Lab 1- GameEye Product Description

Christopher Diasanta

Old Dominion University

CS 411W

Professor Janet Brunelle

4 May 2020

Version 1.0

Table of Contents

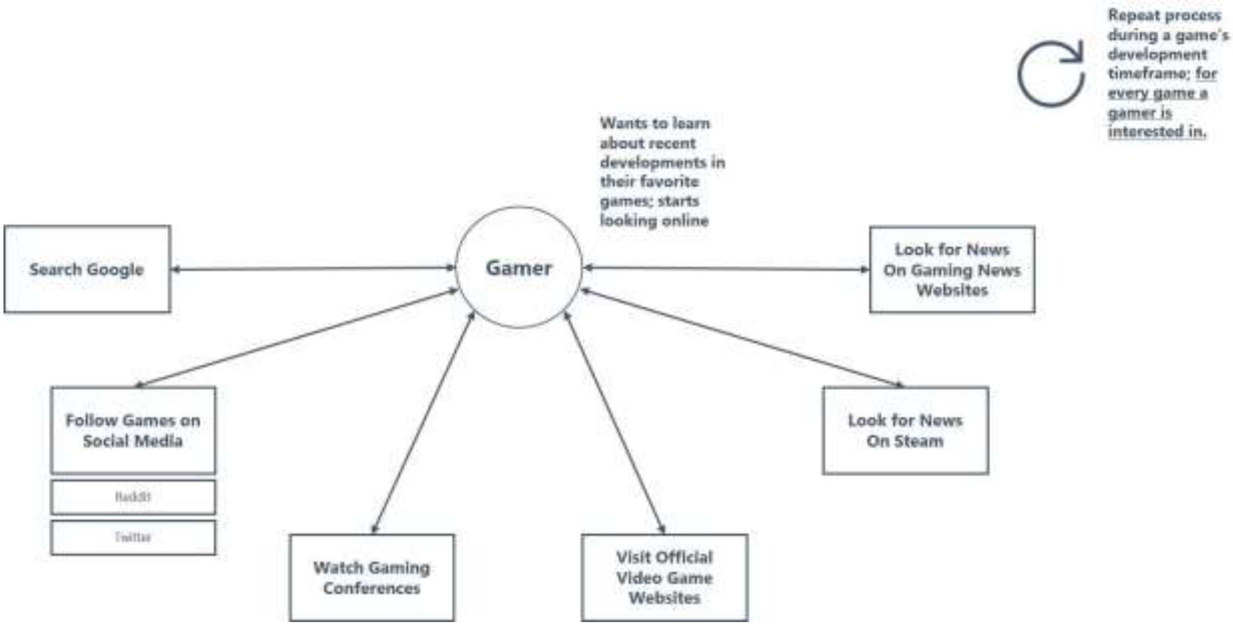
Table of Figures

1. Introduction

In recent years, there has been a surge in the amount of video games released, and 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. 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. 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. This process would be repeated for every game a gamer may be interested in. 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.

Gamers need an easier way to gather the information they need. The solution to this problem would be GameEye. GameEye is a platform in which users can track games. GameEye users will be sent notifications of news articles or new content on video games they follow (e.g. screenshots and official videos like trailers or gameplay). Users will have a watch list in which they will receive information about current and unreleased games they may be interested in (e.g. potential release date delays and news updates). The news articles and tweets will be classified based on the content. An impact score will be computed to see how important information is for a user to know.

