University of Westminster
School of Electronics and Computer Science

5COS005W Programming

Module leader	Girish Lukka
Unit	Coursework
Weighting:	50%
Qualifying mark	
Description	**Software Development**
Learning Outcomes Covered in this Assignment:	LO4, LO5
Handed Out:	16th Mar 2023
Due Date	02th May 2023, 1:00 pm **Viva – week beginning 8th May, 2023**
Expected deliverables	A single pdf document detailing design, implementation, testing and evaluation of an application and a single zipped file containing all the python script files.
Method of Submission:	online via Blackboard
Type of Feedback and Due Date:	Generic and written feedback to all students three weeks after submission. All marks will remain provisional until formally agreed by an Assessment Board.

Assessment regulations

Refer to section 4 of the *How you study* guide for undergraduate students for a clarification of how you are assessed, penalties and late submissions, what constitutes plagiarism etc.

Penalty for Late Submission

If you submit your coursework late but within 24 hours or one working day of the specified deadline, 10 marks will be deducted from the final mark, as a penalty for late submission, except for work which obtains a mark in the range 40-49%, in which case the mark will be capped at the pass mark (40%). If you submit your coursework more than 24 hours or more than one working day after the specified deadline you will be given a mark of zero for the work in question unless a claim of Mitigating Circumstances has been submitted and accepted as valid.

It is recognised that on occasion, illness or a personal crisis can mean that you fail to submit a piece of work on time. In such cases, you must inform the Campus Office in writing on a mitigating circumstances form, giving the reason for your late or non-submission. You must provide relevant documentary evidence with the form. This information will be reported to the relevant Assessment Board that will decide whether the mark of zero shall stand. For more detailed information, please refer to the University Assessment Regulations.

Coursework brief:

This coursework is about designing, building, testing and evaluating a python GUI application that displays a set of images (png, gif, jpg, etc) of friends as a gallery. Each image displayed will also have a label underneath showing the image name. Each image will be a button that a user can click to reveal another gallery that will contain images of friends of this friend.

The user interacts with the gallery through a set of buttons. The buttons are described below.

A single image here gives an **overview** of the application, detailed images and button interaction images are attached at the end.

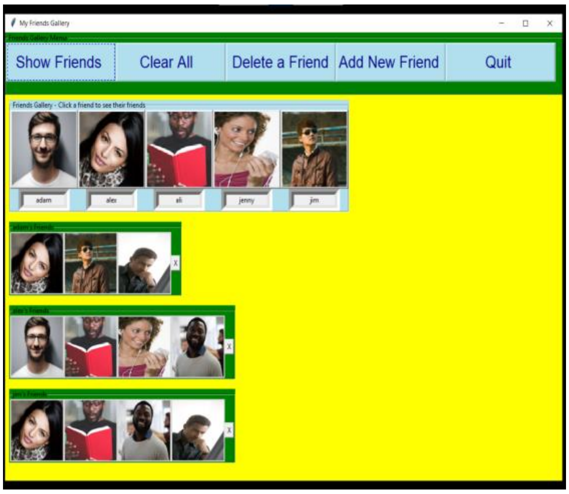


Figure 1 - Overview

Buttons

Button '**Show Friends'** – should display all images from a folder that has all the images of friends. Repeated press of the button should only render one set of these images. The image displayed should show the image name i.e. friend's name and it must also be a **button** – referred to as '**friendButton**'

friendButton – this button when clicked should reveal a gallery in another frame that has the images of friends (as **Labels** not buttons) of this particular friend. Figure 1 shows friends of Adam, Alex and Jim, displayed in their own frame. Each of these frames has a remove button, rendered as X.

Button X – should remove the gallery of that particular friend.

Button 'Clear All' – should clear the gallery, if displayed and just show the main GUI window with a single frame that has the menu buttons. This button should only function if the images are visible.

