Lab 2 – GameEye Product Specification Outline

Jonathan Epps

Old Dominion University

CS 411W

Professor Janet Brunelle

October 18, 2020

Sections 1 & 2

Version 2.0

Table of Contents

List of Figures

List of Tables

1. Introduction

An existing societal problem is that gamers lack a software solution that automatically tracks news updates on developing video games. In 2015, there were almost two billion gamers worldwide. This figure is steadily increasing, as it is expected to rise to over three billion gamers by 2023 (Gough, 2020b). Thousands of video games are released every year. The online gaming platform Steam released 9,050 games in 2018 (Gough, 2020a). Video games also often have long development timeframes, which makes staying updated on numerous games for a long period of time tedious. For example, the popular game *Team Fortress 2* was in development for nine years before it was released in 2007 (Dietz, 2011). This societal problem is caused by inadequate game development monitoring solutions, decentralized news sources, lackluster news verification, and independent game developers having difficulty maintaining public attention. The current solution is too time-consuming to resolve the problem. The current solution is finding news on social media such as Reddit or Twitter; or finding news on a search engine such as Google.

The superior solution to the problem is to automate the process of searching for game information through a downloadable application. The application has a user-friendly interface, multifaceted accessibility, and customizability. It is a platform where users search for and follow games that intrigue them. This simplifies the process of searching for video game news because instead of going to various websites, the user will be able to access this platform to find the news that they desire. Users have their own personal watchlists to keep track of followed games.

1.1 Purpose

The solution to the problem is an application called GameEye. GameEye provides news notifications from multiple sources, a customizable notification system, self-organization, links to full sources, and a web application. GameEye monitors any game a user chooses to follow and will notify them when new content on a game they followed is released online. GameEye uses machine learning to classify news articles, videos, images, and updates. Machine learning also analyzes news articles, videos, images, and updates and determines whether a news article, video, image, or update is important to a specific user. GameEye is a web application that connects video game fans with news notifications about video games in development. A user sets up their account to monitor specified games in order to receive news and information about the games that they follow. Hardware and software are used to implement GameEye's key product features and capabilities.

GameEye offers key product features and capabilities such as prevalent characteristics, authentication, account management, game tracking, searching, web scraping, notifications, settings, and machine learning. Authentication includes secure user login and registration, external provider user login and registration with social media accounts, persistent sessions that do not require a user to login every time they open the app, two-factor authentication, which allows increased security, and password recovery mechanisms. Account management includes the ability for users to change their passwords, modify their profile information, and delete their accounts. Game tracking includes personal watchlists, which display thumbnails next to a game's title; new updates organized by important updates, news articles, images, and videos with thumbnails for news articles; and a list of the most-watched games by users. Searching includes the ability to search for video games based on their titles and support for autocompletion of a

search query. Images, videos from official game YouTube channels, and information from news articles from video game news websites are retrieved using web scraping. Users receive cross-platform push notifications for new game updates with a user interface (UI) count of notifications for each game and resource category and suggested video game notifications based on their most-watched games list. Users have the choice of seeing archived resources, receiving notifications and being able to choose which resource categories to be notified about, and submitting feedback. Game updates and extracting important information for news articles are classified using machine learning. The intended user community of GameEye is gamers who want to stay up to date on their favorite video games.

1.2 Scope

The goal of GameEye's prototype is to provide enough functionality to be able to show off what GameEye is capable of. GameEye includes general features, authentication, account management, searching, game tracking, settings, machine learning, and notifications. The features and capabilities of GameEye's prototype include robust user authentication and external provider authentication support, account management features for users to change their passwords and modify their profile information, game searching with autocompletion, personal game watchlists, web scraping for news articles, redirection to sources upon clicking scraped resources, and a most-watched games list. Other features include settings for toggling the notifications, classification of news articles using machine learning, determining whether news articles are important using machine learning, and cross-platform push notifications for new game updates and UI count of notifications.

The features that demonstrate GameEye's success are scraping data from multiple online sources and for multiple different resource categories such as news articles; offering robust

authentication mechanism for users; game searching and fast autocompletion; a personal watchclist; a list where users can see the most watched games on the platform; cross-platform support and push notifications; customization capabilities; a variety of settings regarding content and notifications; and using machine learning to classify news articles and whether they are important.

