Fortune Cookie

- Game Description

This video game is basic, straightforward, and quick to play. A fortune cookie is a cookie with a piece of paper with a little statement inscribed in it. This statement contains a life message. Each fortune cookie contains a different message for your life. By choosing a random cookie you will be letting luck work for you. As soon as you click on one of the choices, you will receive your message. The message that will come out has already been predestined for you. It may contain a secret, a tip, a teaching for your future, or an important warning sign. Each player receives a distinct fortune cookie, which holds a different message. As a result, your daily message will be determined by your luck.

- Gameplay

Fortune Cookies will be presented on the screen for players to choose. Each cookie contains a hidden message. Players will click the presented fortune cookies on the screen. After clicking on the fortune cookie a message will be revealed.

- Overall Design Description

The following here are the libraries that our game will be dependent upon, dependencies. Labeled in **`bold`** are those that will have to be installed from the Python Package Index using a package manager such as *`pip`* or ***`poetry`*** and those in *`italics`* will be used only during development and will not be shipped with the final program.

- `random`: This provides our game with the randomization functions needed, such as randomly selecting a list of responses the fortune cookie will tell.
- `json`: We'll use this module so that our program can read and write `.json` files. They're used in the program to store small amounts of data like configuration and settings files in a format that both humans and Python can understand.
- `sqlite3`: It is used as our program's database. This is because SQLite is purpose built to be a database unlike JSON, which will allow it to scale well with larger sets of data.
- *`cProfile`* + *`pstats`*: These libraries allow us to measure the performance of our code. Being a game, these would prove themselves to be handy so we can diagnose what is causing slowdowns in our code.
- **`pygame`**: This is the game library that we will be using. Our reason to use it over **`arcade`** can be seen in **Software/Hardware Description**.
- **`rich`**: This allows us to output rich content and formatting within the terminal with less effort on our end. The package provides a few debugging features and in general makes terminal output pretty and easier on the eyes.
- **`loguru`**: This package is used to log errors, warnings and such for debugging purposes, allowing us to record what has gone

wrong. We prefer this over the `logging` module as it has an easier and nicer developer experience overall and improves on the tracebacks.
- **`typer`**/`argparse`: This package allows us to provide a command line interface to our program. This lets us set flags prior to when the program runs
- ***`black`*** **+** ***`isort`***: We use these tools so that we, as a team, no longer need to disagree on how we should style our code and it formats our codebase to be consistent and easy on the eyes to read.

- Software/Hardware Description

The programmers will be using python for prototyping our game. The effect of our work will be immediately visible and possibly to quickly deliver a playable project. Due to the newcomer-friendly syntax of Python, the developers can focus on the basics of game programming, not the complexity of the language itself.

The programmers will be using a popular library that supports developing games in Python named Pygame. This LGPL-licensed engine allows you to create completely commercial projects. Furthermore, the library is built in such a way that it is simple to leverage many processing cores. Because the core functions were written in C and Assembler, your project will run 10–20 times faster than if it were written entirely in Python.

PyGame also has the advantage of being compatible with any operating system, from Windows to the Dreamcast platform.

The programmers will also use Pylance (or Pyright if not possible) for Python to catch bugs during the development rather than at runtime. It also helps in providing better IDE autocompletion.

- Target Platform

Windows, macOS, Linux. Python not available on Android and iOS.

- Target Audience

The game is targeted for all people looking for a simple game to pass the time. This applies to all ages where all the mechanics of the game are easy to understand and simple to use.