# CS 451/551 Final Project

## Deliverable Design Document

Spring 2024 v2.0

Name(s): Jacob Burke

Project title: HowLongToFilter

Connection Information

Port Number: 3061
Hostname: ix.cs.uoregon.edu
Guest Account Login/Password: guest
Database Name: how_long_to_filter

Project URL: http://ix.cs.uoregon.edu/~jburke2/HowLongToFilter/index.html

Highlights:

**howLongToBeat_Webscraper.py:** custom webscraper made that scrapes dynamic webpages to load databases

**index.html:** Starting page for project

**backlogGames.php:** webpage that queries and builds a dynamic html table based off the the resulting data

## Document Summary:

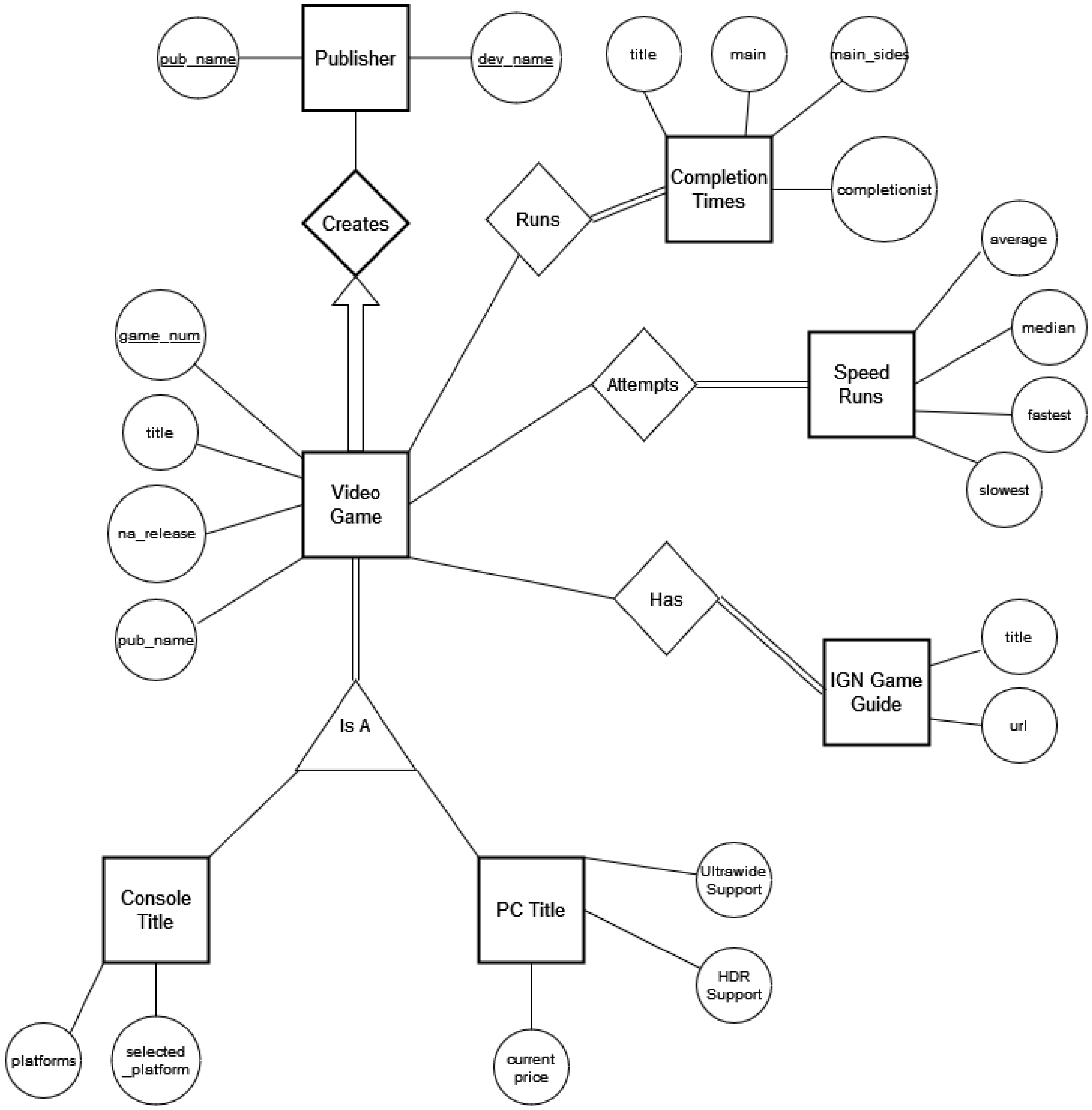
This document is the Deliverable Design Document for the Computer Science 451/551 Database Management Final Project assignment. This document acts as the final submission guide for the project and outlines the project's system architecture including the logical and physical design of the database created, details website functionality, provides a short user guide on said functionality, and links to some of the project's deliverable fragments. The document has detailed diagrams based on the Unified Modeling Langauge which depict the architecture and different modes/components, as well as a Chen diagram of the Database structure.

