|  |  |  |  |
| --- | --- | --- | --- |
| **Requirements** | **Description** | **Implication** | **Tasks** |
| 1. | Must be able to target a folder containing photos, using buttons as a way to browse the folder in the Operating System | Making use of the buttons to obtain the contents in the folder and displaying the pictures in the GUI. | **Task 1**  Learn to operate and interpret values from QT. |
| 2. | Must be able to list the compatible image files (e.g \*.jpg, \*.png) in the folder created.   * Sort descending by file name; | Finding and creating the correct file path for the required images needed in the coursework. | **Task 2**  Developing the correct code that allows sorting algorithms for the necessary files. For example, sorting by descending the file by the name. |
|  | Must be able to list the compatible image files (e.g \*.jpg, \*.png) in the folder created.   * Sort ascending by file name; | Finding and creating the correct file path for the required images needed in the coursework. | Developing the correct code that allows sorting algorithms for the necessary files. For example, sorting by ascending the file by the name. |
|  | Must be able to list the compatible image files (e.g \*.jpg, \*.png) in the folder created.   * Sort descending by file date; | Finding and creating the correct file path for the required images needed in the coursework. | Developing the correct code that allows sorting algorithms for the necessary files. For example, sorting by descending the file by the date. |
|  | Must be able to list the compatible image files (e.g \*.jpg, \*.png) in the folder created.   * Sort ascending by file date; | Finding and creating the correct file path for the required images needed in the coursework. | Developing the correct code that allows sorting algorithms for the necessary files. For example, sorting by ascending the file by the date. |
| 3. | Classes with browser button, so user can navigate through the folder. | Allows the user to be able to move through the folders. | **Task 3**  The classes in the GUI should have buttons to allow the user prompt the folder navigation files. |
|  | They can select the class files (plain texts with extensions “\*. names”. It must correspond to the class. Types of class car, dog, cat, person | Allows the user to be able to move through the folders. | The classes in the GUI should have buttons to allow the user prompt the folder navigation files. These classes files in the folder should be saved as plain text. With the extension” .names. Mus also correspond with the types of class car, dog, cat person |
| 4. | Classes should be listed in a classes plane. | It should have structured file line numbers which are not easily changed in the classes file. So that they can’t be changed by the user | **Task 4**  Only have structured file line numbers which can’t change |
|  | Should be able to sort in ascending order.  Must be preserved in this order. | It should have structured file line numbers which are not easily changed in the classes file. So that they can’t be changed by the user | Classes must be listed in ascending order, it can be executed with the dates of the file names |
|  | Should be able to sort in descending order.  Must be preserved in this order. | It should have structured file line numbers which are not easily changed in the classes file. So that they can’t be changed by the user | Classes must be listed in descending order, it can be executed with the dates of the file names |
| 5. | Must be able to allow the user to add classes | The classes files can be added and removed by the user. This give the user the freedom to do this in the GUI. By doing this they can append them to their various classes’ files | **Task 5**  The GUI should allow the user to add classes. By creating specific algorithms to implement this function. |
|  | Must be able to allow the user to remove classes | The classes files can be added and removed by the user. This give the user the freedom to do this in the GUI. By doing this they can append them to their various classes’ files | The GUI should allow the user to remove classes. By creating specific algorithms to implement this function. |
|  | Must be able to allow the user to append them to the classes file. | The classes files can be added and removed by the user. This give the user the freedom to do this in the GUI. By doing this they can append them to their various classes’ files | The GUI should allow the user to append classes. By creating specific algorithms to implement this function. |
| 6. | Must be able to select and use one of the various shapes:   1. Triangle 2. Square/rectangle 3. Trapezium 4. Polygon (with up to 8 points); | Using the Open CV, we are able to draw and implicate the shapes various shapes into our code. | **Task 6**  Learn to operate and interpret values from OPEN CV. |
| 7. | Must only use shapes provided for annotating the given images. Users must also be able to select any images and draw a shape on the top of the image. Borders must be visible. | Making use of the Open CV, relevant information will be shared in OPEN CV. Doing so, this will allow code to be easily accessed by other platforms. | **Task 7**  Ensure that all the code within the system is shared in the OPEN CV. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Requirements** | **Description** | **Implication** | **Tasks** |