Risk mitigation involves scraping Really Simple Syndication (RSS) Feeds to protect against source website structure changes which would cause web scraping to fail; storing video game database content on GameEye's database for redundancy in case Internet Game Database (IGDB) goes down; making platforms scalable; using load balancers; and multiple instances of servers and databases to protect against high load. Risk mitigation also involves caching content of user devices so that the application does not appear blank in case the main database fails and having more instances of the main database for redundancy; using database encryption and a third party authentication provider to safeguard identifiable information; providing a manual to help users understand the application; and using proper coding practices to prevent NoSQL injection, XSS, data exposure, broken authentication, and access controls.

The prototype utilizes the same architecture as the real-world product for the replicated servers for load balancing that would be present in the real-world product. The prototype has many of the same features that the real-world product has. However, the prototype does not carry functionality for offline support; local caching; connection interruption resiliency; deleting accounts; web scraping for images, videos, Tweets, and Reddit posts; archived resources; submitting feedback; resource classification; important information extraction; and suggested video game notifications. The prototype includes the following features with partial functionality: searching for games, search autocompletion, adding games to a watchlist, web

scraping for news articles, resource thumbnails including website logos, game thumbnails, and source website redirection.

The challenges in developing the prototype include learning new technology and frameworks required for implementation, collecting and labeling data for machine learning models, properly securing communication between the multiple backends, implementing robust caching of frontend content for connectivity interruption resiliency and performance, and implementing robust web scraping with future-proof mechanisms in case source websites change.

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 creation of applications that access the features or data of an operating system, application, or other services.

AWS: Amazon® Web Services; Amazon® subsidiary that provides on-demand cloud computing platforms and APIs.

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.

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.

IGDB: Internet Game Database; database of known video games; accessed by 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.

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

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

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

JUnit Java 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.

PWA: Progressive Web Application; 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.

REST: Representational State Transfer; 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.

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

Spring Framework: 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.

1.4 References

Anderton, K. (2019, June 26). The business of video games: market share for gaming platforms in 2019 [Infographic]. *Forbes*. https://www.forbes.com/sites/kevinanderton/2019/06/26/the-business-of-video-games-market-share-for-gaming-platforms-in-2019-infographic/#54a8e8787b25/

Angelopoulous, A. Cook, J., Diasanta, C., Epps, J., Hund, B., Lewis, B., Wilson, A. (2020). *Lab 1 Outline*. Old Dominion University. https://www.cs.odu.edu/~411yello/pages/labs.html

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

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

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

Gough, C. (2020, September 28). *Google Play: Number of available gaming apps at Google Play from 1st quarter 2015 to 2nd quarter 2020*. Statista. 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. *PCMag*. 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*. Gamasutra.

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

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. 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 offers a detailed description of the software and hardware used for implementing the GameEye prototype, the features that the GameEye prototype provides including any limitations, and the interfaces that GameEye provides to other software.

2. GameEye General Description

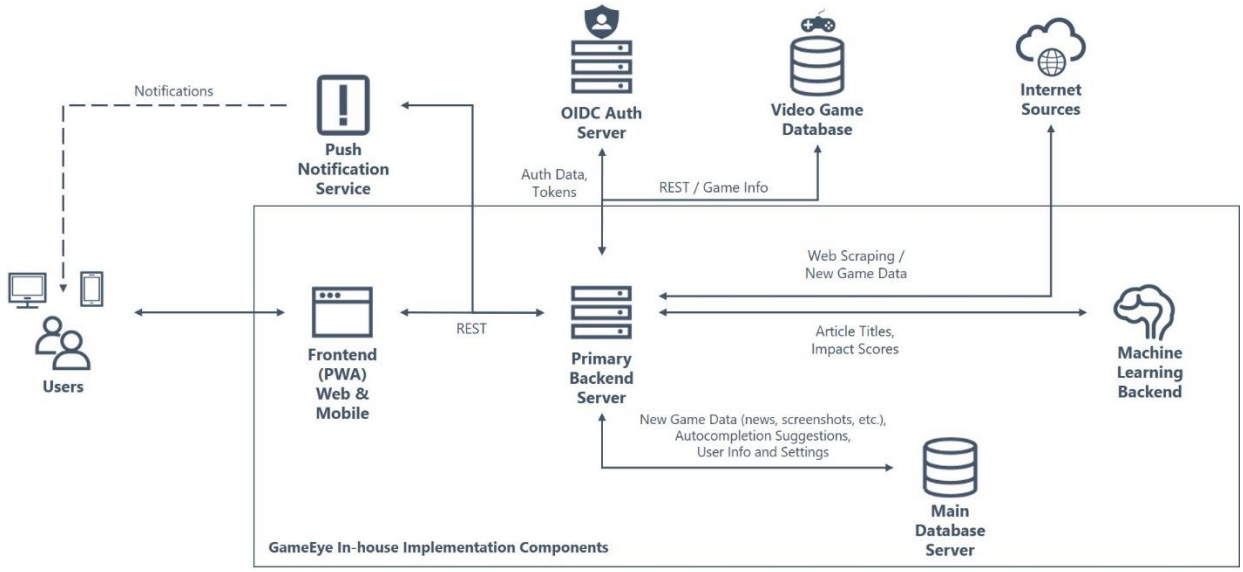
GameEye's prototype has many features that allow it to function. Each feature is an integral part of GameEye's prototype. The application uses many different external interfaces.