Button 'Delete a Friend' – should launch underlying operating system window showing the files in the relevant folder. The user will select the friend image to be deleted which will invoke a message window (popup) seeking confirmation of the deletion. The deletion will only be carried out if the user confirms it and the images will be redisplayed without the deleted one.

Button 'Add New Friend' – should launch underlying operating system window showing the files in any folder that the image may be. The user will select the friend image to be added that will invoke a message window (popup) seeking confirmation of the addition. The addition will only be carried out if the user confirms it and the images will be redisplayed with the added one.

Button 'Quit' – should quit the application after user confirms it via a popup.

The user interface is a single window with a number of frames. The application aesthetics will demonstrate good HCI understanding in areas of colours, fonts, look and feel.

The images below are indicative of how this application is meant to work.

The application will be built in Python3 using the tkinter package only.

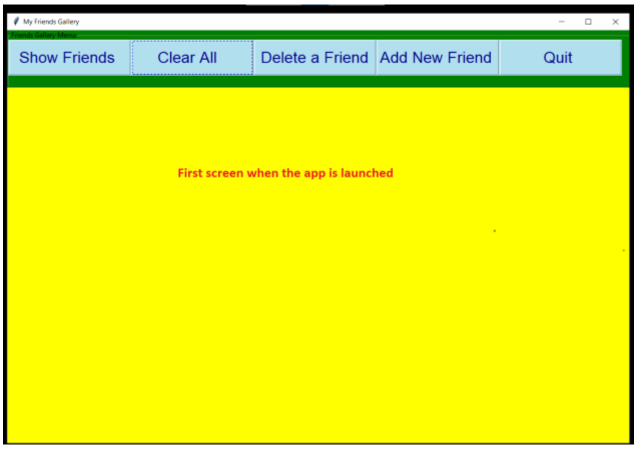
Note use of third party libraries or drag and drop widgets is **not allowed**.

The course work will be marked as per rubric shown below and **viva/demonstration** will take place week beginning 9th May, 2023 - times, rooms and days will be same as your seminar sessions.

Failure to demonstrate your work will result in a maximum mark of 30.

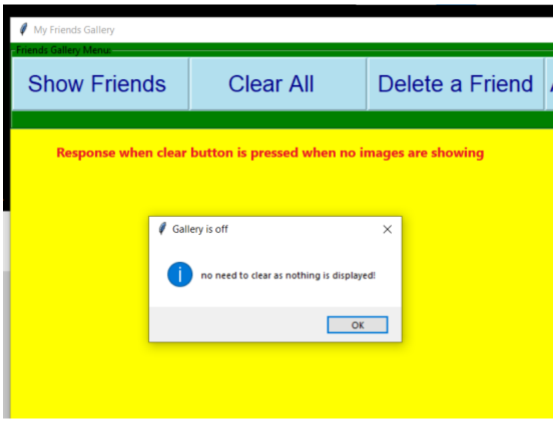
Screen images below indicate the functionality of the application and the **text in red** on the image describes it and is not part of the application.

Opening screen - Menu frame with the buttons



First screen when the app is launched

Clear All activated when no gallery is visible

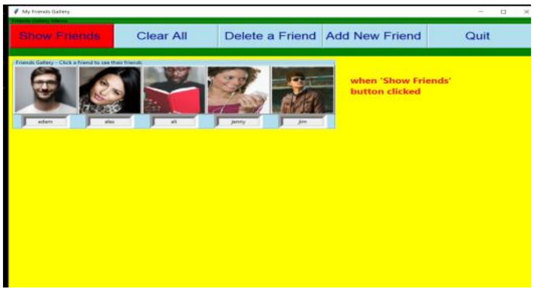


Response when clear button is pressed when no images are showing

Gallery is off

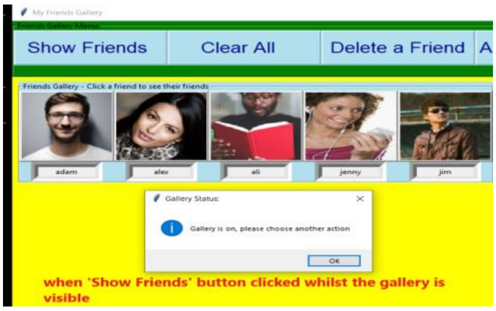
no need to clear as nothing is displayed!

