Lab 1 – GameEye

Jonathan Epps

Old Dominion University

CS 411W

Professor Janet Brunelle

September 12, 2020

Version 1.0

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1 Societal Problem

An existing societal problem is that gamers lack a software solution that automatically tracks news updates on developing video games. In 2014, there were almost 1.82 billion gamers worldwide. This figure is steadily increasing, as it is expected to rise over 2.7 billion by 2021 (Gough, 2019). Thousands of video games are released every year. The online gaming platform Steam released 9,050 games in 2018 (Gough, 2019). Video games also often have long development timeframes, which makes it tiresome to stay updated on numerous games for a long period of time. 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.

1.2 Solution Description

The superior solution to the problem will be to automate the process of search for game information through a downloadable application. The application will have a user-friendly interface, multifaceted accessibility, and customizability. It will be a platform where users search for and follow games that they may be interested in. This will simplify 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 are looking for. Users will even have their own personal watchlists to keep track of followed games.

1.3 Solution

The solution to the problem is an application called GameEye. GameEye will provide news notifications from multiple sources, a customizable notification system, self-organization, links to full sources, and a web application. GameEye will monitor any game a user chooses to follow and will notify them when new content on a game they followed is released online. GameEye will use machine learning to classify news articles, Reddit posts, videos, images, and updates. Machine learning will also analyze news articles, Reddit posts, videos, images, and updates and assign a multi-factor score depending on their impact on a specific user.

2. GameEye Product Description

GameEye is a web application that will connect video game fans with news notifications about video games in development. A user will set up their account to monitor specified games in order to receive news and information about the games that they follow. GameEye will have many key product features and capabilities that will be implemented via various hardware and software.

2.1 Key Product Features and Capabilities

The key product features and capabilities of GameEye are prevalent features, authentication, account management, game tracking, searching, web scraping, notifications, settings, and machine learning. Prevalent features include cross-platform support on desktop and mobile devices based on progressive web application (PWA) technologies, offline support, local caching, and connectivity interruption resiliency features. Authentication includes secure user login and registration, external provider user login and registration (e.g. 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 Reddit posts, images, and videos with thumbnails for news articles; and a list of the most-watched games by users. Searching includes support for autocompletion of a search query. Web scraping involves going online and retrieving images, videos from official game YouTube channels, and information from news articles from video game news websites and Reddit posts from official game subreddits. Notifications include cross-platform push notifications for new game updates, a user interface (UI) count of notifications for each game and resource category, and suggested video game notifications based on a user's most-watched games list. Settings include the option to see archived resources, see impact scores for news articles, receive notifications and being able to choose which resource categories to be notified about, and submit feedback. Machine learning is used to classify game updates, assign impact scores for news articles, and extract important information for news articles.

2.2 Major Components

The major components of GameEye include hardware and software. 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. An example of frontend software is WebStorm Integrated Development Environment (IDE) which is an application that facilitates application development. An example of backend software is IntelliJ Idea IDE which is and IDE developed by JetBrains that allows developers to write Java applications. An example of testing software is

JUnit Java Framework which is a testing framework for Java. An example of machine learning software is Python which is an interpreted, high-level, general-purpose programming language. An example of natural language processing software is spaCy library which is an open-source library for advance natural language processing. An example of a database is MongDB which is a corss-platform document-oriented database program. An example of third party software is Firebase Authentication which provides backend services and ready-made UI libraries to authenticate users to an app. Figure 1 Major Functional Component Diagram (MFCD) lists how GameEye obtains and distributes the information that it receives.

