AT2 Assignment 2:

Client's Briefing - Transcript

Hello, I'm from *QUT News Services* and I want *you* to help us with our new software development project!

Our company wants to market a new desktop application for accessing news from all over the world. It must have a simple Graphical User Interface and a reliable back-end which extracts the latest news stories from online sources.

I'm aware that you've been attending our company's training courses and you now have all the necessary skills needed to complete this project which is due for completion at the end of week 12.

We need you to design and implement an attractive and capable Graphical User Interface using Python 3's Tkinter module. The program must be portable across multiple computing platforms, so you must use standard Python 3 modules only; extension modules that must be downloaded and installed separately are not allowed.

The purpose of the project is to develop an application that gives access to two live websites and extracts information from those sites and displays it in a number of ways.

We've provided you with a program template to get you started. And when you run that template this is what happens.

[*Shows the Python code template and how to run it*]

This template contains very little, just a function definition and some "import" statements. Down the bottom, where it should be doing all the work, it simply has the statement "`pass`". Therefore, the template that we've given to you doesn't do anything at all.

PART 1 – The Front End (GUI)

I've spoken to our back-room boffins and they've created a demo to illustrate the idea. They've included examples of all the widgets needed for the app, but you're encouraged to design your own.

[*Runs the user interface demo*]

The GUI is intended to give its user access to news sources from two different countries. We're very keen to educate our users on news perspectives from all over the world.

The completed GUI will allow the user to select a news source and then see a snapshot in the GUI of the top five (5) news items from that site. It should also be capable of generating an output file containing a more detailed summary of the top (10) news items from the chosen source.

You *don't* need to copy our example, but you must design and implement your own graphical user interface with equivalent capabilities, using Tkinter in Python 3. Specifically:

- Your GUI's window must have a **title** which names your application.
- Within the window there must be an **image** and a name which clearly identify your application. The name and image can be separate widgets or can be combined into one, as per our demo. This image can be sourced from wherever you like, or you

could even create one yourself if you are so inclined. In any event, this image is to be stored locally, in the same directory as your application .py file.

- There must be **one or more widgets that allow the user to choose a source** for news stories from two different international sources. Our boffins have used radio buttons for this purpose but there are other kinds of widget that could be suitable for the job, such as menus, lists or push buttons.
- There must be a **widget for requesting a snapshot** of the top (5) news stories from the chosen source.
- There must be a **widget for displaying a snapshot** of those top (5) news stories.
- There must be a **widget for generating an HTML summary** of the top ten (10) news articles.
- There must be a **widget for opening the live website** of the chosen source.

Your solution must be portable across different desktop computing platforms (Microsoft Windows, Apple macOS and Linux) as far as possible. For this reason you should use Tkinter's "grid" geometry manager and *not* the "place" manager. It would be particularly difficult to achieve a nice layout with the simple "pack" manager.

PART 2 – The Back End (web scraping)

You must choose a category of news (e.g., sports, finance, fashion, weather, stock market etc.) and then find two different websites in that category, from different countries, which are regularly updated. The websites must publish articles that each include:

- a title (or heading);
- a photo/image;
- a summary/synopsis (giving more information about the news article); and
- a publication date/time of the news article.

Fortunately, there's a readily available source of such news stories, known as "RSS feeds". RSS stands for Really Simple Syndication. RSS documents are written in an extension of HTML called XML. They're used for publishing information that is updated frequently in a format that can be processed easily by computer programs, which is just what we need. For our purposes we will extract story elements from the documents via pattern matching.

[*Demonstrates RSS source code in the browser*]

When the user chooses from one of the news sources, they should be able to then choose to:

- **Display in the GUI a snapshot** of the top five (5) news articles from that news source. This should generate a list of those articles, including for each:
    1. The date and/or time of publication of the story; (Note that different news sources produce their publication dates in different formats either absolute, such as "May 4th", or relative, such as "2 hours ago". You don't need to standardise the date format, just display it as it appears in the source web document); and
    2. The News item headline (or title).

- **Generate a summary** of the top ten (10) news articles, storing it as an HTML output file, and opening it immediately in the user's default browser. The output HTML file must include:
    1. The HTML5 standard tags for an HTML page;
    2. An appropriate heading describing the news source;
    3. A link to the image contained in the live news source;
    4. The URL of the news source as a hyperlink; and;
    5. Details of the first ten (10) news items, each of which includes:
        a) The news item number (from 1 to 10);
        b) The title of the news article;
        c) The image from the news article;
        d) A short summary/synopsis of the article; and
        e) The date and/or time of publication of the news article.

    This generated HTML file must open automatically in the user's browser. Take a look at the function imported into the supplied template and named "urldisplay" – which makes this surprisingly easy.

    Most importantly, of course, the news must be up-to-date and accurate. The headline, summary, photo and date must all belong to the same story. There must be no errors in the way the text is displayed. For instance, no HTML tags or entities or Unicode character mark-ups should be visible to the user. If these appear in the downloaded text you will need to delete or replace them before displaying the text.
- **Open the live website** in the user's browser so that the user can see the original source of the news articles. Again, make use of the "urldisplay" function mentioned above.

[Demonstrates a prototype meeting those requirements]

The final requirement is that **your application must be robust** to changes in the news source and errors. Our customers will hate it if the program crashes all the time! Not only do you need to allow for updates to the news, but you also need to allow for the possibility that the news servers deliver something unexpected like a temporary "out-of-service" message or even that the server is entirely offline. No matter what happens, your program must always display informative messages to the user and it must never produce red error messages in the shell window. Your training in week 10 will give you more information about how to make your program robust.

You must upload your submission to our corporate IT system before the deadline, which is Friday Week 12. Please upload multiple drafts as you develop your solution as insurance against system failures (both yours and ours).

You'll need to submit both a Python program and any image files you have used to support your GUI.

Put your image files in the same folder as your Python code so that we don't have to worry

about different file path representations across different computing platforms. Put all the items needed to support your GUI in a "zip" archive and upload it as a single file.

[*Demonstrate zipping the application files ready for submission*]

Do *not* use any other compression formats such as RAR or 7Z. To create a zip archive on a Mac simply right-click on the folder and select "Compress". Under Microsoft Windows select "Send to" a compressed archive.

So that it will work on any platform that we may use to assess your solution, your program must be portable and must run in a standard Python 3 environment. It must ***not*** rely on any modules that need to be downloaded and installed separately, such as "PIL", "Pillow" or "Beautiful Soup", because we won't be able to run your code.

That's it. Remember that the sooner you start coding, the longer it will take. Make some time for thorough analysis of these requirements, and then for design and strategy development.