Lab 1 - GameEye Product Description

Christopher Diasanta

Old Dominion University

CS 411W

Professor Janet Brunelle

6 October 2020

Version 2.0

Table of Contents

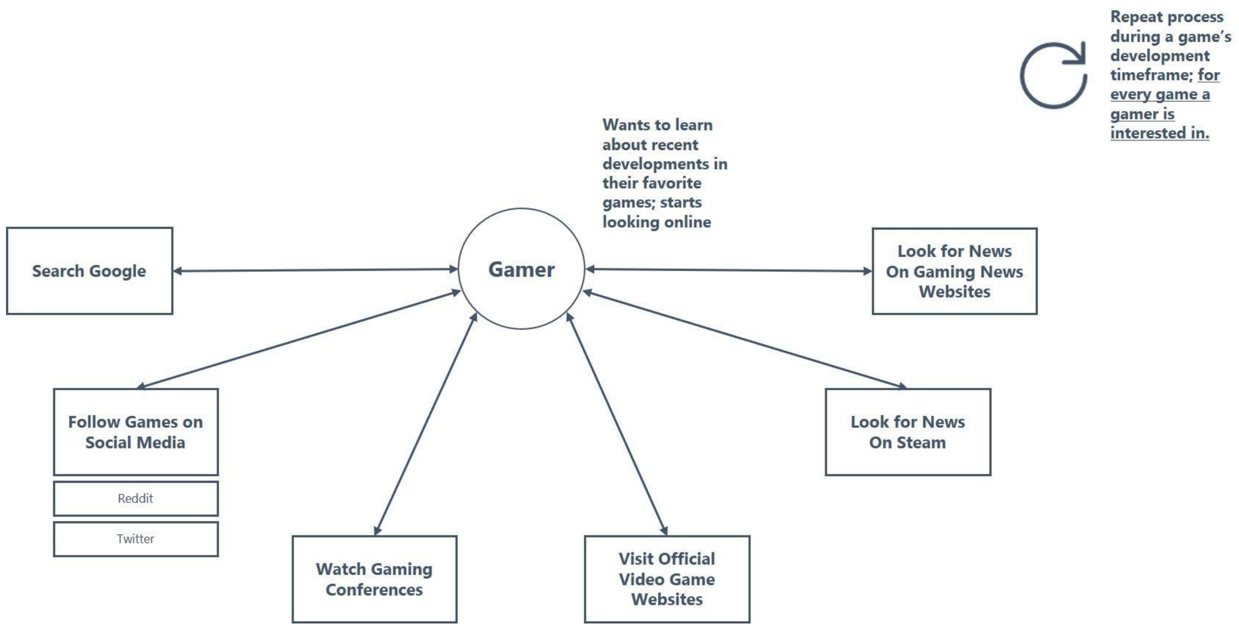
Table of Figures

1. Introduction

In recent years, there has been a surge in the amount of video games released. These video games involve long development times, so staying up-to-date about videogames and sorting through information from game developers can be time-consuming and challenging for gamers. In 2018 alone, over 9,000 video games were released on the popular video game distribution platform, Steam (Gough, 2019). There is a lack of a convenient way for gamers to keep up-to-date about developments on the vast amount of video games since information about video games is decentralized. Independent game developers have difficulty maintaining public attention because of a lack of centralized information. Figure 1 shows the current process flow a gamer might follow to stay up-to-date on the latest video games.

Figure 1

Current process flow diagram describing means of gamer attaining information.


Repeat process during a game's development timeframe; for every game a gamer is interested in.
Wants to learn about recent developments in their favorite games; starts looking online
Search Google
Gamer
Look for News On Gaming News Websites
Follow Games on Social Media
Reddit
Twitter
Look for News On Steam
Watch Gaming Conferences
Visit Official Video Game Websites

A gamer may want to search about recent developments in one of their favorite video games, so they may start looking online for news. They might search Google, follow games on social media, watch game conferences, visit official video game websites, look for news on Steam, or look for news on gaming news websites. A gamer would repeat this process for each game of interest. Some of these news sources, such as social media, may be unverified or spread false information. Some independent game developers may have difficulty maintaining public attention as a result.