Show Friends button pressed when no gallery is visible



when 'Show Friends' button clicked

Show Friends button pressed when gallery is visible

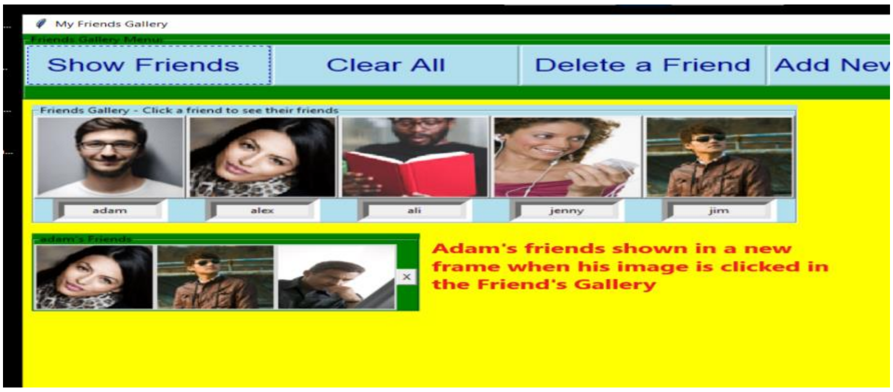


Gallery Status:

Gallery is on, please choose another action

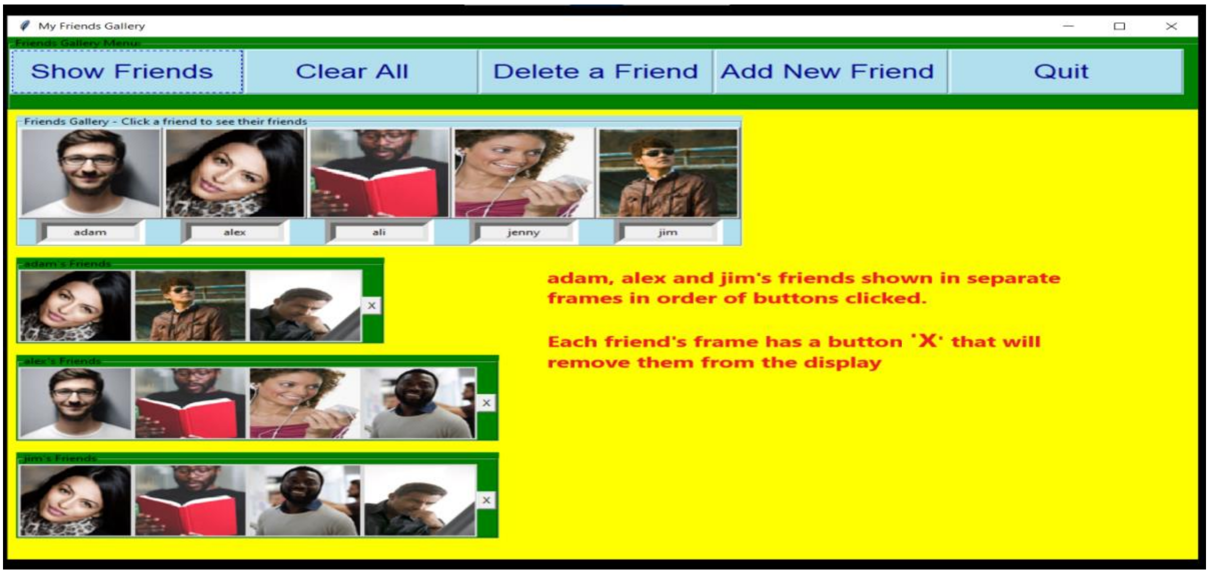
when 'Show Friends' button clicked whilst the gallery is visible

