CS39440 - GamePile

MAJOR PROJECT REPORT

Department of Computer Science,
Aberystwyth University

Last updated: **16th April, 2024**

v0.1 - DRAFT

Produced by:

Kal Sandbrook
kas143@aber.ac.uk
BSc in *Computer Science - G400 BSc*

Supervised by:

Dr. Edore Akpokodje
eta@aber.ac.uk
Lecturer in Computer Science

Generative AI

No Generative AI tools have been used for this work.

Name Kal Sandbrook
Date 9th April 2024

Declaration of originality

I confirm that:

- This submission is my own work, except where clearly indicated.
- I understand that there are severe penalties for Unacceptable Academic Practice, which can lead to loss of marks or even the withholding of a degree.
- I have read the regulations on Unacceptable Academic Practice from the University's Academic Registry (AR) and the relevant sections of the current Student Handbook of the Department of Computer Science.
- In submitting this work I understand and agree to abide by the University's regulations governing these issues.

Name Kal Sandbrook
Date 9th April 2024

Consent to share this work

By including my name below, I hereby agree to this project's report and technical work being made available to other students and academic staff of the Aberystwyth Computer Science Department.

Name Kal Sandbrook
Date 9th April 2024

Acknowledgements

Abstract

Contents

1 - Background, Analysis & Process

1.1 - Background

1.1.1 - Aims

The aim of this project was to create a native desktop application that helps users to manage their backlogs (lists of games they want to play) and libraries (games they own). The application allows users to add games to the library and mark them as part of their backlog, in progress or completed. The application will also allow users to search through their games and filter them based on various attributes such as genre, platform or completion status.

Further to this, the application will also allow users to search for games to add to their library via the use of a Third-Party API. This will use a fuzzy search algorithm to allow users to search for games even if they are unsure of the exact name. The application will also allow users to view detailed information about the games in their library, such as the aforementioned attributes.

Some optional features that could be implemented include the ability to export a users game library graphically akin to an old-school forum signature, to facilitate sharing of game completion progress on forums and social media platforms and a recommendation system that suggests games to add to the users library based on their existing library and completion status.

1.1.2 - Research into Similar Tools

Initial research for this project involved investigation into similar tools that already exist. These tools were found by searching online for "game backlog manager" and "game library manager". The most popular and recommended tools were found to be websites called "How Long To Beat"[1] and "Backloggery"[2]. These websites allow users to track their game completion progress and backlog, but they are web-based and do not offer a native desktop application, they are also limited to the library of games that they have in their database, not allowing for users to easily add their own games.

The research into these tools proved useful as it allowed for the mapping out of features that are essential for a backlog manager. In How Long To Beats case, the ability to track completion progress and the ability to search for games in a database were key features. Backloggery brought to light the idea of a graphical representation of a users library, which could be a unique feature to implement in this project. However, the design of the website was found to be very outdated and not visually appealing, with the sites design not having majorly changed since 2007, which could be a key area for improvement in this project, along with the performance improvements that come with a native application.
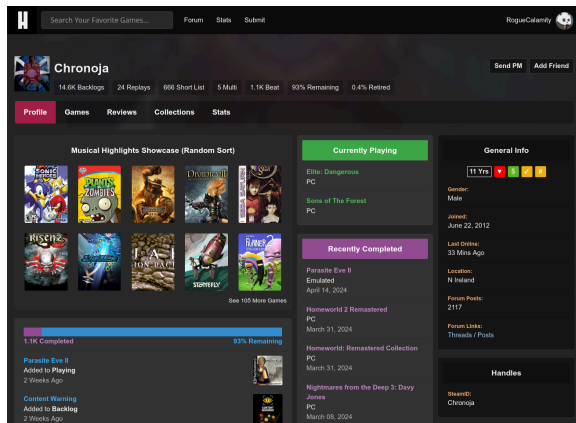
Figure 1: A screenshot of a profile on How Long To Beat.

Figure 2: A screenshot of the front page of The Backloggery.

Another technology that was investigated during development was the "Video Game Preservation Platform" Lutris[3] - however, Lutris is more focused on game installation and management, as opposed to tracking completion and a backlog. This tool was useful for understanding how an application could be used to launch games, which could be a feature to implement in this project. The design of Lutris was also taken into account, as it contains a modern design running on a native ui framework (GTK[4]).
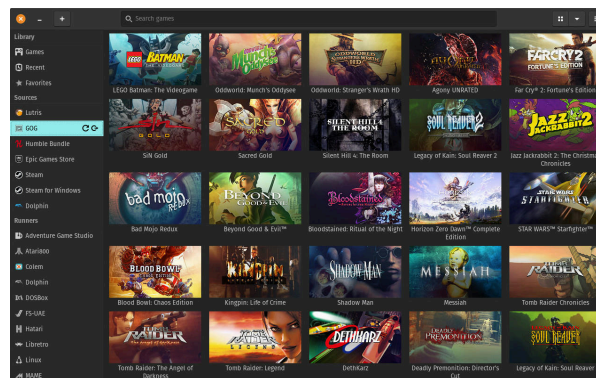

Figure 3: A screenshot of the Lutris application.

1.1.3 - Motivation

The motivation to undertake this project comes from a personal interest in the topic, as efficient tracking of game libraries and backlogs is not something readily available in the market, at least, not in a native desktop application. As existing tools are mainly web-based, they do not offer the same level of performance and integration that a native application could offer, such as being able to launch games directly from the application or being able to use the application offline.

