StudyStar⭐ Developer's Guide

Overview:

Welcome to StudyStar⭐ - an app for all your studying needs! This is a data-driven TKinter quiz application written in Python 3.7. The purpose of StudyStar⭐ is to allow users to create their own question set .csv files and have a way to have the sets presented back to them in order to quiz themselves on the content. This prevents the need of using physical flashcards or things of that nature that require a lot of manual effort. The target users for StudyStar⭐ are students, professionals, and anyone who needs a resource for studying to reinforce learning material.The overall goal is having a user be able to load a file,store that content, present that content back, and keep a running score of points based on correct answers.

Problem Addressed/Task:

The problem it will help alleviate is studying can be tedious, frustrating, and time-consuming for students. By having a virtual flashcard set, study materials can be accessed whenever the student has their computer instead of relying on physical notecards. Creating the flashcards virtually saves a lot of time and resources compared to the traditional on-paper method. The app will incentivize successful studying through score-keeping, which can help motivate a learner to continue learning the concepts. Overall, the objective is to allow the user to study content relevant to them and be motivated to interact with their material by earning points for correct answers. Target users are any person wanting a platform to help them study material or learn any type of content. The primary expected users would be students, ranging in age from middle school to college.

The application allows a user to select a .csv file with their question set data, load that saved set from the file, and run the program with the question set data selected..

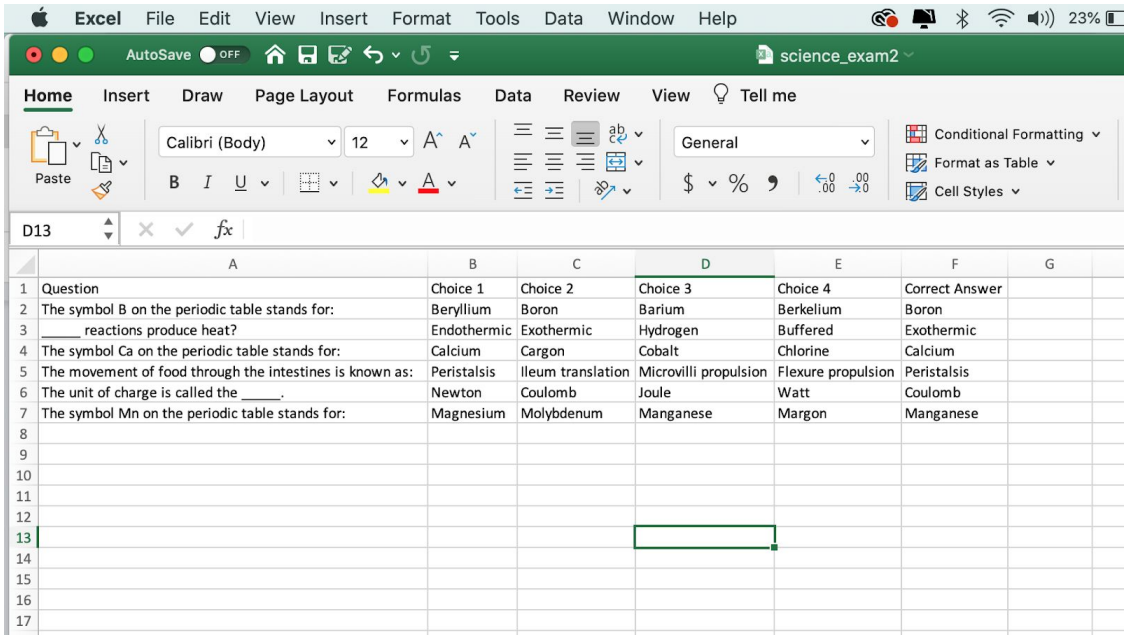
Install/deployment/admin issues:

Outside of proper configuration of the .csv files the application uses, there is nothing additional that the developer needs to do. The next section details how to configure the .csv question set files.