Adam image clicked



Adam's friends shown in a new frame when his image is clicked in the Friend's Gallery

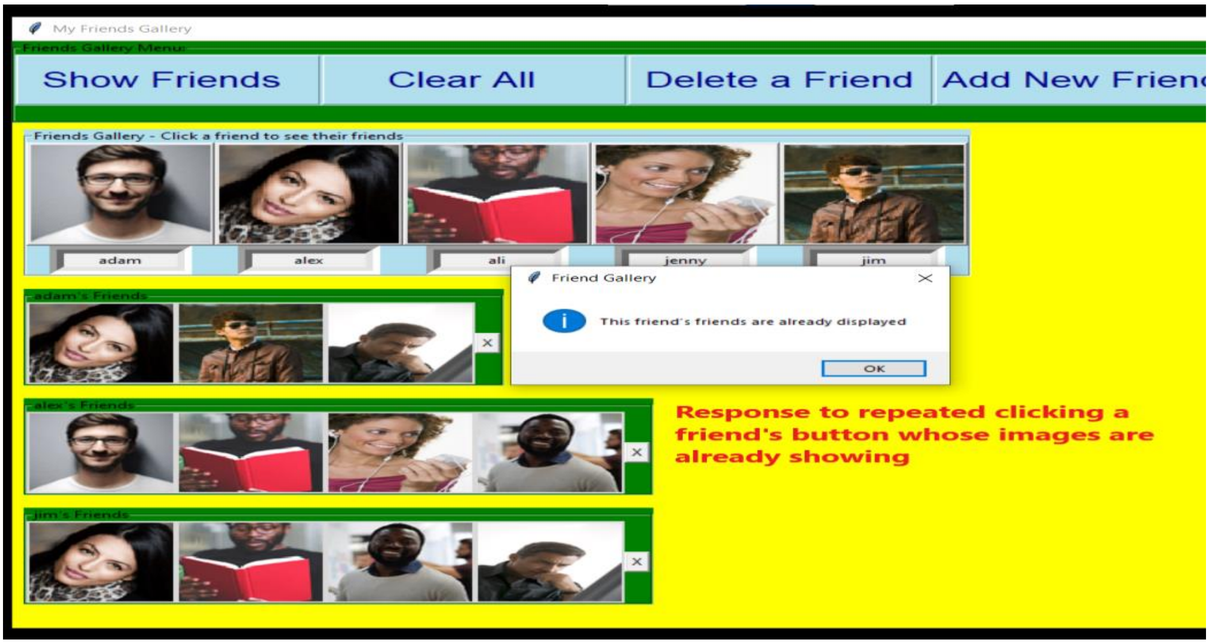
Adam image, alex image and jim image clicked in order



adam, alex and jim's friends shown in separate frames in order of buttons clicked.