Gamers need an easier way to gather the information they. The solution to this problem is GameEye. GameEye is a platform in which users can track games. GameEye users receive notifications of news articles or new content like gameplay on video games they follow. Users have a watch list in which they receive information about current and unreleased games they may be interested in such as potential release date delays and news updates. The news articles and tweets are classified based on the content.

2. GameEye Product Description

GameEye is a platform that will provide gamers an easy way to follow up on their favorite games. It aggregates news information about developing video games, and users receive notifications on the new information. GameEye centralizes videogame news for users to access.

2.1 Key Product features and Capabilities

GameEye will be cross-platform progressive web application working on both mobile and desktop devices. Users will be able to use the app offline as it provides offline support through local caching.

Users will be able to securely login and register. An external provider will be utilized for user login and registration. The use of an external provides allows users to login or register using social media like their Google or Microsoft account. GameEye will utilize persistent sessions allowing the user ease of access as they do not need to login to their account every time. Two-factor authentication may be used as well for increased security. In case a user may forget their password, a password recovery mechanism will be used to recover it. Users may change their password, modify their profile information, and delete their account.

GameEye will provide a personal watchlist or list of video games an individual wants to stay informed about. In the watch list, a thumbnail of the game is displayed next to the title. The list will show the new game updates in the form of a user interface(UI) count. The watchlist will organized by important updates, news articles, tweets, Reddit posts, images, and videos. Thumbnails are included for tweets and news articles. A separate list of games featuring the most-watched games on the platform is provided in order to find popular game titles. Users may search for video games based on their titles, and the search system provides will support auto completion of the video game titles.

Multiple sources will be aggregated to provide game development updates. GameEye utilizes web scraping in order to aggregate news articles from video game news websites, tweets from official developer Twitter feeds, Reddit posts from official developer game subreddits, videos from game developer YouTube channels, as well as images. These different sources will be categorized on the watchlist by videos, images, or even tweets.

Cross-platform push-notifications will provide users with new game updates. A user interface (UI) count of notifications is displayed for each game and each resource category. Users are notified of suggested video games to follow based on the most-watched games list.

Users will be provided with options for their GameEye experience. Users may choose if they wish to see archived resources, see the importance scores of news articles and tweets, choose to receive notifications, and which categories to be notified about.

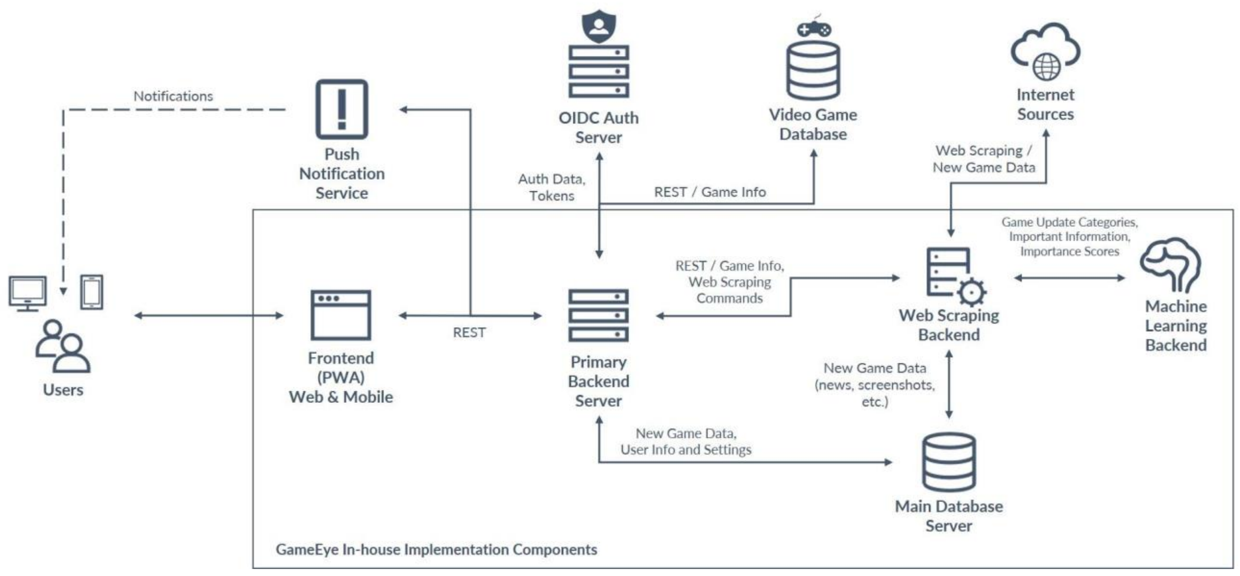
Machine learning will be implemented in order to provide game update classification such as release date announcement, delay, major game update, or minor game update. Importance scores are assigned for news articles and tweets utilizing machine learning. Important information is extracted from news articles and tweets.

