

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

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.

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

Single-page Application: A web application that interacts with the user by dynamically rewriting the current web page with new data from a web server instead of reloading new pages.

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

Spring Boot: Application framework and inversion of control container for the Java platform.

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

Web Scraping: Automated extraction of data from websites.

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

[This space is intentionally left blank]

1.4. References

Angelopoulos, A., Cook, J., Diasanta, C., Epps, J., Hund, B., Lewis, B., Wilson, A. (2020, October 4). *Lab 1 – GameEye Product Description*. Retrieved from: https://www.cs.odu.edu/~411yello/pages/labs.html

Anderton, K. (2019, June 26). *The Business of Video Games: Market Share for Gaming Platforms in 2019 [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

Dietz, J. (2011, June 23). *30 Games That Emerged from Development Hell*. Retrieved from: https://www.metacritic.com/feature/games-that-shed-vaporware-status

Gough, C. (2019, August 9*). Number of games released on Steam 2018*. Retrieved from: https://www.statista.com/statistics/552623/number-games-released-steam/

Gough, C. (2019, August 9). *Number of gamers worldwide 2021*. Retrieved from: https://www.statista.com/statistics/748044/number-video-gamers-world/

Gough, C. (2019, October 9). *Google Play: Number of available games by quarter 2019*. Retrieved from: https://www.statista.com/statistics/780229/number-of-available-gaming-apps-in-the-google-play-store-quarter/

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

Rose, M. (2014, May 15). *How the surge of Steam releases will affect game developers*. Retrieved from:

https://www.gamasutra.com/view/news/217583/How_the_surge_of_Steam_releases_will_affect_game_developers.php

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

[This space is intentionally left blank]

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 gives a detailed description of (a) the software and hardware used for implementing the prototype, (b) the features that the prototype will provide including any limitations, and (c) the interfaces that the prototype has with other software. The functional requirements of the prototype are enumerated and detailed in a separate document titled “Lab 2 Section 3 – GameEye Product Requirements”.

2. General Description

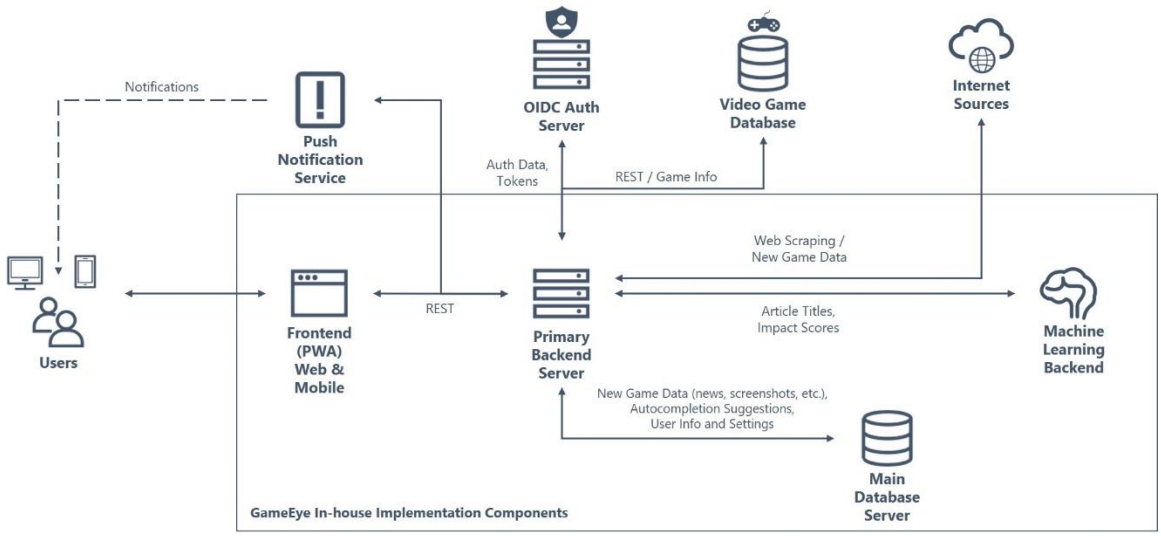
GameEye has five core functionalities. The first is the aggregation of online resources about video games, including news articles, tweets, Reddit posts, images, and videos. The second is the organization of online resources by game and type (e.g. grouping news articles, grouping tweets). The third functionality uses machine learning to classify and rank news articles and tweets based on their subject and their impact on the game and the user. The fourth is the notification of users about new content. The fifth is the ability for users to search for games to add to their watchlists, with support for autocompleting game titles.