2.1 Prototype Architecture Description

GameEye is comprised of the following major components: fronted, primary backend, machine learning backend, main database, Elasticsearch database, Firebase Authentication, and Firebase Cloud Messaging. Cross-platform support on desktop and mobile devices based on progressive web application (PWA) technologies, offline support, local caching, and connectivity interruption resiliency features are prevalent characteristics. Figure 1 GameEye's Prototype Major Functional Component Diagram (MFCD) lists GameEye's features and capabilities as a prototype.

Figure 1

GameEye's Prototype Major Functional Component Diagram



Hardware and software make up GameEye's major components. Hardware consists of the frontend server, the main backend server, the web scraping backend server, the machine learning backend server, and the main database server. Software consists of frontend software, backing software, testing software, machine learning software, natural language processing software, databases, and third party software. WebStorm Integrated Development Environment (IDE) is a frontend software application that facilitates application development. IntelliJ IDEA IDE is a backend software IDE developed by JetBrains that allows developers to write Java applications. JUnit Java Framework is a testing framework for Java. Python is an interpreted, high-level, general-purpose machine learning programming language. spaCy library is an open-source library for advance natural language processing. MongoDB is a cross-platform

document-oriented database program. Firebase Authentication is third party software that provides backend services and ready-made UI libraries to authenticate users to an app.

The frontend is a cross-platform single-page web application built with Angular that provides the GameEye user interface and communicates with the primary backend using its Representational State Transfer (REST) Application Programming Interface (API) and Firebase services. The primary backend is a server built with Spring Boot that provides a REST API to the frontend and orchestrates web scraping. It communicates with the main database, Elasticsearch database, machine learning backend, IGDB, and Firebase servers. The machine learning backend is a server built with TensorFlow Serving that provides importance functionality and a REST API for the primary backend to interact with. The main database is a MongoDB database that stores all of the data related to GameEye. The Elasticsearch database stores game titles, allows for advanced and efficient searching on them, and powers GameEye's autocompletion functionality. Firebase Authentication is an authentication service provided by Google as part of the Firebase platform. Firebase Cloud Messaging is a push notification service provided by Google as part of the Firebase platform. This service is used to notify users of games that they are following.

2.2 Prototype Functional Description

GameEye's prototype design is comprised of backend and frontend components. The backend components include collecting metadata about video games from IGDB, storing video game information in MongoDB, supporting searching on video game titles using Elasticsearch, regularly scraping video game news websites and collecting news articles, analyzing scraped news article titles by determining which game they are referring to, removing articles that do not reference a specific game or that have already been scraped, determining whether news articles

are important using machine learning, notifying users when new articles have been scraped based on their preferences, providing a REST API to the frontend for communicating data, enforcing role-based endpoint authorization to prevent unauthorized users from accessing the backend, and providing utilities to developers and testers.

The frontend components include providing a page where users can login and create an account using an email and password combination or using their Google or Microsoft account, a page where users can see their watchlist, a page where users can add new games to their watchlist, a page where users can see the different resources available for a game, a page where users can see the news articles associated with a game, a page where users can change their password and account information, a page where users can change their settings; and notifying users when news articles have been scraped based on their preferences. Table 1 lists GameEye's features in a prototype state with a description for each feature.

Table 1

Prototype Features

Feature	Description	Implementation
General		
Cross-Platform Support (Desktop, Mobile)	Ability to use GameEye on desktop and mobile devices.	Full Functionality
Authentication		
User Login	Access an existing account on GameEye.	Full Functionality
User Registration	Create an account on GameEye.	Full Functionality

Feature	Description	Implementation
External Provider Login & Registration	Login with a Google or Microsoft account. If logging in for the first time on GameEye, an account is automatically created.	Full Functionality
Persistent Sessions	Access account without having to log in again after closing GameEye without logging out and reopening it.	Full Functionality
Password Recovery	Send a link to the user's email address that allows them to reset their password.	Full Functionality
Account Management		
Change Profile Information	Allow users to change their name and email address.	Full Functionality
Change Password	Allow users to change their password while logged in.	Full Functionality
Searching		
Search for Games	Allow users to search for video games to add to their watchlist.	Partial Functionality: searching mechanisms will be fully functional but not all games will be available
Search Autocompletion	Search results appear based on characters typed in the watchlist search bar (e.g. Hollow Knight and Hollow Knight: Silksong appear when "Hollow Kn" is typed in the search bar).	Partial Functionality: autocompletion mechanisms will be fully functional but not all games will be available