2.2 Major Components

GameEye utilizes many components: a frontend server, main backend server, web scraping backend server, machine learning backend server, and a main database server. Figure 2 shows how the components will communicate to deploy the web application. The objects within the square are GameEye's in-house implementation components. Objects outside the square are the external components such as the push notification service, video game database, internet sources, and OIDC authorization server.

Figure 2

GameEye's Major Functional Component Diagram


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

The frontend server provides the frontend of the web application for the user to access. It runs the user interface of the web application. The main backend server will expose a REST API for communication with the frontend, web scraping backend, databases, and the other third-party services. It will coordinate web scraping backends and power the dynamic functionality of the frontend. The web scraping backend server is also an external-facing web server, and it exposes a REST API for launching web scraping operations. This server performs the web scraping of information from news sources. It communicates with the machine learning backend to classify scraped video game data and extract important information before placing it in the database. The machine learning backend server is an internal GPU-powered TensorFlow serving server will expose a REST API for machine learning inference. It communicates with the web scraping backend to classify video game data and extract important information. The main database server is an internal MongoDB database server that stores all data related to the application. It will store information related to the users, news websites, games, images, and archived resources

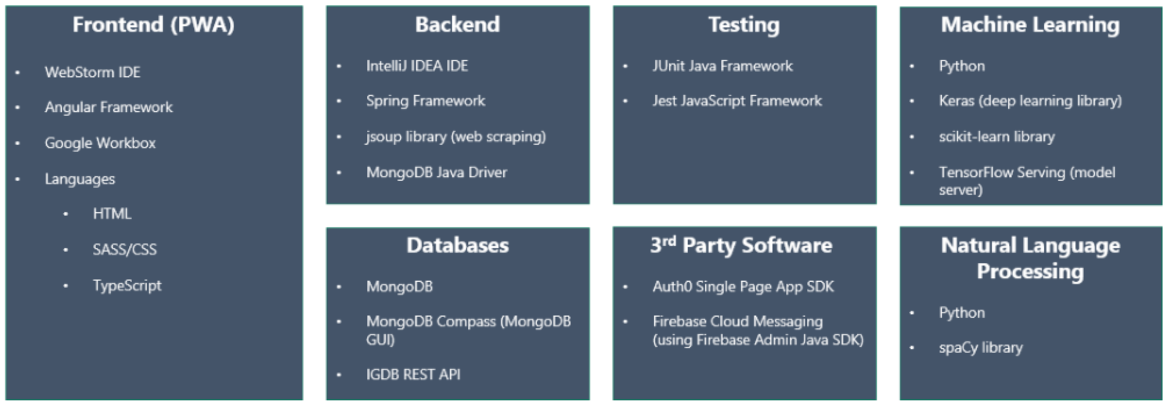
The frontend of GameEye is developed using WebStorm IDE and Angular FrameWork to build the UI. Google Workbox is implemented to provide GameEye with offline support. The frontend of GameEye is written in HTML, SASS/CSS, and typeScript. This will provide the necessary components to create the frontend web application.

