Wordcloud generator

# Indice

1 Indice 2

2 Introduzione 3

2.1 Informazioni sul progetto 3

2.2 Abstract 3

2.3 Scopo 3

3 Analisi 4

3.1 Analisi del dominio 4

3.2 Analisi e specifica dei requisiti 4

3.3 Use case 7

3.4 Pianificazione 7

3.5 Analisi dei mezzi 9

3.5.1 Software 9

3.5.2 Hardware 9

4 Progettazione 9

4.1 Design dell’architettura del sistema 9

4.2 Design dei dati e database 9

4.3 Design delle interfacce 12

4.4 Design procedurale 12

5 Implementazione 13

6 Test 13

6.1 Protocollo di test 13

6.2 Risultati test 15

6.3 Mancanze/limitazioni conosciute 15

7 Consuntivo 15

8 Conclusioni 17

8.1 Sviluppi futuri 17

8.2 Considerazioni personali 17

9 Bibliografia 18

9.1 Bibliografia per articoli di riviste: 18

9.2 Bibliografia per libri 18

9.3 Sitografia 18

10 Glossario 19

11 Indice delle figure 19

12 Allegati 19

# Introduzione

## Informazioni sul progetto

In questo capitolo raccogliere le informazioni relative al progetto, ad esempio:

* Allievi: Alessandro Curiale, Christian Monga, Edoardo Ratti
* Docente: Geo Petrini
* SAMT sezione informatica modulo 306
* 27.01.2023 – 05.05.2023

## Abstract

È una breve e accurata rappresentazione dei contenuti di un documento, senza notazioni critiche o valutazioni. Lo scopo di un abstract efficace dovrebbe essere quello di far conoscere all’utente il contenuto di base di un documento e metterlo nella condizione di decidere se risponde ai suoi interessi e se è opportuno il ricorso al documento originale.

Può contenere alcuni o tutti gli elementi seguenti:

* **Background/Situazione iniziale**
* **Descrizione del problema e motivazione**: Che problema ho cercato di risolvere? Questa sezione dovrebbe includere l'importanza del vostro lavoro, la difficoltà dell'area e l'effetto che potrebbe avere se portato a termine con successo.
* **Approccio/Metodi**: Come ho ottenuto dei progressi? Come ho risolto il problema (tecniche…)? Quale è stata l’entità del mio lavoro? Che fattori importanti controllo, ignoro o misuro?
* **Risultati**: Quale è la risposta? Quali sono i risultati? Quanto è più veloce, più sicuro, più economico o in qualche altro aspetto migliore di altri prodotti/soluzioni?

Esempio di abstract:

*As the size and complexity of today’s most modern computer chips increase, new techniques must be developed to effectively design and create Very Large-Scale Integration chips quickly. For this project, a new type of hardware compiler is created. This hardware compiler will read a C++ program, and physically design a suitable microprocessor intended for running that specific program. With this new and powerful compiler, it is possible to design anything from a small adder, to a microprocessor with millions of transistors. Designing new computer chips, such as the Pentium 4, can require dozens of engineers and months of time. With the help of this compiler, a single person could design such a large-scale microprocessor in just weeks.*

## Scopo

Il nostro progetto ha lo scopo di ‘‘creare delle immagini’’, ovvero in base ad un’immagine e altre opzioni disponibili nell’applicativo, in sé un’immagine dopo un’elaborazione dovrà comparire modificata. L’applicativo deve catturare il soggetto o i soggetti dell’immagine, evidenziarne i bordi per poi sostituire il contenuto con delle scritte relative ad un testo di input, un file txt oppure una fonte url.

# Analisi

## Analisi del dominio

Questo capitolo dovrebbe descrivere il contesto in cui il prodotto verrà utilizzato, da questa analisi dovrebbero scaturire le risposte a quesiti quali ad esempio:

* Background/Situazione iniziale
* Quale è e come è organizzato il contesto in cui il prodotto dovrà funzionare?
* Come viene risolto attualmente il problema? Esiste già un prodotto simile?
* Chi sono gli utenti? Che bisogni hanno? Come e dove lavorano?
* Che competenze/conoscenze/cultura posseggono gli utenti in relazione con il problema?
* Che conoscenze teoriche bisogna avere/acquisire per poter operare efficacemente nel dominio?

Ultimamente quando vediamo manifesti e pubblicità ci sembrano un po’ spogli, senza fantasia,

allora il nostro team di sviluppo ha pensato a come dare un po’ inventiva al marketing.

Idealmente vogliamo sviluppare un classico worldcloudgenerator per poterlo fornire a scuole e ditte.

