1 Introduzione

"Exif Viewer" è un'applicazione desktop realizzata in python che consente di estrarre da un'immagine caricata dal proprio computer i così detti metadati EXIF (acronimo di Exchangeable Image File Format). Questi ultimi rappresentano una serie di informazioni riguardo alla fotografia scattata rappresentati come coppie chiave valore.

2 Realizzazione

Per realizzare l'applicazione in questione è stato usato PyQT5, una libreria che permette l'utilizzo in python del framework Qt GUI.

Di seguito è riportato il diagramma UML che illustra la struttura del software.

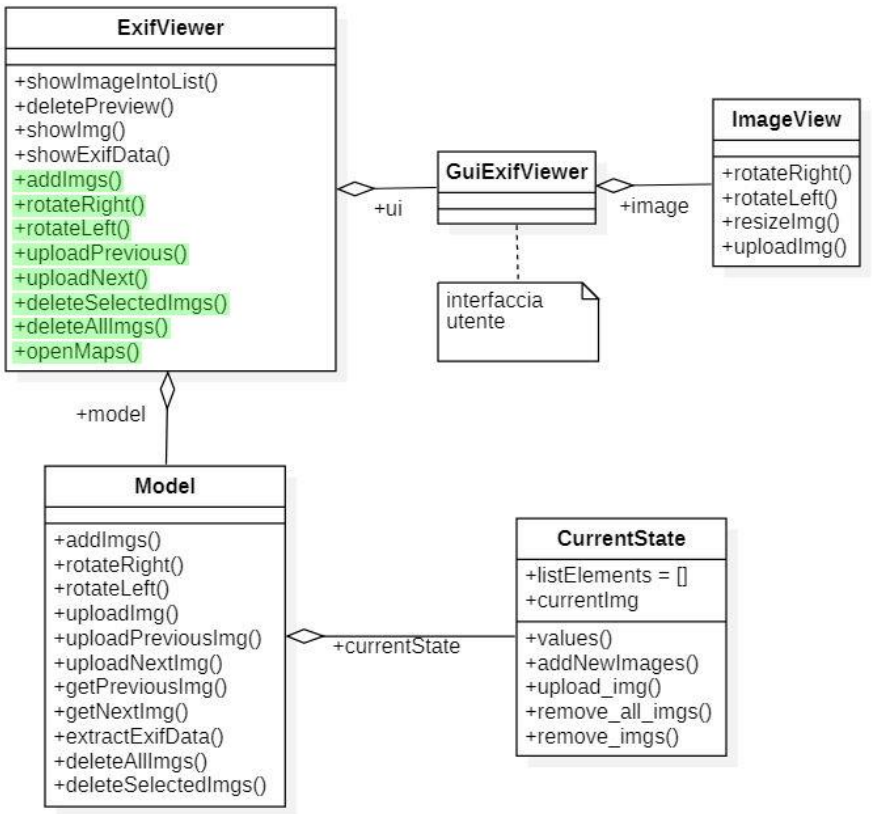

Figura 1 - Diagramma UML. I metodi evidenziati sono quelli invocati direttamente dall'utente tramite interazione con l'interfaccia.

La classe **GuiExifViewer** implementa l'interfaccia utente, realizzata tramite GUI, pertanto contiene tutti i dettagli implementativi relativi alla realizzazione grafica dell'interfaccia.

La classe **CurrentState** contiene lo stato corrente dell'applicazione, ovvero la lista *listElements* all'interno della quale è tenuta traccia di tutte le immagini caricate e il parametro *currentImg* che indica di quale immagine è attualmente mostrata l'anteprima . Si hanno poi una serie di metodi che manipolano tali elementi, tra cui *addNewImages()* che aggiunge le nuove immagini caricate alla lista, *remove_imgs()* che rimuove una o più immagini selezionate, *remove_all_imgs()* che svuota completamente la lista.

La classe **Model** definisce tutte quelle operazioni che richiedono la manipolazione delle immagini presenti all'interno dell'applicazione. Il metodo *addImgs()* consente di caricare una o più nuove immagini, a sua volta richiamerà il metodo *addNewImages()* della classe CurrentState che le aggiungerà alla lista. Nel caso in cui sia il primo caricamento che si effettua viene richiamato anche il metodo *Model.uploadImg()* che estrae i dati exif ed a sua volta richiama *CurrentState.upload_img()* che setta *currentImg* con la prima immagine tra quelle appena importate. Il metodo *deleteSelectedImgs()* prima cancella una o più immagini selezionate rimuovendole dalla lista tramite il metodo *remove_imgs()* della classe CurrentState e poi reimposta l'anteprima ottenendo tramite *searchUpImg()* la prima immagine della lista che non è stata cancellata e caricandola con *uploadImg()*. Il metodo *deleteAllImgs()* elimina tutte le immagini presenti svuotando la lista *listElements* della classe CurrentState. I metodi *uploadPreviousImg()* e *uploadNextImg()* si occupano di caricare la visualizzazione rispettivamente dell'immagine precedente e dell'immagine successiva nella lista. Entrambi richiameranno *uploadImg()* che si occupa sia di settare l'attributo *currentImg* che dell'estrazione dei dati EXIF relativi all'immagine corrente.