Data/data configuration:

Data used in the quiz comes from .csv files that are imported into the program as reader objects. Modification and creation of this data can be done directly to the .csv file, but must the .csv file must maintain the column structure and have a header row. It also must remain comma delimited. It’s important to note that there is currently no error handling in the quiz application for incorrect .csv structuring, which is why it’s essential to set up the .csv files correctly.

To begin using StudyStar⭐, users will need to have previously created .csv files storing question sets prior to starting the application. The structuring requirements for these .csv files to work properly with the application are described below. To see an example of what this structuring looks like, please see the 3 .csv file examples (random_questions.csv, math_quiz4,csv, science_exam2.csv) located in the "questions csv files" folder in the root project folder. You can also look at the example image shown below. You are encouraged to save the .csv files you create in this “questions csv files” folder as well.



	A	B	C	D	E	F	G
1	Question	Choice 1	Choice 2	Choice 3	Choice 4	Correct Answer	
2	The symbol B on the periodic table stands for:	Beryllium	Boron	Barium	Berkelium	Boron	
3	_____ reactions produce heat?	Endothermic	Exothermic	Hydrogen	Buffered	Exothermic	
4	The symbol Ca on the periodic table stands for:	Calcium	Cargon	Cobalt	Chlorine	Calcium	
5	The movement of food through the intestines is known as:	Peristalsis	Ileum translation	Microvilli propulsion	Flexure propulsion	Peristalsis	
6	The unit of charge is called the _____.	Newton	Coulomb	Joule	Watt	Coulomb	
7	The symbol Mn on the periodic table stands for:	Magnesium	Molybdenum	Manganese	Margon	Manganese	
8							
9							
10							
11							
12							
13							
14							
15							
16							
17							

.csv file structuring guidelines:

1. Have the first row of the .csv file be a header row with the names "Question", "Choice 1", Choice 2", Choice 3", Choice 4", "Correct Answer". Although it is technically possible to have more than 4 "Choice" options, it is not recommended, due to keeping the GUI clean and quiz clean from complications.
2. Type your questions in the "Question" column. The app will use this column to present this back as the question while taking the quiz. You can type them in whatever formatting/wording works best for you. Some examples:The unit of charge is called the _____.;The symbol Mn on the periodic table stands for:;100 + 100 = ?; Who founded Apple?
3. Type your question answer options in the "Choice" columns. The app will use these columns to present the content back in radio buttons displaying answers to choose from while taking the quiz.
4. Type the correct answer out of the options in the "Correct Answer" column. The "Correct Answer" column designates the correct answer and is what the application uses to know if the user selected an answer correctly or incorrectly.

5. Save your .csv file anywhere on your computer (not required, but recommended to save all files in in the actual project root folder called "questions csv files") with a name that describes the study question set, such as "science_exam7", "math_quiz2", "english_test1", etc.

NOTE: You can add as many questions to the .csv files as you like. The app will run through however many questions/answer groupings that are in the file, whether that number be 5, 50, 100, etc.

User interaction and flow ("walkthrough"):

The final work flow for the program is described below:

1. User starts the quiz application by running main.py
2. Main screen: user is shown the main screen, which displays a running clock with the current time, the StudyStar⭐ title, a “Quit Quiz” button, an “Instructions” button, a label that reads “Click to choose a .csv file”, and a “Load File and Start Quiz” button.
3. At any points prior to starting the quiz, during the quiz, or after the quiz, the user can click the “Instructions” button to see instructions for successfully taking the quiz.
4. User selects the “Load File and Start Quiz” button.
5. A filedialog appears where a user is asked to select a .csv file from their computer containing their question set. This is restricted automatically to .csv files, so no other extensions will be able to be selected.
6. User selects their desired .csv file and clicks “Open” in the filedialog box.
7. The quiz begins automatically with the questions from the selected .csv file.
8. Each question and corresponding answer options group is shown 1 at a time.
9. The user selects an answer to the question by clicking on the radio button matching their desired answer. An answer can be changed as many times as needed by simply clicking on a different radio button option. Once the “Check

Answer” button is clicked, however, the selection is final with no changes able to be made.