2.1. Prototype Architecture Description

The prototype for GameEye consists of several in-house and third-party components. The in-house components are: (a) frontend, (b) primary backend, (c) web scraping backend, (d) machine learning backend, and (e) main database server. The third-party components are: (a) IGDB, a video game database, (b) Firebase Authentication, a Google authentication service, and (c) Firebase Cloud Messaging, a push notification service. These components are illustrated in Figure 1.

Figure 1

Major Functional Components of the GameEye Prototype.



All in-house components are hosted on a virtual machine provided by the ODU Computer Science department. The virtual machine is protected by the CS department's firewall and can only be accessed from within the network or by using a VPN. All servers hosted on the VM, including the frontend, backend, and databases, listen on separate ports. All the servers are secured and require some form of authentication to access. In more detail, the components hosted on this VM are: (a) Angular Frontend, (b) Spring Boot primary backend, (c) TensorFlow Serving machine learning backend, (d) MongoDB database, (e) ElasticSearch database. The Angular frontend provides a web-based interface for GameEye. The Spring Boot primary backend provides a REST API for the frontend, performs web scraping, communicates with the machine learning backend, accesses the two databases, and communicates with IGDB and Firebase services. The TensorFlow Serving machine learning backend provides a REST API to the primary backend for receiving news article titles and providing impact scores. Finally, the

MongoDB database stores all data related to GameEye while the ElasticSearch database stores game titles and provides advanced searching features.

2.2. Prototype Functional Description

The GameEye frontend is a single-page web application built with, HTML, CSS, and TypeScript. It serves as the user interface for GameEye. As a web application, it is cross-platform and targets desktop and mobile devices. The frontend communicates with the primary backend to retrieve data. The primary backend also coordinates and schedules web scraping. It provides a REST API for communication with the frontend, web scraping backend, databases, and other third-party services. The primary backend is built with the Spring Boot framework and Java.

Web scraping is carried out by the primary backend in the prototype, but in the real-world product, it would be carried out by a cluster of computers. The primary backend orchestrates web scraping, launching the operations twice a day and processing scraped data before storing it in the database. A REST API is provided for controlling web scrapers, such as forcing web scraping outside of scheduled times, if necessary. The library used for web scraping is jsoup.

Machine learning is implemented as a dedicated backend. It communicates with the primary backend web scrapers to process new data. This backend is responsible for computing impact scores. It is built with Python, Keras, scikit-learn, and spaCy. The machine learning backend uses a TensorFlow Serving server exposing a REST API for high-performance machine learning inference.

GameEye has extensive data storage requirements and would normally require a dedicated database server. For the prototype, the two databases used by GameEye are hosted on the same

server together with the two backends and the frontend. The first database is a MongoDB instance that stores all the data related to GameEye, including data retrieved from web scraping. Notably, it stores data retrieved from IGDB for increased performance and redundancy. The second database is ElasticSearch, and it stores game titles. More importantly, it maintains complex data structures for efficient searching. Because of this, the ElasticSearch database is used to provide auto-completion suggestions for game titles. Both databases are accessed by the primary backend.

The prototype implementation also makes use of a few third-party services: (a) IGDB, (b) Firebase Authentication, and (c) Firebase Cloud Messaging. IGDB is a large video game database that GameEye uses to obtain video game metadata. Firebase Authentication is used to implement authentication and authorization mechanisms. Finally, Firebase Cloud Messaging is used to send push-notifications to users.

Table 1

Lists the features that will be available in the real-world product and the prototype.