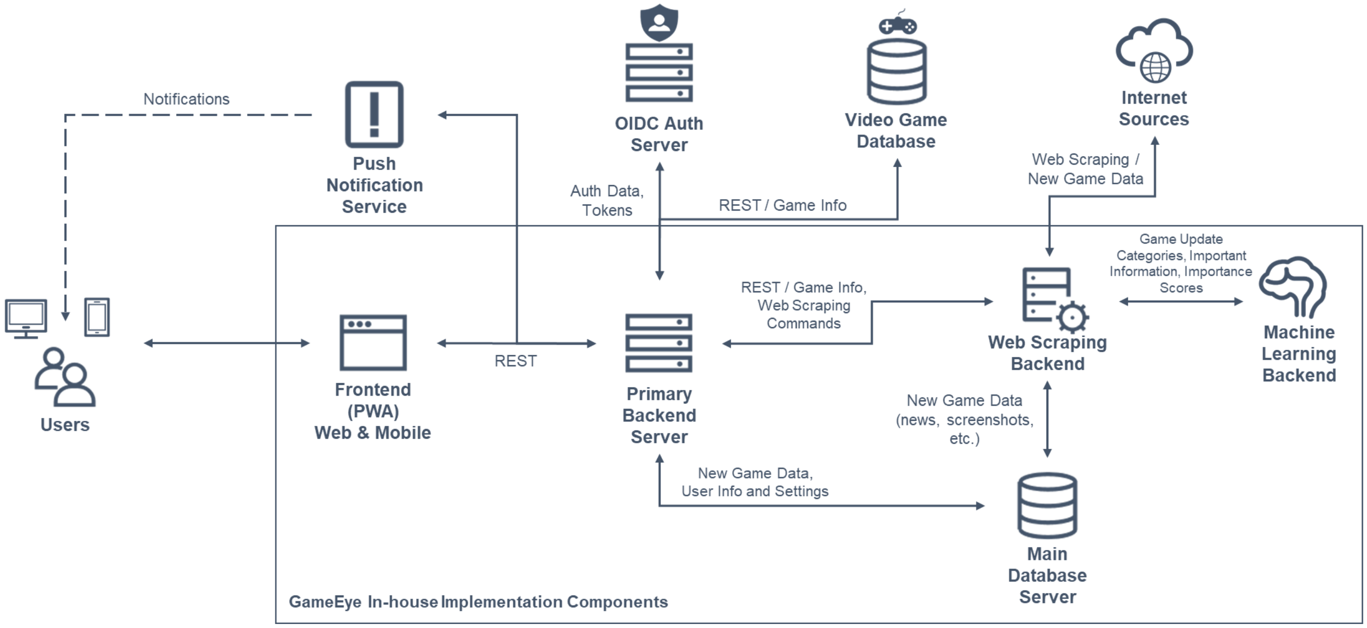
Notifications
Push Notification Service
OIDC Auth Server
Video Game Database
Internet Sources
Auth Data, Tokens
REST / Game Info
Web Scraping / New Game Data
Game Update Categories, Important Information, Importance Scores
REST / Game Info, Web Scraping Commands
Users
Frontend (PWA) Web & Mobile
REST
Primary Backend Server
Web Scraping Backend
Machine Learning Backend
New Game Data (news, screenshots, etc.)
New Game Data, User Info and Settings
Main Database Server
GameEye In-house Implementation Components

Figure 1: GameEye's Major Functional Component Diagram.

3. Identification of Case Study

GameEye is initially provided to members of the ODU Gaming Club and other local users before its widespread release for gamers who want to stay notified on upcoming games that they are excited for and want to keep track of games that have upcoming releases or already released. People that want to become gamers in the future may be interested in the app.

4. GameEye Product Prototype Description

4.1 Prototype Features and Capabilities

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, a most-watched games list; settings for toggling the visibility of archived resources, impact scores, and notifications; classification of news articles using machine learning, scoring the impact of news articles using machine learning, and cross-platform push notifications for new game updates and UI count of notifications. The features that demonstrate that GameEye successfully 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, and a variety of settings regarding content and notifications; and using machine learning to classify news articles and their impact scores. 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 our database for redundancy in case IGDB goes down; making platforms scalable, using load balancers, and multiple instances of servers and databases to protect against high load; 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.

4.2 Prototype Architecture

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. Table 1 lists GameEye's features in a prototype state and as a real-world product.

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 Password	Full Functionality	Full Functionality
Delete Account	Full Functionality	No Functionality
Searching		

Feature	**RWP**	**Prototype**
Search for Games	Full Functionality	Partial Functionality
Search Autocompletion	Full Functionality	Partial Functionality
Game Tracking		
Add Games to Watchlist	Full Functionality	Partial Functionality
News Articles (Web Scraping)	Full Functionality	Partial 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 (Includes Website Logos)	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	Full Functionality
Show Impact Scores Option	Full Functionality	Full Functionality
Receive Notifications Option	Full Functionality	Full Functionality
Receive Notifications Per Category Option	Full Functionality	Full Functionality
Submit Feedback	Full Functionality	No Functionality

Feature	RWP	Prototype
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

Table 1: RWP vs Prototype Features.

4.3 Prototype Development Challenges

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.

5. Glossary

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; application that facilitates application development.

IGDB: Internet Game Database; database of known video games; accessed by REST API to populate GameEye's database.

Impact Score: A score from 1-3 on the impact some news has on a game and its players; it is computed using machine learning.

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 back-end development of GameEye.

JavaScript: Object-oriented language used to create dynamic, interactive effects on webpages.

Jest JavaScript Framework: Testing framework maintained by Facebook Inc.

JSoup Library: Java library for working with real-world HTML; extracting and manipulating data.

JUnit Java Framework: A testing framework for Java.

Keras (Python Deep Learning Library): Open-source neural-network library written in Python.

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

Noise Filtering: Shown data catered to an individual's content preferences.

OIDC Authentication: 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.

Python: 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; **w**eb feed that allows users and applications to access updates to websites in a standardized, computer-readable format.

Scikit-learn Library: Software machine learning library for the Python programming language.

SpaCy Library: An open-source software library for advanced natural language processing.

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

Tester: GameEye beta testers; users of the application in its prototype phase who will provide feedback on their experience.

Web Scraping: Data scraping for extracting data from websites.

WebStorm: IDE developed by JetBrains to write JavaScript code.

6. References

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

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

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

Gough, C. (2019, October 9). *Google Play: Number of available games by quarter 2019*. Statisa. 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