This laid out some of the priorities for the project, such as performance improvements over existing tools, a modern and visually appealing design and the ability to be able to use the application without an internet connection. To this end, investigation into what technologies could be used to achieve these goals was undertaken.

1.1.4 - Research of UI Design

Research into UI design was conducted as a part of this project to ensure that the application was user-friendly and visually appealing. This research involved examining modern desktop applications, notably those in the KDE ecosystem, as well as inspecting the design guidelines for the Qt Framework [5] (The "4Cs": Consistency, Continuity, Context and Complementary being considered throughout design iterations.) and the KDE Human Interface Guidelines[6] (which can be summarised as "Simple by default, Powerful when needed.")

These principles promote reusing design patterns from other applications so that users can easily understand how to use the application, and that complex tasks should feel simple to the user. This research was used to inform the design of the application during development, ensuring a intuitive and user-friendly experience.

1.1.5 - Options for Third-Party API

For the API module of the project, a third-party API was required to allow users to fetch game data from the internet. Ideally, this API would allow for searching for games by name and return detailed information about the game - at the very least, the release date, genre, platform and a description of the game.

This API would also need to be free to use, and preferably not involve a complicated authentication process to access the data (such as having to sign up for an API key). Appropriate Licensing is also a consideration, as the API will be used in an educational capacity.

Two initial options were IGDB[7] (Internet Game Database) and the Steam Web API. The IGDB API was ruled out as it requires the user to have a Twitch account in order to sign up for an API key - which would require functionality that is out of scope for this project, requiring the application to sign up with the *Twitch Developers* system.

The Steam Web API does not require any authentication to access the data, and was found to be a good option for this project. The API does not support searching, but a list of games can be obtained and searched through locally. Steam Web API also provides information of a requisite level of detail for this project.

1018 words

1.2 - Analysis

1.2.1 - Identification of Requirements

1.2.2 - Data Storage

1.2.3 - User Interface

1.2.4 - Performance

1.3 - Process

2 - Requirements

3 - Design

4 - Implementation

4.1 - Class Diagram

5 - Testing

6 - Evaluation

4.1 - Class Diagram

Bibliography

[1]   "How Long To Beat." Accessed: Apr. 16, 2024. [Online]. Available: https://howlongtobeat.com/

>   One of the investigated similar applications that inspired the creation of this application. HLTB is a website that provides information on how long it takes to beat a video game, along with providing functionality to track games and create a backlog.

[2]   "The Backloggery." Accessed: Apr. 16, 2024. [Online]. Available: https://backloggery.com/

>   Another similar application that was investigated. The Backloggery is a website that allows users to track their video game collection and progress.

[3]   "Lutris - Open Gaming Platform." Accessed: Apr. 16, 2024. [Online]. Available: https://lutris.net/

>   The Lutris website, which provides an open gaming platform for Linux. It was investigated as a similar application to the one being developed.

[4]   "The GTK Project - A free and open-source cross-platform widget toolkit." Accessed: Apr. 16, 2024. [Online]. Available: https://www.gtk.org/

>   The GTK Project website, which provides information on the GTK toolkit used for developing graphical user interfaces. It was not used in the development of the application, but used by other applications that were investigated.

[5]   "How the 4Cs of UX design benefit your software development." Accessed: Apr. 16, 2024. [Online]. Available: https://www.qt.io/4cs-of-ux-design

>   An article that discusses the 4Cs of UX design and how they can benefit software development.

[6]   "KDE Human Interface Guidelines." Accessed: Apr. 16, 2024. [Online]. Available: https://develop.kde.org/hig/

>   The KDE Human Interface Guidelines, which provide guidance on designing user interfaces for KDE applications.

[7]   "IGDB API Documentation." Accessed: Feb. 07, 2024. [Online]. Available: https://api-docs.igdb.com/

>   One of the APIs considered for the application. It was not used in the final implementation, but was investigated during the planning phase.

[8]   "Qt." Accessed: Apr. 16, 2024. [Online]. Available: https://www.qt.io/

>   The development framework used for the application. This allows the creation of native applications across multiple platforms with ease.

[9]   "Requests - HTTP for Humans." Accessed: Apr. 16, 2024. [Online]. Available: https://docs.python-requests.org/en/master/

>   The python library that was used to make HTTP requests to the API.

[10]   "TheFuzz - Fuzzy String Matching in Python." Accessed: Apr. 16, 2024. [Online]. Available: https://github.com/seatgeek/thefuzz

The python library that was used to perform fuzzy string matching.

[11] "PyInstaller." Accessed: Apr. 16, 2024. [Online]. Available: https://pyinstaller.org/en/stable/

The tool used to package the application into a standalone executable.

7 - Appendices

Appendix A — Third-Party Code and Libraries

Qt [8]

The Qt Framework (particularly Qt Widgets) was used for the majority of development for this project. It provides a wide variety of features - notably its Qt Widgets, Qt SQL & Qt Concurrent modules.

For academic purposes, Qt Community Edition (the edition used for development) follows the *GNU Lesser General Public License* ("(L)GPL"), a copy of which can be accessed here: https://www.gnu.org/licenses/lgpl-3.0.txt

No tools such as Qt Creator or Qt Designer were used in development.

Python Libraries

For the API module of the project, Python was used due to its excellent range of libraries. The following libraries were used:

- *Requests* [9]
  - Used for making HTTP requests to the API.
- *TheFuzz* [10]
  - Used for fuzzy string matching, particularly using the jaro-winkler algorithm.
- *PyInstaller* [11]
  - Used for packaging the API module into an executable.