Lab 2 - GameEye Product Specifications

Christopher Diasanta

Old Dominion University

CS 411W

Professor Janet Brunelle

18 October 2020

Version 1.0

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. 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 of the tedious routine.

1.1 Purpose

Gamers need an easier way to gather the information they need. The solution to this problem is GameEye. GameEye is a platform in which users can track games. It aggregates information by scraping from different sources such as news articles, Tweets, Reddit posts, images, and videos. 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 given an impact score, classified, and information is extracted using machine learning algorithms. Information is not scraped in real-time but rather scraped regularly every day. GameEye targets gamers of different backgrounds from experienced to casual gamers. GameEye centralizes video game news for users to access; it does not replace news sources or other media. It aggregates news information about developing video games and sends notifications about the new information. Articles are linked directly to the original source of information.

1.2 Scope

GameEye is a progressive web application that aggregates news and uses machine learning to deliver relevant content to a user. GameEye will aggregate video game news by scraping articles from video game news site's RSS feeds. These articles are organized based on the game being referenced, classified based on content, such as release or delays, and an impact score is calculated for each article using machine learning. GameEye provides a personal watchlist to users, in which they may search and add games to the list. Users will receive notifications for new content of video games in their watchlists.

GameEye's prototype features a subset of functionality in comparison to the real-world product. The prototype does not have full functionality for news aggregation. It scrapes news articles directly from news sources but does not scrape Tweets or Reddit Posts. The scraped articles are assigned an impact score. The prototype does not have content archival and the database has limited games due to hardware limitations. GameEye's prototype lacks some features but it demonstrates the real-world product.

1.3 Definitions, Acronyms, and Abbreviations

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]

1.4 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

1.5 Overview

This product specification provides a detailed description of the software and hardware components, capabilities, external interfaces of the GameEye prototype. The remaining sections of this document provides an in-depth description of the software and hardware configurations used to implement the GameEye prototype, features and limitations of the prototype, and other interfaces the prototype will provide to other software.

2. GameEye Product Description

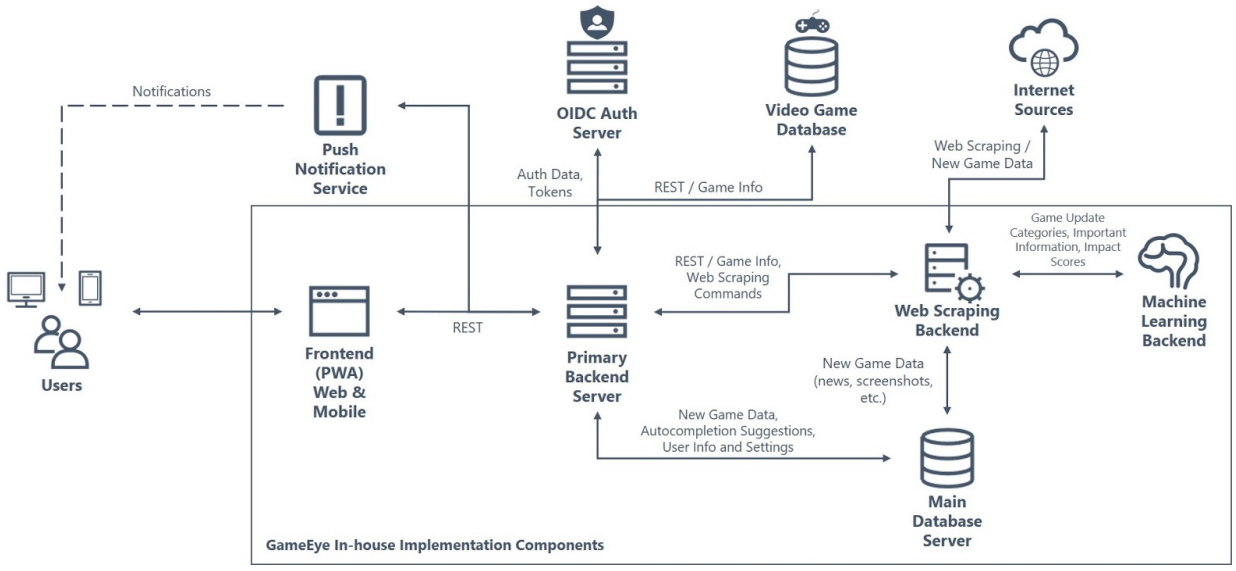
GameEye's prototype encompasses a robust architecture to provide functionality to demonstrate the real-world product. It interfaces with external software dependencies to provide functionality for the prototype.