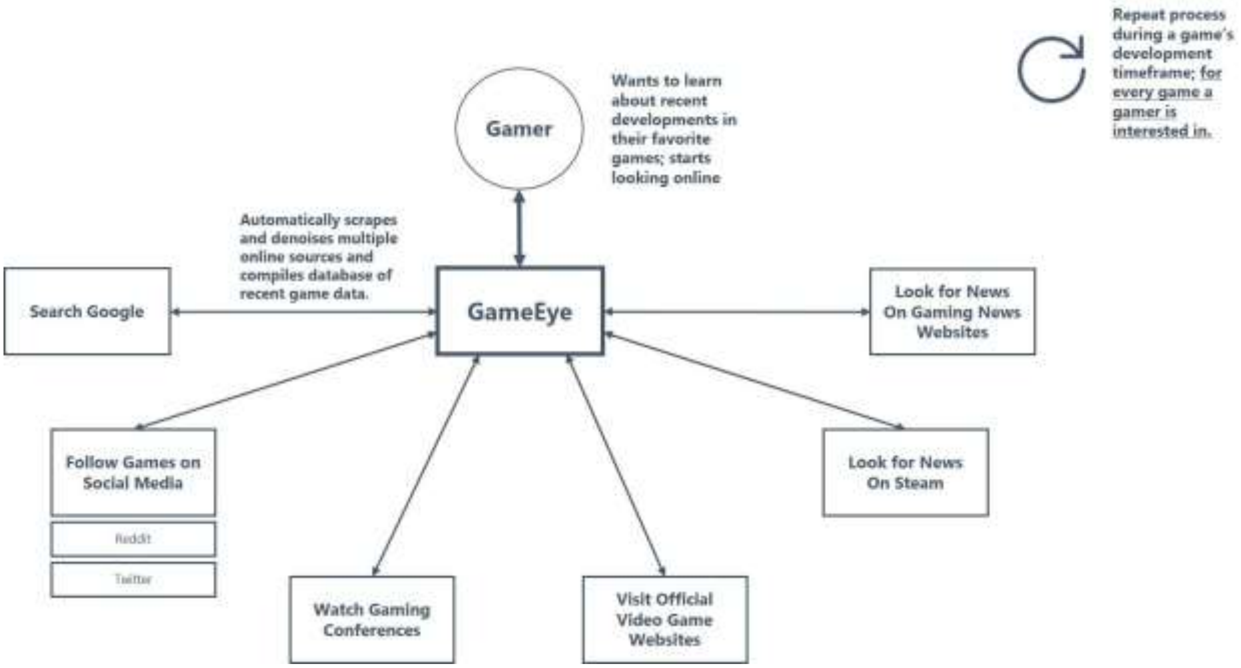
[THIS SPACE INTENTIONALLY LEFT BLANK]

2. GameEye Product Description

GameEye is a web application that will aggregate news information about developing video games, and users will receive notifications on the new information.

Figure 2

Solution Process Flow Diagram describing aggregation of information to gamer


Repeat process during a game's development timeframe; for every game a gamer is interested in.
Gamer
Wants to learn about recent developments in their favorite games; starts looking online
Automatically scrapes and denoises multiple online sources and compiles database of recent game data.
Search Google
GameEye
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

Users will be able to receive information from one place. GameEye will provide the user with notifications for their favorite video games.

2.1 Key Product features and Capabilities

General

GameEye will be a cross-platform progressive web application working on both mobile and desktop devices. It features offline support through local caching and connectivity interruption resiliency features.

Authentication

GameEye will feature authentication such as secure user login and registration. An external provider will be utilized for user login and registration (e.g. logging in using social media accounts). There will also be persistent sessions allowing the user ease of access as they do not need to login to their account everytime. Two-factor authentication may be used as well for increased security. There will also be a password recovery mechanism in the event a user forgets their password.

Account Management

Users have the ability to change their password, modify their profile information, and delete their account.

Game Tracking

GameEye will provide a personal watchlist or list of video games an individual wants to stay informed about. In the watchlist there will be a thumbnail displayed next to each game title. The list will show the new game updates and will be organized by important updates, news articles, tweets, Reddit posts, images, and videos. Thumbnails will also be included for tweets and news articles. There will also be a separate list of games featuring the most-watched games on the platform.

Searching

Users will be able to search for video games based on their title, and the search system will feature support for auto completion.

Web Scraping

Multiple sources will be aggregated to provide game development updates. GameEye will utilize 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.

Notifications

Cross-platform push-notifications will be used to provide users with new game updates. A UI count of notifications will be displayed for each game and each resource category. The user will also be notified of suggested video games to follow based on the most-watched games list.

Settings

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

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 scoring will be assigned for news articles and tweets utilizing machine learning. Important information will also be extracted from news articles and tweets.