The backend of GameEye is developed using IntelliJ IDEA IDE and Spring Framework. The Jsoup library is used to provide the web scraping capabilities, and MongoDB Java Driver is used to build the database. Together these software components will provide the game content needed for the frontend application.

Figure 3 lists the softwares that will be used in the development of GameEye.

Figure 3

GameEye's Software Components Diagram



Testing of GameEye will be provided by utilizing JUnit Java Framework to test the Java code and Jest JavaScript Framework to test the JavaScript. The JUnit Java and Jest JavaScript frameworks will be used to provide unit tests to the backend logic and testing for the frontend application.

GameEye uses Python code to implement the machine learning aspect and is hosted using the Tensorflow serving. The Keras and Scikit-learn Python libraries is used to code the machine learning aspects. It will provide the necessary components to implement the assigning of impact scores to the articles.

Natural language processing is a part of GameEye as it allows the web scraped information to be categorized. The Python spaCy library is used to perform the natural language processing. Important information, such as game delays or announcements, will be extracted and highlighted to the user.

Databases are constructed using MongoDB and MongoDB Compass. The database is populated using the IGDB REST API to retrieve data on known video games. Web scraped content will be stored onto this database and will be retrieved for the front end.

Third party software such as Firebase Authentication is used to provide the two-factor authentication in case a user wishes for a more secure login procedure. Firebase Cloud Messaging is utilized to provide the users with push notifications to their desktop or mobile device.

3. Case Study

GameEye is being developed with gamers in mind. In order for GameEye to be refined for the mass market, it will need to be tested by a smaller group. GameEye will be first deployed for use by the ODU Esports Club. Students from the thirteen different varsity teams can test the web application and provide feedback such as user experience and noise filtering. Students partaking in the case study will submit an anonymous survey about their GameEye experience. From there GameEye could be deployed to a greater group of gamers. Any gamer from experienced to novice could use GameEye. Stakeholders such as video game news sites and video game developers may also use GameEye in the future. Video game news sites could see an increase in web traffic because GameEye links the articles to the source websites and displays a news site’s logo next to each scraped article. Video Game developers may have an easier time maintaining interest around their developing games because of how saturated video game selections have become.

[THIS SPACE INTENTIONALLY LEFT BLANK]