Each friend's frame has a button 'X' that will remove them from the display

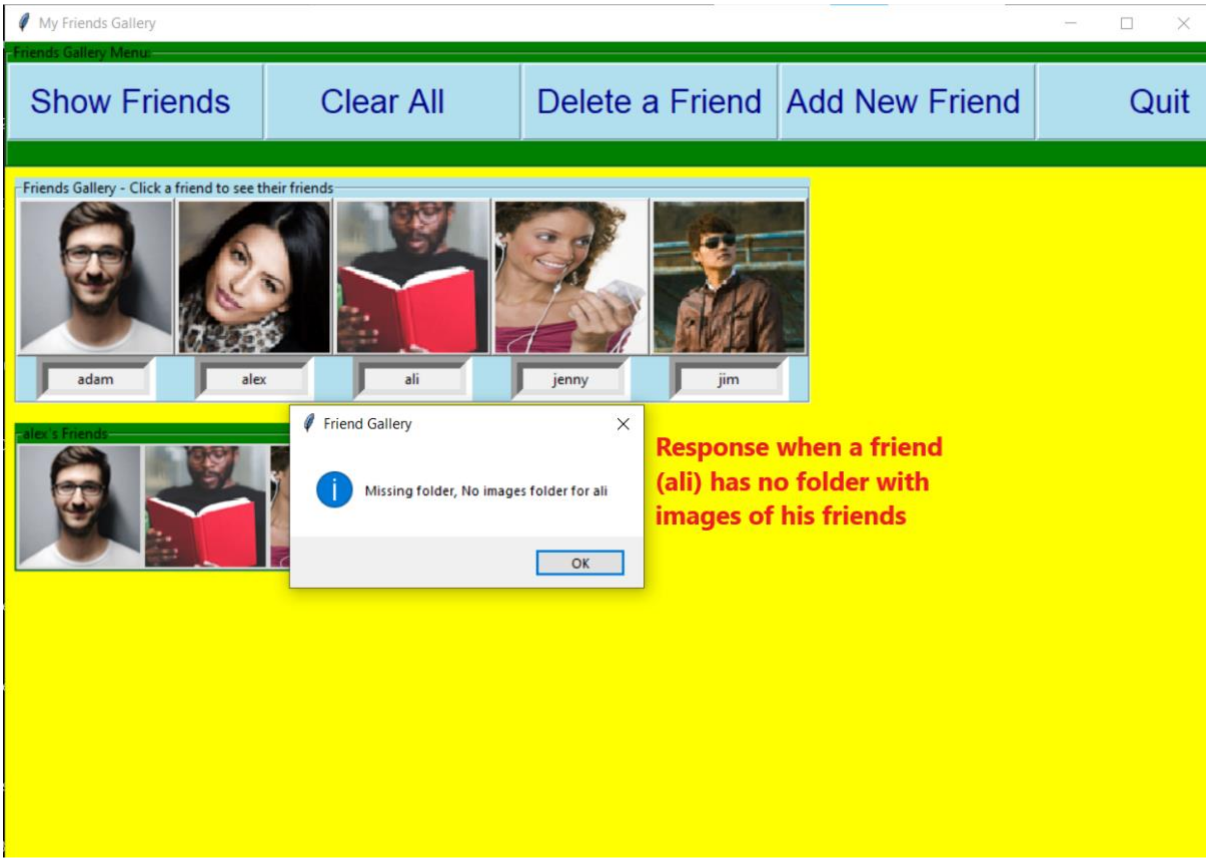
Adam image clicked again



This friend's friends are already displayed

Response to repeated clicking a friend's button whose images are already showing

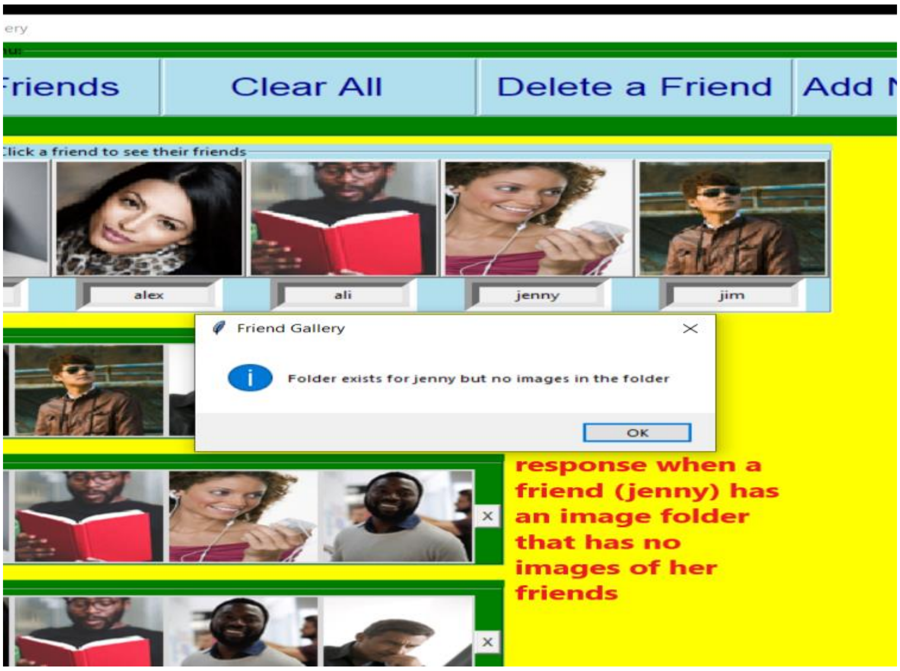
Clicking a friend for whom there is no image folder (ali)