## Table of Contents:

## 2. System Summary

The HowLongToFilter Website was built to demonstrate mastery of creating and filtering MySQL tables and using a database as part of a website. HowLongToFilter in particular was created to alleviate an issue I have had with gathering info about video games that are in my backlog. Specifically, I was annoyed with the limitations that there was no way to sort my backlog based on the release date. In recent years I've had the issue where I will complete an older title, only for it to have a remastered version announced and released shortly afterwards. The website ingests information about my backlog that's stored on HowLongToBeat.com and attempts to consolidate information about the library. Information that is consolidated includes user-submitted completion times from HowLongToBeat, game availability/compatibility and settings information for PC titles, sorting library by game publishers, and price tracking information provided from Steam. With all of this data consolidated into databases, they can be joined together and presented in a table on the website. Application functions include ingesting a user's backlog off of the HowLongToBeat website, web scraping and consolidating data using the games listed in said backlog, displaying the consolidated info in a table format, while presenting multiple filtering options to help a user determine what title they would like to play next.

## 3. Logical Design



## 4. Physical Design

### video_games:

- **game_num (PK):** Unqiue identification number for each game title scraped from the HowLongToBeat website
- **title:** Game title
- **na_release:** North American release date for game
- **dev_name:** Name of the Game Development company that created the game
- **pub_name:** Name of the Publisher that supported publishing the game

**completion_times:**

- **game_num (PK):** Unqiue identification number for each game title scraped from the HowLongToBeat website
- **title:** Game title
- **main:** Number of hours it takes to beat main game content
- **main_sides:** Number of hours it takes to beat main game content plus some side content
- **completionist:** Number of hours it takes to beat all game content

**speed_runs:**

- **game_num (PK):** Unqiue identification number for each game title scraped from the HowLongToBeat website
- **average:** Average speed run time (speed run is the time it takes to beat tha main game as quick as possible, usually resulting in the credits rolling)
- **median:** Median speed run time
- **fastest:** Fastest speed run time recorded
- **slowest:** Slowest speed run time submitted

**console_titles:**

- **game_num (PK):** Unqiue identification number for each game title scraped from the HowLongToBeat website
- **platforms:** The gaming platforms that the game has officially released on
- **selected_platforms:** The gaming platform that the user has selected they intend to play the title on

**pc_titles:**

- **game_num (PK):** Unqiue identification number for each game title scraped from the HowLongToBeat website
- **ultrawide_support:** Lists if a pc game supports 21:9 resolution (this was manually entered)
- **hdr_support:** Lists if a pc game supports High Dynamic Range color content (this was manually entered)
- **current_price:** Lists the current price listed on the Steam storefront (webscraping for this is currently bugged, with only a few titles getting prices)

**publisher:**

- **Pub_name (PK):** Name of the Publisher that supported publishing the game
- **Dev_name (PK):** Name of the Game Development company that created the game

**ign_wiki:**

- **game_num (PK):** Unqiue identification number for each game title scraped from the HowLongToBeat website
- **title:** Game title
- **url:** Lists the url link to the IGN game walkthrough webpage

## 5. List of Applications

**Select Games Table:** This is the main display table used to show all of the titles in the users gaming backlog. It combines the video_games, speed_runs, & completion_times tables to display information from each table respectively. Further applications support filtering this main view

**Select Publisher Table:** This table view lists all of the publishers and corresponding developers that have created the game titles found in the backlogs. It only utilizes the publisher table

**Select IGN Walkthrough:** This table view shows all of the game titles in the users backlog that have IGN walkthroughs, and then provides a link to said walkthrough webpages. It only utilizes the ign_wiki table.

**Select Platform Filters:** This dropdown allows for the user to filter between all platforms, console titles only, or pc titles only. The all platforms selection is the default for the Select Games Table application. Console Only combines the vide_games tables with the console_titles table while filtering out PC titles. The same can be said in reverse for the PC Only selection which utilizes the video_games & pc_titles tables while filtering out console titles.

**Select Sorting Method:** This dropdown allows for filtering on any Select Games Table selection and any of the Select Platform Filters. Sorting Methords include Ascending by Alphabetical Titles, Descending by Alphabetical Titles, Ascending by North American Relase Date, and Descending by North American Release Date.