La classe **ImageView** definisce i metodi *rotateLeft()* e *rotateRight()* che si occupano della rotazione dell'immagine visualizzata di 90° rispettivamente verso sinistra e verso destra. Espone inoltre il metodo *uploadImg()* che setta l'anteprima dell'immagine.

La classe **ExifViewer** espone una serie di metodi che sulla base dell'interazione con l'interfaccia utente richiamano i corrispondenti metodi della classe Model e fa uso dei valori di ritorno per richiamare i metodi che vanno a modificare l'interfaccia. Nell'accezione del pattern MVC la classe ExifViewer ricopre i ruoli di View e Controller. Il metodo *addImgs()* viene richiamato quando si vuole fare l'upload di nuove immagini. Richiama il metodo *Model.addImgs()* e usa i valori di ritorno per richiamare *showImageIntoList()* che aggiorna la visualizzazione aggiungendo le nuove immagini inserite. Nel caso in cui sia il primo caricamento richiama anche *showImg()* che setta l'anteprima ed a sua volta richiama *showExifData()* per mostrare la tabella dei dati exif. I metodi *deleteSelectedImgs()* e *deleteAllImgs()* richiamano entrambi l'omonimo metodo della classe Model e usano i valori di ritorno per passarli come parametro di *deleteImageIntoListAndPreview()* per rimuovere dalla vista le immagini cancellate. I metodi *uploadPreviousImg()* e *uploadNextImg()* richiamano gli omonimi metodi della classe Model e usano i valori ritornati per richiamare *showImg()*. I metodi *rotateRight()* e *rotateLeft()* sono booleani che ritornano *True* se è presente un'immagine corrente e nel caso richiamano i corrispondenti metodi della classe ImageView. Il doppio click sull'immagine nella lista richiama il metodo *uploadImage()* che fa uso dell'immagine ritornata da *Model.uploadImage()* per richiamare *showImg()* e settare l'anteprima. Al click dell'icona di Google Maps si invoca il metodo *openMaps()* che va a leggere il parametro *maps_url* settato dal metodo *showExifData()*. Se l'immagine corrente ha tra i suoi metadati le coordinate gps si aprirà una finestra del browser con Google Maps impostato sulla località in questione.

Di seguito è riportato uno screenshot che illustra l'interfaccia utente dell'applicazione.

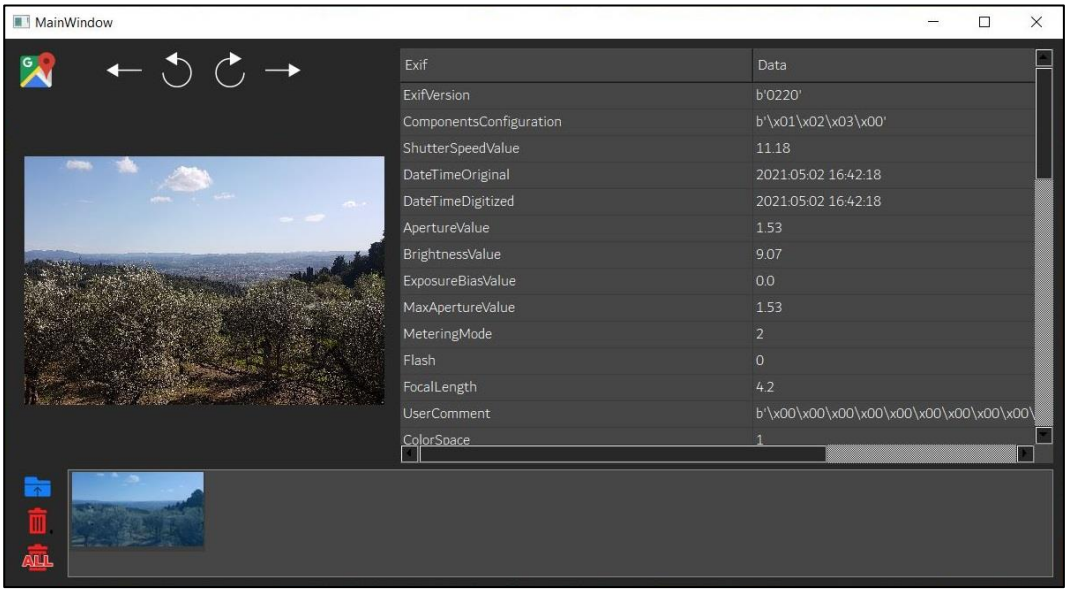

Figura 2 - Screenshot dell'applicazione