Feature	Description	Implementation
Game Tracking		
Add Games to Watchlist	Allow users to add games to their watchlist.	Partial Functionality: Not all games will be available
Remove Games from Watchlist	Allow users to remove games from their watchlist.	Full Functionality
News Articles (Web Scraping)	Web scraping used to obtain news articles about video games by scraping popular gaming news websites.	Partial Functionality: not all intended news websites will be scraped
Resource Thumbnails (Includes Website Logos)	Display thumbnails for various resources such as news articles. These are scraped images. News website logos are also collected and displayed next to articles.	Partial Functionality: not all resources will be implemented, only news articles. Not all news websites will be used.
Game Thumbnails	Display images that represent a game. Shown in the watchlist, title bar, and search results.	Partial Functionality: not all games will be available
Source Website Redirection	Redirect a user to the official news article page when clicking on a news article inside GameEye.	Partial Functionality: not all news websites will be available
Resource Organization	The various resources will be organized by the type (e.g. news articles, tweets, etc.).	Partial Functionality: only news websites will be available
Most-Watched Games List	A list of the most-watched games by GameEye users.	Full Functionality

Feature	Description	Implementation
Settings		
Receive Notifications Option	Users can choose whether or not they want to receive notifications.	Full Functionality
Machine Learning		
Importance	Machine learning is used to determine whether or not a resource is important.	Partial Functionality: only for news articles, not enough training data to get the desired accuracy
Notifications		
Push-Notifications for New Resources	Users will receive push-notifications when new resources have been scraped.	Partial Functionality: only for news articles
UI Count of Notifications for Each Game	An indicator showing how many unseen notifications there are for each game in a user's watchlist.	Full Functionality
UI Count of Notifications for Each Resource Category	An indicator showing how many unseen notifications there are for each resource category.	Partial Functionality: only for news articles
Cross-Platform Notifications	Notifications will be received in both the desktop and mobile versions of GameEye.	Full Functionality

Half of the GameEye prototype’s features have full functionality while the other half only have partial functionality.

2.3 External Interfaces

GameEye's prototype makes use of many different external interfaces, namely hardware interfaces, software interfaces, user interfaces, and communications protocols and interfaces.

2.3.1 Hardware Interfaces

The hardware interfaces that GameEye's prototype uses are the frontend server, the main backend server, the web scraping backend server, the machine learning backend server, and the main database server. The frontend utilizes Angular CLI server to run the web application. The main backend utilizes Java class MainBackendApplication. The web scraping backend utilizes Java files MockNewsScraper, UniversalScraper, and WebScraper. The machine learning backend utilizes Java class MachineLearningService. The main database utilizes a MongoDB server.

2.3.2 Software Interfaces

The software interfaces that GameEye's prototype uses are MongoDB, Elasticsearch, IGDB, Firebase Authentication, and Firebase Cloud Messaging. MongoDB is a cross-platform document-oriented database program. Elasticsearch is a database that stores game titles, allows for advanced and efficient searching on them, and powers GameEye's autocompletion functionality. IGDB is a database of known video games that is accessed through a REST API in order to populate GameEye's database. Firebase Authentication is an authentication service provided by Google as part of the Firebase platform that provides backend services and ready-made UI libraries to authenticate users to an app. Firebase Cloud Messaging is a push notification service provided by Google as part of the Firebase platform.

2.3.3 User Interfaces

The user interfaces that GameEye's prototype uses are a monitor for providing the graphical user interface (GUI), keyboard for data entry, and a mouse for maneuvering and selecting.

2.3.4 Communications Protocols and Interfaces

The communications protocols and interfaces that GameEye's prototype uses are REST API and a set of administrative IGDB endpoints that connect with IGDB game data. REST API is a set of functions and procedures that allows GameEye to access the features or data of an operating system, application, or other services, such as IGDB, that makes use of a software architectural style used in creating web services. The specific IGDB endpoints that GameEye uses are the IGDB Replication Endpoint that accepts maximum and minimum IGDB IDs to populate GameEye's database with IGDB game data for games within the minimum-maximum range and the Game Endpoints that retrieve game information such as game logos and news articles pertaining to games.