2.2 Major Components

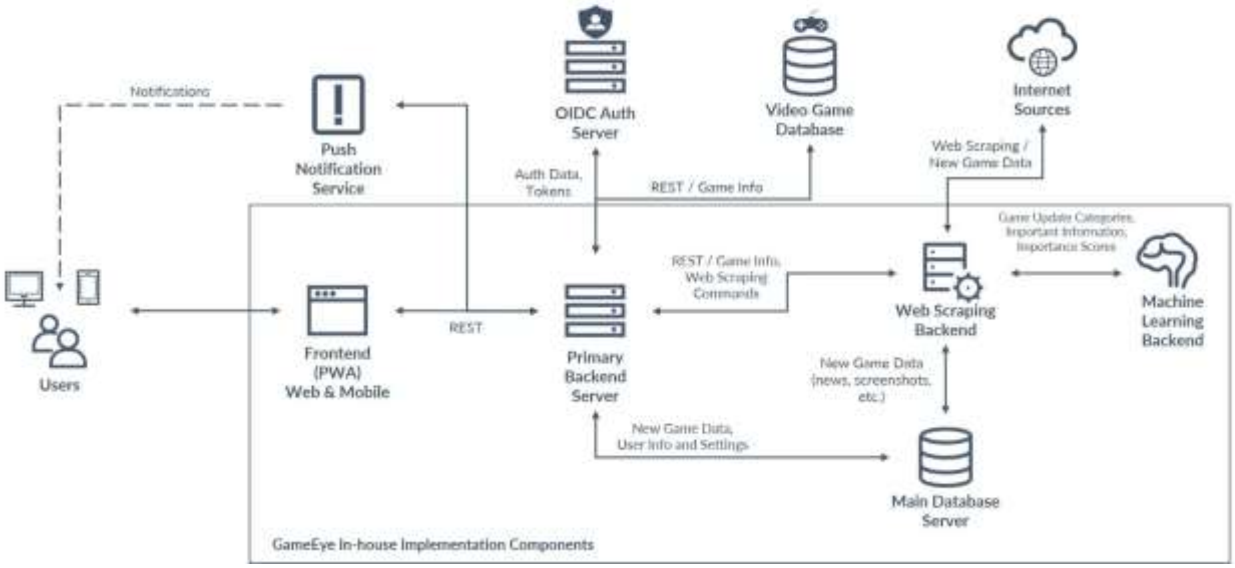
Hardware

GameEye will be utilizing 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 communicate together. 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

[THIS SPACE INTENTIONALLY LEFT BLANK]

Figure 3

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 will provide the frontend of the web application for the user to access. 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 also coordinates web scraping backends and powers 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. It communicates with the machine learning backend to classify scraped video game data and extract important information before putting it in the database. The machine learning backend server is an internal GPU-powered TensorFlow serving server exposing 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.

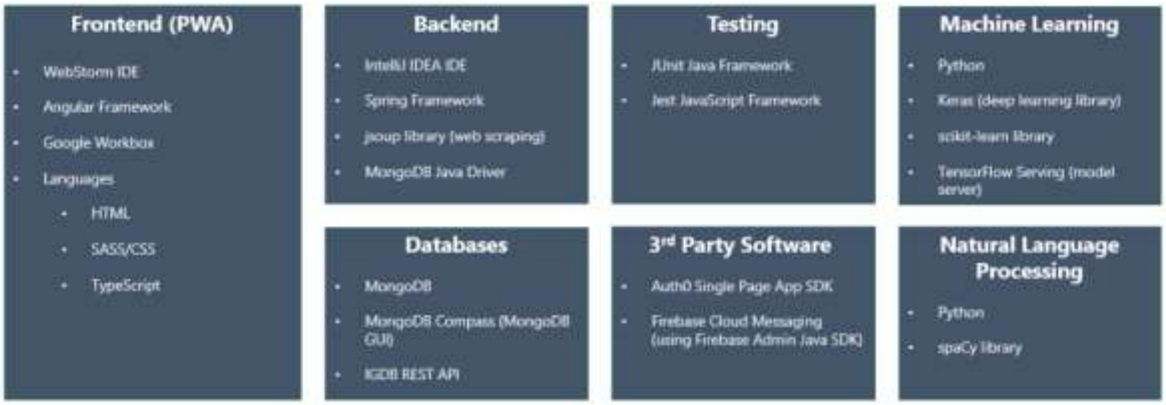
Software

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

The backend of GameEye will be developed using IntelliJ IDEA IDE and Spring Framework. The Jsoup library will be used to provide the web scraping capabilities, and MongoDB Java Driver will be used to build the database. **.**