Category	Feature	RWP	Prototype
General			
	Cross-Platform Support (Desktop, Mobile)	Full Functionality	Full Functionality
	Offline Support	Full Functionality	No Functionality
	Local Caching	Full Functionality	No Functionality
	Connection Interruption Resiliency	Full Functionality	No Functionality
Authentication			
	User Login	Full Functionality	Full Functionality
	User Registration	Full Functionality	Full Functionality
	External Provider Login & Registration	Full Functionality	Full Functionality
	Persistent Sessions	Full Functionality	Full Functionality
	Password Recovery	Full Functionality	Full Functionality

Account Management			
	Change Profile Information	Full Functionality	Full Functionality
	Change Password	Full Functionality	Full Functionality
	Delete Account	Full Functionality	No Functionality
Searching			
	Search for Games	Full Functionality	Partial Functionality
	Search Autocompletion	Full Functionality	Partial Functionality
Game Tracking			
	Add Games to Watchlist	Full Functionality	Partial Functionality
	Remove Games from Watchlist	Full Functionality	Full Functionality
	News Articles (Web Scraping)	Full Functionality	Partial Functionality
	Tweets (Web Scraping)	Full Functionality	No Functionality
	Reddit Posts (Web Scraping)	Full Functionality	No Functionality
	Images (Web Scraping)	Full Functionality	No Functionality
	Videos (Web Scraping)	Full Functionality	No Functionality
	Resource Thumbnails	Full Functionality	Partial Functionality
	Game Thumbnails	Full Functionality	Partial Functionality
	Source Website Redirection	Full Functionality	Partial Functionality
	Resource Organization	Full Functionality	Full Functionality
	Archived Resources	Full Functionality	No Functionality
	Most-Watched Games List	Full Functionality	Full Functionality
Settings			
	Show Archived Resources Option	Full Functionality	No Functionality
	Show Impact Scores Option	Full Functionality	Full Functionality
	Receive Notifications Option	Full Functionality	Full Functionality
	Receive Notifications Per Category Option	Full Functionality	No Functionality
	Receive Notifications Per Impact Score Option	Full Functionality	Full Functionality
Machine Learning			
	Impact Scoring	Full Functionality	Partial Functionality
	Resource Classification	Full Functionality	No Functionality
	Important Information Extraction	Full Functionality	No Functionality
Notifications			
	Push-Notifications for New Game Updates	Full Functionality	Full Functionality
	UI Count of Notifications for Each Game	Full Functionality	Full Functionality
	UI Count of Notifications for Each Resource Category	Full Functionality	Full Functionality

	Cross-Platform Notifications	Full Functionality	Full Functionality
	Suggested Video Game Notifications	Full Functionality	No Functionality

2.3. External Interfaces

This section identifies the physical, logical, and user interfaces used by the GameEye prototype. The characteristics of each interface are detailed.

2.3.1. Hardware Interfaces

The prototype does not make use of any specialized hardware interfaces. All components are hosted inside the ODU CS network and utilize its computer and networking infrastructure.

2.3.2. Software Interfaces

The prototype makes use of several software libraries. Angular is the framework used for implementing the frontend, while Spring Boot is used for implementing the backend. Machine learning is implemented using TensorFlow and Keras, and the impact score model is served using TensorFlow Serving. The SpaCy library is used for text processing and scikit-learn is used as a general utility machine learning library. Web scraping is implemented using the jsoup library, which is used for parsing HTML. The MongoDB database is accessed using a specialized MongoDB driver for Spring Boot. Similarly, the ElasticSearch database is also accessed using a specialized driver.

2.3.3. User Interfaces

The GameEye frontend is a web application. An internet connection is required for accessing it while a mouse and keyboard or a touchscreen are required to interact with it. The frontend is displayable on most screen sizes, ranging from desktop screens to mobile device

screens. The frontend officially supports popular web browsers such as Google Chrome, Mozilla Firefox, Microsoft Edge, Opera, Brave, and Apple Safari.

2.3.4. Communications Protocols and Interfaces

GameEye utilizes standard TCP/IP, HTTP, and REST for web communication. SSL/TLS is used for securing communication with Firebase services. OAuth 2.0 and OpenID Connect are the protocols used for authentication and integration with third-party authentication backends, such as Google and Microsoft. Communication with MongoDB is done using a regular TCP/IP socket, while communication with ElasticSearch is carried out via its specialized Transport Client.

3. Product Requirements

The requirements for the GameEye prototype are included in a separate document titled “Lab 2 Section 3 – GameEye Product Requirements”. It includes the key functional requirements of the product and the necessary illustrations to expand on the concepts. Assumptions and constraints are also described in that document.