10. The program checks to see if the user’s chosen answer matches correctly with the designated correct answer from the questions set .csv file.
    a. If the answer is correct:
        i. A label appears giving feedback “Correct! + 10 points!”
        ii. User gets 10 points added to their score total.
        iii. User gets 1 added to their total number questions right score.
        iv. Program moves on to the next question.
        v. The progress bar at the bottom of the screen moves forward, in increments matching the number of total questions in the set.
    b. If the answer is incorrect:
        i. On the 1st attempt out of 2 total attempts:
            1. A label appears giving feedback “Incorrect! -5 points. Try again.”
            2. User loses 5 points from their score total.
            3. User loses 1 out of their 2 attempts.
            4. User is able to re-select a different answer and hit “Check Answer”
        ii. On the 2nd attempt out of 2 total attempts:
            1. A label appears giving feedback “Incorrect! -5 points.No tries left. The correct answer is: _____”
            2. User loses 5 points from their score total.
            3. Program moves on to the next question.
            4. The progress bar at the bottom of the screen moves forward, in increments matching the number of total questions in the set.
11. If the last question in a question set has been answered, the program breaks the asking questions loop.

12. Summary screen: The program congratulates the user, shows the total number of points out of points possible, and shows the total number of questions answered correctly out of number possible.
13. Exiting the quiz: The user is able to exit the quiz by clicking the “Quit Quiz” button, which closes the application.

Program structure:

Main.py - initiates the primary loop and runs program

|

CreateQuizApp.py - main.py calls CreateQuizApp.py (main application functions/classes and creates GUI components)

|

QuestionsModel.py - CreateQuizApp.py calls QuestionsModel.py (main program for reading the questions set .csv files.)

|

tkinter - CreateQuizApp.py calls tkinter directly from Python

The primary program is executed through a sequence of functions located within CreateQuizApp.py:

PRIMARY STEP	**FUNCTION**
1. Main Screen	build_gui(), clock()
2. View instructions	about()
3. Select .csv file	browse_files()
4. Start quiz	start()
5. Get and show questions	get_question(),
6. Get and show answer options	get_options()
7. Check if user’s answer is correct	is_answer_correct()

8. Provide feedback	check_and_update()
9. Move on to next question in set	get_next_question()
10. Move progress bar forward	step()

In addition to these primary functions, the quiz program also contains sub-level functions that are designed to handle and support specific aspects of the main functions. More detail about these sub-functions can be found in the docstrings within the program.

Known Issues:

As of 08/07/20, there are no known bugs or issues.

Future work:

The project was developed based on the available time and resources available upon the time of creation. In the future, this project's main core could easily be expanded and improved with the following features:

1. Development of a second creator application that would expand on the core by allowing users to not only read a .csv file, but also select an option that would create and write a questions .csv file right in the GUI itself. This second question set creator app would also have a mode that would allow users to edit existing questions sets.
2. Development of the "Study Again" button that would display at the end of the quiz and take the user back to the home screen to select a .csv file and study again. This would prevent needing to quit the application and re-boot in order to study another set.
3. Development of the ability at the end of the quiz to re-study only the questions that a user got incorrect. This would reinforce the material and promote practice with the questions a user struggled with.

4. Development of the code's exception handling for .csv files. There could be error messages that would alert users and not allow the program to accept .csv files that do not follow the outlined structuring requirements. If a .csv file does not meet the structuring requirements, the quiz would not start with that file and the user would be reminded of the requirements and asked to select a different .csv file.
5. Inclusion of images/graphical elements to improve the excitement of the application and increase the motivation of users to study.
6. Inclusion of the option to have audio playing in the background while taking the quiz.
7. Improvement of the layout/grid of the widgets. Restructuring could improve the way the quiz responds to longer worded question strings and longer answer options.
8. Transforming this application into a web app using Flask or Django.