AH SDD Project: Game Assist Tool

Contents

[Analysis 1](#_Toc114575481)

[Description of problem 1](#_Toc114575482)

[User Survey 2](#_Toc114575483)

[Constraints 2](#_Toc114575484)

[UML Use Case Diagram 3](#_Toc114575485)

[Requirements Specification 3](#_Toc114575486)

[End-User Requirements 3](#_Toc114575487)

[Functional Requirements 3](#_Toc114575488)

[Project Plan 3](#_Toc114575489)

# Analysis

## Description of problem

* No maintained tool exists which records match data on the game Magic: The Gathering Online. This leaves users to record data manually on an excel document, if at all, which is very time consuming.
* The end users are people who play Magic: The Gathering Online, they are fairly technically adept (as they play an online game), their age range is roughly 18-45. The end users also spend a decent to a lot of money on the program. This is relevant as they typically want to maximise their win percentage, and are capable of installing, and using, an assist tool.
* The app is built on Electron. On the app, a python file can be run which navigates a specific app (Magic: The Gathering Online), by reading the screen. It navigates to the game history, from which it reads each record of data, each record contains the information of a match (best of 3 games), stored in a server-side video. The app uses Tesseract to read text on the screen. For each game it starts a loop and watches the game and reads all information given, which it categorises and stores in a local database (sqlite3). When the loop is exited, it also reads and stores information from a website (using selenium, also name of python library used here), if internet connection is established and compares that data to some of the data from the match to determine which “deck” each player is playing in the match, which it stores in the local database. All information stored in the local database is easily readable in the Electron app.
* The project meets these requirements:
  + Is an SDD project integrated with a local database.
  + The project will use a sort algorithm, object-oriented and procedural programming, an array of records (specifically dictionaries and lists, as python doesn’t have arrays), create a local database (if it doesn’t already exist) and update, select, delete records (based on what the end-user selects) from that database.

## User Survey

## Constraints

* Technical Constraints
  + The program will run on Windows 7 or later.
  + The program itself does not take up lots of space, however, the databases have no limit on how large it can be (this is down to the user’s discretion).
  + Python’s performance is relatively slow, however, it is very sufficient for this program.
* Business Constraints
  + The scheduling and timescales must be met, otherwise the program will be incomplete.
  + There is no budget for the program.
  + There is a single person working on the program.
  + There are no licensing issues with the program.
* Feasibility Study
  + Economic
    - There are no costs associated with the project as it uses free software. It also generates no revenue. However, it can be further built upon and in such a way that can create revenue (e.g. membership program for more features, not yet implemented).
  + Time
    - The program will take 80 hours to complete.
  + Legal
    - There are no legal issues that are currently present: all web scraping is within current precedent (site doesn’t have a robots.txt, so web scraping isn’t disallowed, and the web driver doesn’t login/create an account) and no highly personal data is stored (other than the optional username and password for the game and the app itself), and that data is stored locally, inaccessible by the internet.
  + Technical
    - All the necessary resources currently exist for the project to be completed.

## UML Use Case Diagram

## Requirements Specification

### End-User Requirements

### Functional Requirements

* Inputs
* Processes
* Outputs

## Project Plan

User Stories

AS <user role> I want <what?>

Sean Greaves is 23 years old man that lives in America, on their own in a flat. They’ve being playing MTGO for 4 years and are very committed to the game. They have a middling income job at Specsavers and are highly competitive as they are a high-performance swimming athlete in their spare time.

User Scenarios

end

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task | | Start Date | End date | Resources Required |
| Analysis | | | | |
| 1. | Description of problem | 27/08/22 | 29/08/22 |  |
| 2. | Create user survey | 31/08/22 | 05/09/22 | Survey monkey – create account |
| 3. | Constraints |  |  |  |
| 4. | UML Use Case diagram |  |  | Using Lucidchart |
| 5. | Create persona and user stories |  |  |  |
| 6. | Requirements specification: end-user req. |  |  |  |
| 7. | Requirements specification: functional req. |  |  |  |
| 8. | Project plan |  |  | Using Gantt Project |
| Design | | | | |
| 9. | Pseudocode design |  |  |  |
| 10. | UML class diagram |  |  |  |
| 11. | Project Design |  |  |  |
| 12. | User-Interface Design |  |  |  |
| Implementation | | | | |
| 13. | Implementation |  |  |  |
| 14. | Research and development of new skills |  |  |  |
| 15. | Log of ongoing testing |  |  |  |
| End Testing | | | | |
| 16. | Final Test Plan |  |  | Using pytest |
| 17. | Requirements Testing |  |  |  |
| 18. | Testing with personas and test cases |  |  |  |
| Evaluation | | | | |
| 19. | Evaluation Report |  |  |  |