4. Product Prototype Description

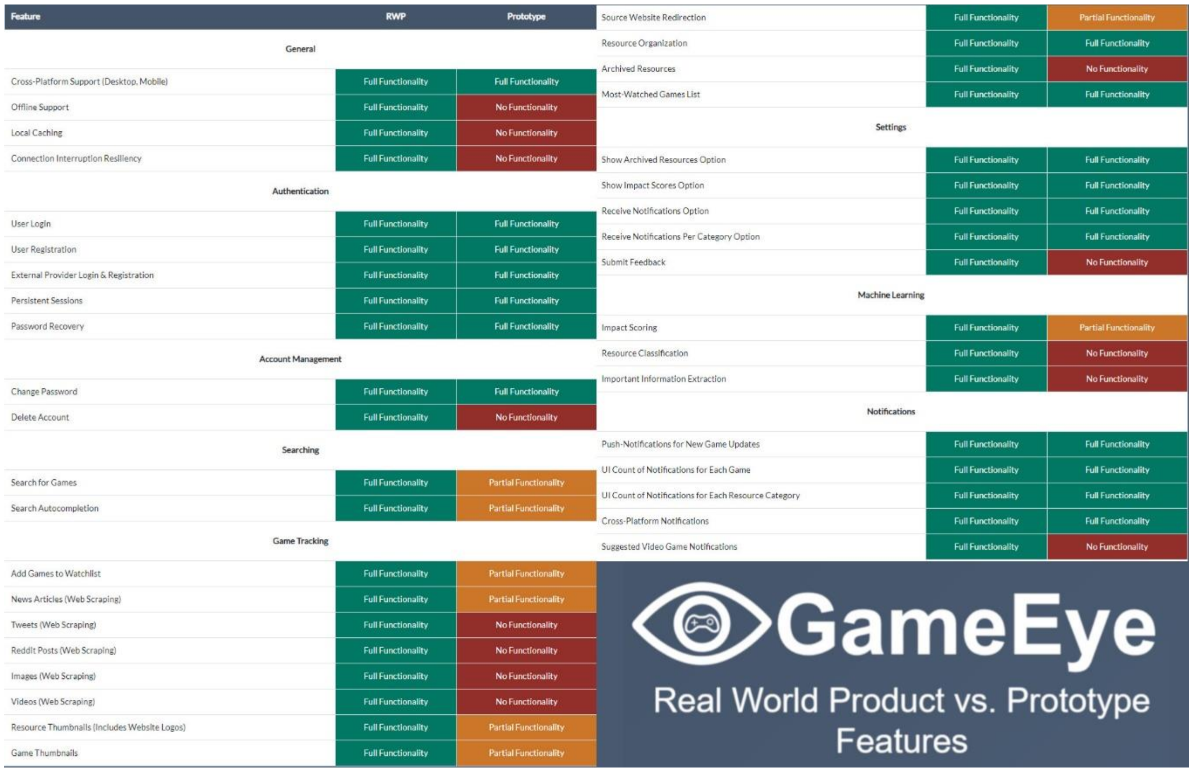
The GameEye prototype is still a progressive web application, but features and architectures are omitted from the real world product.

4.1 Prototype Feature and Capabilities

Although the real world product is ambitious, the prototype will still provide various features that the real world product will deploy. Figure 4 shows features that will be implemented in the prototype.

Figure 4

GameEye's RWP vs. Prototype table



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		
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
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 (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
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

GameEye
Real World Product vs. Prototype Features

The prototype will provide cross platform support between desktop and mobile. Authentication features will be fully functional, it will be able to provide a robust user authentication and external provider support for logging in. Necessary account management features will be implemented for users to change their passwords and modify their profile information. When searching for games, there will be an auto-completion feature. Users can add games to a personal watchlists. The GameEye prototype will provide redirection to sources upon clicking scraped resources. A most-watched games list will be implemented to promote popular games on the GameEye platform to users. GameEyes's settings will provide the availability to toggle visibility of archived sources, visibility of importance scores, and notifications. The prototype will have partial functionality for classification and impact scoring of news articles using machine learning. The GameEye prototype will feature cross-platform push-notifications and UI count of notifications for new game updates.

The provided features will demonstrate that GameEye can successfully do many tasks. It will be able to scrape data from multiple online sources and different resource categories. It will successfully provide robust authentication mechanisms for users to login, robust game searching and fast auto-completion, a personal watchlist of games for each user, a list of most watched games on the platform, customization capabilities regarding content and notifications, and cross-platform support and push-notifications. The prototype will demonstrate that it can use machine learning to classify news articles and tweets as well as score their importance.