**howLongToBeat_Webscraper.py:** This provided python program was used to webscrape game title data off of the HowLongToBeat website. A majority of work on this project went towards implementing this webscraper, as the website loads dynamically so I had to switch from BeautifulSoup and learn how to use the Selenium webscraper. In it's current implementation, a local MySQL database must already be setup in order to load data into, and the backlog url is hard coded into the program.

## 6. User's Guide

**index.html and backlogGames.php:** Once the index.html page has loaded, the user will be presented with a barebones page prompting them to make selection from 3 drop down menus. The dropdown menu selections are described above in section 5. It is worth noting that the Select Platfrom and Select Sorting Method drop down menus only support the Select Games - Games option, and will be disabled and cleared until that option is selected. Once the user has made all filtering option choices, they can hit the submit button below and will be re-directed to the backlogGames.php. At the top of the page, the specific query used to generate the table is shown, and then directly below that is the result of the query dynamically loaded into a table. The column fields and number of rows dynamically change based off the input query. If a user would like to make another selection, a back button is provided at the bottom of the page which will return the user to the index.html homepage.

**howLongToBeat_Webscraper.py:** The provided webscraper can be run to gather data off of the HowLongToBeat webpage, specifically from a users backlog (https://howlongtobeat.com/user/WraithW0lf/games/backlog/1). The url is hard coded into the program in the programs current version. Same goes with the information required to connect to a local MySQL database, which must also have all of it's tables/schemas setup before running the webscraper. Once the webscraping program is launched, each of the titles listed on the users backlog will be gathered, and then individual webpages for each titles will be scraped. Once gathered, the data is formatted and then loaded into the MySQL webpage. This process typically takes between 5-10 minutes. To run the provided python prgoram, both the 'selenium' and 'mysql-connector-python' modules must be installed which can be accomplished with the following commands from the command line:

```
pip install selenium
pip install mysql-connector-python
```

## 7. Contents of Tables

Contents of tables can either be viewed directly on the MySQL server running on ix.dev, or by viewing the HowLongToFilter.sql dump file found in the UofOregon-CS-451/Final_Project github repo here: https://github.com/jwburke256/UofOregon-CS-451/tree/main/Final_Project

## 8. Implementation Code

Code is provided on the UofOregon-CS-451/Final_Project github repo here: https://github.com/jwburke256/UofOregon-CS-451/tree/main/Final_Project

## 9. Conclusion

Originally I set out on this project to do some webscraping with BeautifulSoup and then focus on making a ton of applications for my tables. However, early on I adjusted this plan, as the HowLongToBeat was not a simple static webpage and BeautifulSoup would not work with the site. After doing some research I ended up pivoting to using the Selenium Python library which was a first for me. Because of this, a majority of the project was spent working on the webscraper and integrating it's output into a local MySQL database I setup. This took far longer than expected, and due to the enhanced difficulty of using Selenium over BeautifulSoup, I ran out of time to also webscrape both the pcgamingwiki.com and SteamDB.info websites. I manually entered a few lines for ultrawide_support and hdr_content fields that would have come from the pcgamingwiki.com website. And for the price info, HowLongToBeat still lists prices, but the prices aren't as up to date or don't show as much historical data when compared to pulling directly from SteamDB.

For the table and webpage creation itself, the tables more or less turned out as I expected and provide the necessary functions and filtering that I needed. Once I loaded the data into a local MySQL server, I dumped and ported the server to ix.dev. For the frontend, I am not much of a fan of front end development so instead of focusing on CSS and Javascript interactions, I opted to work on how the table itself populates data. I was not only able to get the tablet to dynamically adjust it's size but I was also able to list the column fields from the SQL query as column headers. The work doing this will allow for much quicker integration and future support for different types of SQL queries to be ran. For the walkthrough table, I was also able to figure out how to make the url's presented to function as hyperlinks within the table.

For future work, adjusting the webscraper to take in a provided backlog url so anyone can pull their own backlog library data would be a good place to start. I would also like to continue learning Selenium and see if I can figure out how to debug the price info, or completely pivot instead to pulling price data from SteamDB.

## 10. References

IEEE Std 1016-2009. (2009). IEEE Standard for Information Technology—Systems Design—Software Design Descriptions. https://ieeexplore.ieee.org/document/5167255

Sommerville, Ian. *Software Engineering*. Pearson Higher Ed, 2011.

Wilson, Chris. (2024). CS 451/551 Final Project Outline Downloaded from https://classes.cs.uoregon.edu/24S/cs451/final.html