2.1. Prototype Architecture Description

The prototype contains different components in its architecture. Figure 1 displays the major functional components in GameEye's prototype implementation.

Figure 1

GameEye's Major Functional Component Diagram



The prototype contains a frontend that serves the single-page web application for the backend. It communicates with the backend using its REST API. It also communicates with Firebase services to provide the Firebase Authentication to allow users to login to the application. Push-notification services are provided using Firebase Cloud Messaging. It is built using Angular. The primary backend server provides a REST API to the frontend. This component orchestrates the webscraping and communicates with the main database, ElasticSearch database, machine learning backend, Internet Game Database, and Firebase servers. It is built with Spring Boot. The machine learning backend server provides essential functionality to web scraping articles. It assigns impact scores to the webscraped articles. This component provides a REST API to interact with the backend and is built using TesnsorFlow Serving. GameEye's main database is another component and this stores all related information to GameEye such as articles and users. It is a MongoDB database. Another database in GameEye is the ElasticSearch database. This database stores game titles and allows for advanced and efficient searching on them. This database provides the auto-completion functionality in searching for games. These components come together to provide functionality to the prototype.

2.2. Prototype Functional Description

The backend and frontend consists of different components to provide the functionality of the GameEye web application. The backend provides the necessary functionality for the data, whereas the frontend provides the UI functionality for the user. Figure 2 below shows the prototypes functionality.

Figure 2

Prototype Features Table

Prototype Features Table

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
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
	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 they 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
	Settings	
Show Impact Scores Option	Users can choose whether or not they want to see the impact scores of a resource.	Partial Functionality: only for news websites
Impact Score Levels Option	Users can choose for which impact scores to receive notifications for.	Partial Functionality: only for news websites
Receive Notifications Option	Users can choose whether or not they want to receive notifications.	Full Functionality
	Machine Learning	
Impact Scoring	Machine learning is used to give a score to a resource based on how impactful it is.	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

The prototype's backend collects metadata about video games from IGDB and stores the video game information in MongoDB. This redundancy ensures that GameEye will still be functional in the event that IGDB connection is lost. The backend provides support for searching video game titles using ElasticSearch. It regularly scrapes and collects news articles from video game news sites. These articles are then analyzed to determine which video games are being referred to. Irrelevant or previously scraped articles are removed from the list of collected articles. The backend will send the articles to the machine learning backend to compute impact scores. Users will be notified through push notifications when new articles are scraped based on their preferences. To communicate with the frontend, the backend provides a REST API for communicating data. The backend enforces role-based endpoint authorization to prevent unauthorized users from accessing the backend. The backend supports many features, but the frontend provides the user functionality.

GameEye's frontend is the user interface between the user and the backend. Users access GameEye through a login page where they may login and create an account using email/password or using a Google/Microsoft account. The frontend contains different pages which users can access. Users have a page which they can access their watchlists and a separate page to add new games to their watchlist. Users can see differences available for a game such as articles when accessing their watchlist. Those articles will be displayed on a page and a link to the article will be displayed. Users have the ability to change some settings such as account information or password in the settings page. Users are further notified of newly scraped articles through a user interface count on the watchlist.

2.3. Software Interfaces

Many software interfaces are used in GameEye's prototype. MongoDB is used to build the prototype's database which stores data related to the application such as user information and scraped articles. ElasticSearch is an important dependency as it provides the auto-completion feature in game searching and database to find games being referenced in articles. IGDB provides an API in which GameEye uses to clone the video game database. It stores relevant information to a game such as title, genres, and a thumbnail of the game. FireBase Authentication provides the secure login and security features of GameEye. It allows users to create accounts using emails or Google/Microsoft accounts. Firebase Cloud Messaging provides the prototype the ability to send push notifications. These push notifications notify users of new relevant articles to a particular game.