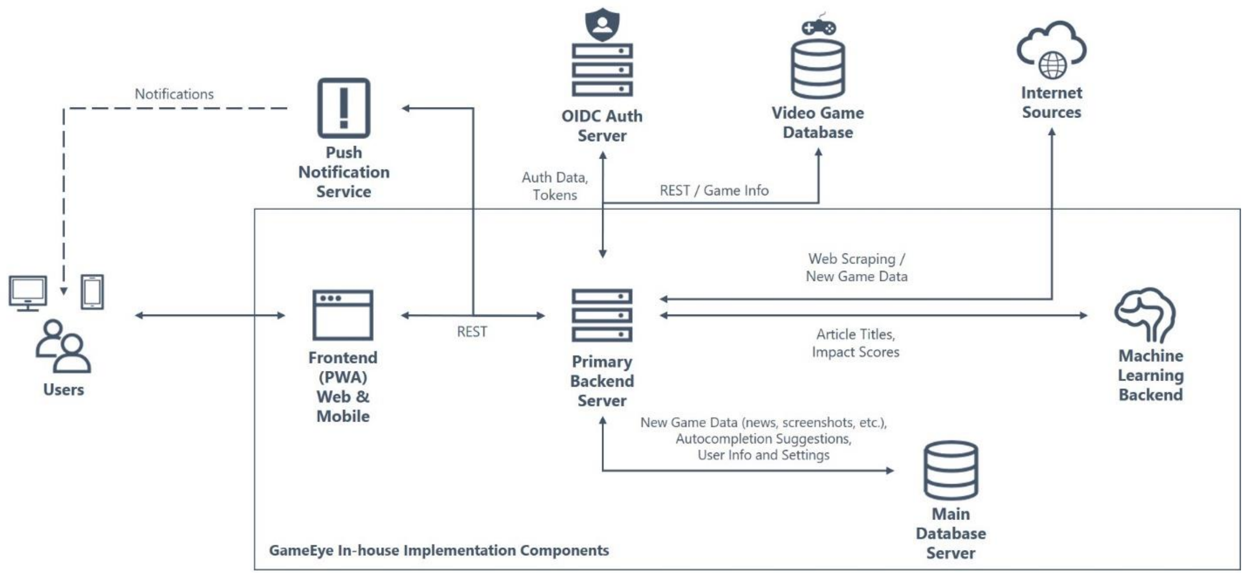
There are some risks involved in developing GameEye. News websites may change their structure causing webscraping to fail. To mitigate this RSS feeds will be scraped as website structure stays static. GameEye will store video game database content on its own database for redundancy. In the event that IGDB goes down, GameEye will still have its own database. Servers may experience high loads, so the platform will be made scalable to protect against high load. User content would be cached on a user's device to prevent the application going blank in case the main database fails. The platform will use database encryption and third party authentication providers to safeguard user's personally identifiable information. A manual will be provided to help users understand the application and usage. When developing both the frontend and backend, proper coding practices will be used to prevent SQL injections, cross-site scripting, or data exposure.

4.2 Prototype Architecture

The prototype will utilize the same architecture as the real-world product with a few exceptions. Figure 5 shows the major functional components of the prototype.

Figure 5

GameEye Prototype's Major Functional Component Diagram


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

The prototype will not use replicated servers for load balancing. The primary back end server will perform the web scraping instead of a dedicated server as in the real world product. Implementing the same architecture as the real world product will prove to be costly.

4.3 Prototype Development Challenges

There are challenges to overcome in developing the prototype. Knowledge of new technologies and frameworks will be required for implementation. Data will need to be collected and labeled for the machine learning models. There may be complications for properly securing communication between the multiple backends. Challenges may arise in implementing robust caching in the frontend content for connectivity interruption resiliency and performance. There might be issues with creating implementing future-proof, robust web scrapers because source websites may change.

[THIS SPACE INTENTIONALLY LEFT BLANK]

5. Glossary

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.

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

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.

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 OAuth2.0 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; Software architectural style used in creating web services.

RSS Feed: Really simple syndication; Web 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: 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.

[THIS SPACE INTENTIONALLY LEFT BLANK]

6. References

Anderton, K. (2019, June 26). Video game market share and growth. [Infographic]. *Forbes* 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*. MetatCritic https://www.metacritic.com/feature/games-that-shed-vaporware-status

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

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

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