Missing folder, No images folder for ali

Response when a friend (ali) has no folder with images of his friends

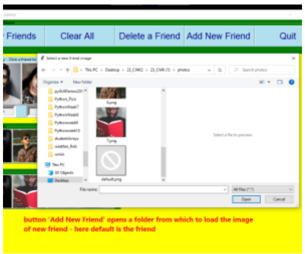
Clicking a friend for whom there is an image folder but no images (jenny)



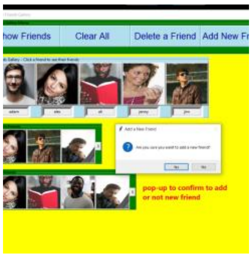
Folder exists for jenny but no images in the folder

response when a friend (jenny) has an image folder that has no images of her friends

Add New Friend clicked

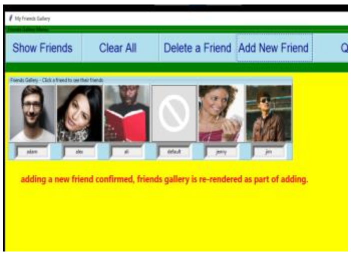


button 'Add New Friend' opens a folder from which to load the image of new friend - here default is the friend



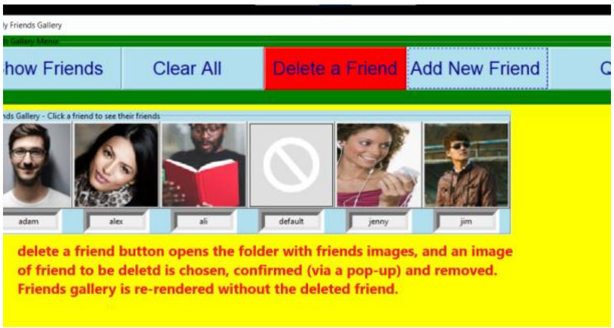
Are you sure you want to add a new friend?

pop-up to confirm to add or not new friend



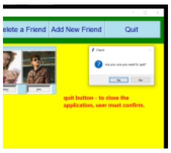
adding a new friend confirmed, friends gallery is re-rendered as part of adding.

Delete dialog when ‘Delete a Friend’ activated



delete a friend button opens the folder with friends images, and an image of friend to be deletd is chosen, confirmed (via a pop-up) and removed. Friends gallery is re-rendered without the deleted friend.

Quit confirmation



Are you sure you want to quit?

quit button - to close the application, user must confirm.

Rubric:

Criteria	Level of achievement				Mark
Initial GUI Window	Initial window fails to render or is an ad-hoc window with little relevance to the coursework brief (0 – 2)	Initial window renders correctly but the menu is ad-hoc with little attention to aesthetics (3 – 4)	Initial window renders correctly with good menu design – clear and clean spacing of all the buttons. No background images or colours applied. (5 – 7)	Initial window renders correctly with a menu frame that has all the buttons. The background colours and images have been used to good effect. (8 – 10)	10
Button: Show Friends	Gallery fails to render correctly, ad-hoc images may appear. Repeated press of the button will show unexpected behaviour and the app may crash. (0 – 6)	Gallery is partially rendered with a number of images missing. Repeated press of the button will show unexpected behaviour and the app may crash. (7 – 13)	Gallery renders correctly once. Repeated press of the button will show unexpected behaviour and the app may crash or gallery display will increase in size. (14 – 19)	Gallery is clearly and cleanly displayed with equal sized images that are buttons with names. Repeated press of the ‘Show Friends’ button is correctly managed and the button background changes when active. (20 – 25)	25
Button: friendButton	Friend image is rendered as a label and provides no interaction. (0 – 3)	Friend image is rendered as a button but does not render friend’s friend gallery. (4 -6)	Friend image is rendered as a button and friend’s friend gallery is rendered correctly without the ‘X’ button. (8 – 11)	Friend image is rendered as a button and friend’s friend gallery is rendered correctly with a ‘X’ button. ‘X’ button works as expected – clearing this gallery. (12 - 15)	15
Button: Clear All	Button is visible but inactive when clicked, failing to clear or crashes the app. (0 – 4)	Gallery is cleared and the frame used for image display is not destroyed so that further image displays are not possible. (5 – 9)	Gallery is cleared and the frame used is destroyed if and only if the gallery is on and the GUI defaults to initial display. Repeated presses are not managed and the app crashes. (10 – 12)	Gallery is cleared and the frame used is destroyed if and only if the gallery is on and the GUI defaults to initial view. Repeated press of the button is correctly managed and the app does not crash. Button background changes when active. (13 – 15)	15

Button: Delete a Friend	Not implemented or crashes the app. (0 -2)	Deletion of a friend is fully implemented incorporating file system utilities without a confirmation, however, re-rendering of the gallery still has the deleted friend showing. (3)	Deletion of a friend is fully implemented incorporating file system utilities without a confirmation and correct re-rendering of the gallery without the deleted friend. (4)	Deletion of a friend is fully implemented incorporating file system utilities with a confirmation and re-rendering of the gallery without the deleted friend. Button background changes when active. (5)	5
Button: Add New Friend	Not implemented or crashes the app. (0 - 2)	Addition of a friend is fully implemented incorporating file system utilities without a confirmation, however, re-rendering of the gallery does not include the new friend (3)	Addition of a friend is fully implemented incorporating file system utilities without a confirmation and correct re-rendering of the gallery with new friend (4)	Addition of a friend is fully implemented incorporating file system utilities with a confirmation and re-rendering of the gallery with new friend. Button background changes when active. (5)	5
Quit	Not implemented (0 – 1)	Crashes the app (2 - 3)	Fully implemented but no background change effective. (4)	Fully implemented with button background changes when active. (5)	5
pdf document	pdf document is poorly structured that fails to convey the design, implementation and evaluation. (0 – 1)	Basic pdf document with a structure that attempts to convey the design, implementation and evaluation. (2 - 3)	Good pdf document with a clear structure that conveys the design, implementation and evaluation. (4)	Excellent pdf document with a clear structure that fully conveys the design, implementation and evaluation. (5)	5
Enhanced features	Not attempted (0)	Enhanced features identified with some justification. (1 – 2)	Enhanced features identified with some justification and partial implementation (3-4)	Enhanced features identified with full justification and correct implementation (5)	5
Coding style	No evidence of any coding standards (0)	Variable names follow convention – camel case or snake case. (1 – 2)	Partial adherence to coding conventions and appropriate use of comments and referencing. (3-4)	Full adherence to coding conventions and appropriate use of comments and referencing. (5)	5
Testing	No testing undertaken or only 1 or 2 test cases.	Limited, ad-hoc testing with some evaluation.	Limited test cases fully documented with some justification. Results recorded as expected versus observed.	Test cases fully documented with full justification. Results recorded as expected versus observed.	5

	(0 -1)	(2 – 3)	(3-4)	(5)	

Submission details:

You must submit two files, one a pdf document with all your design and details of the tests carried out. The second file will be a zipped file with your python code and the images. File names should be your student id e.g.

w123123.pdf and w123123.zip

PYTHON scripts:

- **The code should be fully commented.**
- **All code used from other sources should be fully referenced.**