Figure 4

GameEye's Software Components Diagram



Frontend (PWA)

- WebStorm IDE
- Angular Framework
- Google Workbox
- Languages
    - HTML
    - SASS/CSS
    - TypeScript

Backend

- IntelliJ IDEA IDE
- Spring Framework
- jsoup library (web scraping)
- MongoDB Java Driver

Testing

- JUnit Java Framework
- Jest JavaScript Framework

Machine Learning

- Python
- Keras (deep learning library)
- scikit-learn library
- TensorFlow Serving (model server)

Databases

- MongoDB
- MongoDB Compass (MongoDB GUI)
- IGDB REST API

3rd Party Software

- Auth0 Single Page App SDK
- Firebase Cloud Messaging (using Firebase Admin Java SDK)

Natural Language Processing

- Python
- spaCy library

Testing of GameEye will be provided by utilizing JUnit Java Framework to test the Java code and Jest JavaScript Framework to test the JavaScript.

GameEye will use Python to code the machine learning aspect and will be hosted using the Tensorflow serving (model server). The Keras and Scikit-learn Python libraries will be used to code the machine learning aspects.

Natural language processing will be a part of GameEye as it will allow the web scraped information to be processed, categorized, and assigned an importance score. Python will once again be used to implement this feature and the spaCy library will be used to do so.

Databases will be constructed using MongoDB and MongoDB Compass (GUI). The database will be populated using the IGDB REST API to retrieve data on known video games.

Some third party software will be used such as Firebase Authentication the two-factor authentication. Firebase Cloud Messaging will be 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 ODU game club. Students in the ODU game club can test the web application and provide feedback such as user experience, noise filtering, etc. From there GameEye could be deployed to the masses. Any gamer from experienced to novice could use GameEye. Stakeholders such as video game news sites (e.g. IGN,GameSpot, etc.) 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.

4. Product Prototype Description

4.1 Prototype Feature and Capabilities

Although the real world product is ambitious, the prototype will still provide various features. The prototype 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 will have a personal game watchlist and web scraping for news articles and tweets. 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. GameEyes's Settings will provide the availability to toggle visibility of archived sources, visibility of importance scores, and notifications. The prototype will have classification and impact scoring of news articles and tweets 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 for multiple different resource categories. It will successfully provide robust authentication mechanisms for users, robust game searching and fast autocompletion, a personal watchlist 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 that can easily be mitigated such as scraping RSS feeds to protect against source website structure changes which would cause web scraping to fail. GameEye will store video game database content on its own database for redundancy. The platform will be made scalable to protect against high load. Content would be cached on a user's device to prevent blank application 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. Lastly developers will use proper coding practices to prevent No SQL injection, XSS, or data exposure.

4.2 Prototype Architecture

The prototype will utilize the same architecture as the real-world product with a few exceptions; it will not use replicated servers for load balancing. Some features will not be implemented or will be partially implemented. Web scraping of resource thumbnails tweets and news articles will be partially implemented, but scraping of images, videos, and Reddit posts will not be implemented. Users will not have the option for two-factor authorization. There will be only partial functionality for offline support, local caching, and connectivity interruption resiliency. Users will not be able to submit feedback or receive suggested video game notifications. Impact scoring and resource classification will also be partially implemented.

4.3 Prototype Development Challenges

There will be many challenges to overcome when developing the prototype. Developers will be required to learn new technology and frameworks 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. There is also a challenge in implementing robust caching in the frontend content for connectivity interruption resiliency and performance. Lastly there is also the trouble of implementing robust web scraping with future-proof mechanisms when 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 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