| 8. | The annotations(shapes) should be displayed on the image. | These shapes will be displayed on the GUI after the images have been implemented. | **Task 8**  Improve the code which enables the shapes to be implemented properly. |
| 9. | The annotation file/file name select must have these following options:   1. Open/load annotation files (\*. annotations) | The files should be able to load and open the annotations in the GUI. | **Task 9**  Developing the correct code that allows open/loading algorithms for the necessary files. For example, opening and loading the file by the name. |
|  | The annotation file/file name select must have these following options:  2. Save the annotation file. Warning must be displayed when the user chooses to overwrite the file. | The files should be able to save the annotations in the GUI and should be seen in the file extension (. \*annotations). It should throw an error message if the user chooses over writes a file | Developing the correct code that allows save algorithms for the necessary files. For example, opening and loading the file by the name. It should throw an error message if the user chooses over writes a file |
|  | The annotation file/file name select must have these following options:  3. Change the name of an existing file | The files should be able to change the annotations in the GUI and should be seen in the file extension. | Developing the correct code that allows change algorithms for the necessary files.  For example, creating code that can change an existing file name. |
|  | The annotation file/file name select must have these following options:   1. Must have a hierarchical data format 5 (HDF5 standard). | Hierarchical Data Format is a set of file formats designed to store and organize large amounts of data | Developing the correct code with allows us to allows us to store large datasets using HDF5. |
|  | The annotation file/file name select must have these following options:   1. The data must be stored in each of the annotation file:   Number of annotated images for each image | By using the right dataset this will enable us to be able to annotate the number of images in the GUI for each of the images | Developing the correct code that allows us to use correct datasets. |
|  | The annotation file/file name select must have these following options:  5b. The data must be stored in each of the annotation file:  Image file name | Using the correct file name would make it easier to locate and store images. | Developing the correct code that allows us to use correct datasets. |
|  | The annotation file/file name select must have these following options:  5c. The data must be stored in each of the annotation file:  Number of shapers per image  Shape type  Point\_1 (x,y);  Point\_2 (x,y);  Point\_n (x,y); | Make an annotation file. | Developing the correct code that allows us to use correct datasets. |
| 10. | The image selected should be displayed in an image pane. | The user will be able to select the image. In an image pane. This enables them to know what images have been selected. | **Task 10**  Ensure the correct file paths have been created. |
| 11. | Must perform shape operation using the move.   1. Increase size | The user will be able to manually control the GUI and see what happens in the GUI | **Task 11**  Develop a Navigational algorithm for the system. That allows the shape to increase in size. |
|  | Must perform shape operation using the move.  2. Move the vertices of the polygons | The user will be able to manually control the GUI and see what happens in the GUI | Develop a Navigational algorithm for the system. That allows the user to move the vertices of the polygons |
|  | Must perform shape operation using the move.  3. Delete shape | The user will be able to manually control the GUI and see what happens in the GUI | Develop a Navigational algorithm for the system. That allows the user to delete the shape |
|  | Must perform shape operation using the move.  4. Copy and paste shape | The user will be able to manually control the GUI and see what happens in the GUI | Develop a Navigational algorithm for the system. That allows the user to copy and paste the shape in the GUI |
|  | Must perform shape operation using the move.  5.Visualise the name of the class on the top of the shape | The user will be able to manually control the GUI and see what happens in the GUI | Develop a Navigational algorithm for the system. That allows the shape to display its class |
| 12. | Must use threads to autosave files. | This saves the user time for when they manipulate images in the GUI. | **Task 12**  Improve algorithms of the systems. |
| 13. | Must use the data structures developed in 1st term for storing data in the memory | By using the appropriate data structure this will enable the project to have a structure  E.g. binary Trees, linked lists and Hash tables. | **Task 13**  Look for the best/fastest data structure to use. |
| 14. | Must use a sort and a search algorithm | This allows a more structural approach to the project. It allows the user to find things within the GUI easily | **Task 14**  Developing the correct code that allows sorting/searching algorithms for the necessary files. |