Sicuramente alle scuole tornerebbe utile per fare brainstorming, dunque aumentare l’interesse degli allievi nell’esprimere le proprie opinioni, ma il nostro obbiettivo principale è quello di tornare utili alle ditte per fornire una migliore pubblicità hai clienti.

Anche se questo tipo di applicativo è già presente a noi interessa riprodurlo per poterlo manipolare a nostro piacimento, dunque aggiungere sviluppi futuri gestire nuove richieste dei clienti in modo da orientarci maggiormente nel mondo del marketing.

Il software è molto semplice, chiunque è capace di utilizzare uno smartphone è anche in grado di utilizzarlo. Inoltre forniremo un breve manuale di utilizzo per coloro che avranno dubbi.

Prima di tutti per entrare nell’ottica del dominio occorre saper scegliere il linguaggio di programmazione, nel nostro caso abbiamo deciso python, questo perché è molto semplice gestire le librerie, dunque interfacce e strumenti, utilizzeremo opencv per gestire le immagini, kivy per quanto riguarda le interfacce e tkinter per completare dove kivy non riesce a arrivare.

……………….

## Analisi e specifica dei requisiti

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-001 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Upload immagine | | | | |
| **Note** | L’utente trascina e rilascia nello spazio apposito l’immagine su cui lavorare | | | | |
| **Sotto requisiti** | | | | | |
| **001** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-002 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Parti da mantenere fisse dell’immagine | | | | |
| **Note** | L’utente sceglie quali parti trasformare in parole e quali no | | | | |
| **Sotto requisiti** | | | | | |
| **001** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-003 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | L’utente inserisce le parole da rappresentare nelle immagini | | | | |
| **Note** | L’utente sceglie le parole da inserire nell’immagine | | | | |
| **Sotto requisiti** | | | | | |
| **001** | L’utente può inserirle tramite testo | | | | |
| **002** | L’utente può inserirle tramite file | | | | |
| **003** | L’utente può inserirle tramite URL di una pagina che verrà scaricata | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-004 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Si può personalizzare il font family | | | | |
| **Note** | L’utente può scegliere il font family del testo scritto nell’immagine | | | | |
| **Sotto requisiti** | | | | | |
| **001** | L’utente sceglie da un drop down il font family da applicare | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-005 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Il colore delle parole si adatta con il colore dell’immagine | | | | |
| **Note** | Il colore delle parole deve rispecchiare il colore dell’immagine di base | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-006 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Set di parole bloccate | | | | |
| **Note** | Il programma contiene già un set di parole bloccate che non verranno mostrate nell’immagine | | | | |
| **Sotto requisiti** | | | | | |
| **001** | L’utente inserisce delle parole che non vuole che vengano mostrate | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-007 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Parole con importanza maggiore | | | | |
| **Note** | L’utente può inserire delle parole che devono essere mostrate più grande di come sarebbero normalmente | | | | |
| **Sotto requisiti** | | | | | |
| **001** | L’utente sceglie da un drop down il font family da applicare | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-008 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Aggiunta del bordo all’immagine | | | | |
| **Note** | L’utente può scegliere di aggiungere un bordo all’immagine creata | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Il bordo può avere un colore | | | | |
| **002** | Il bordo deve avere uno spessore a scelta | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-009 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Aggiornamento in tempo reale dell’immagine | | | | |
| **Note** | L’immagine si deve aggiornare in tempo reale seguendo le varie modifiche apportate dall’utente | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-010 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Scelta risoluzione immagine | | | | |
| **Note** | Si può scegliere la risoluzione dell’immagine una volta scaricata | | | | |
| **Sotto requisiti** | | | | | |
| **001** | La risoluzione dell’immagine ha delle proporzioni definite da cui scegliere | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-011 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Scelta formato dell’immagine | | | | |
| **Note** | L’utente può scegliere il formato con cui deve essere scaricata l’immagine | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Il formato può essere .jpg o .png o .webp | | | | |

**Spiegazione elementi tabella dei requisiti:**

**ID**: identificativo univoco del requisito

**Nome**: breve descrizione del requisito

**Priorità**: indica l’importanza di un requisito nell’insieme del progetto, definita assieme al committente. Ad esempio, poter disporre di report con colonne di colori diversi ha priorità minore rispetto al fatto di avere un database con gli elementi al suo interno. Solitamente si definiscono al massimo di 2-3 livelli di priorità.

**Versione**: indica la versione del requisito. Ogni modifica del requisito avrà una versione aggiornata.

Sulla documentazione apparirà solamente l’ultima versione, mentre le vecchie dovranno essere inserite nei diari.

**Note**: eventuali osservazioni importanti o riferimenti ad altri requisiti.

**Sotto requisiti**: elementi che compongono il requisito.

## Use case

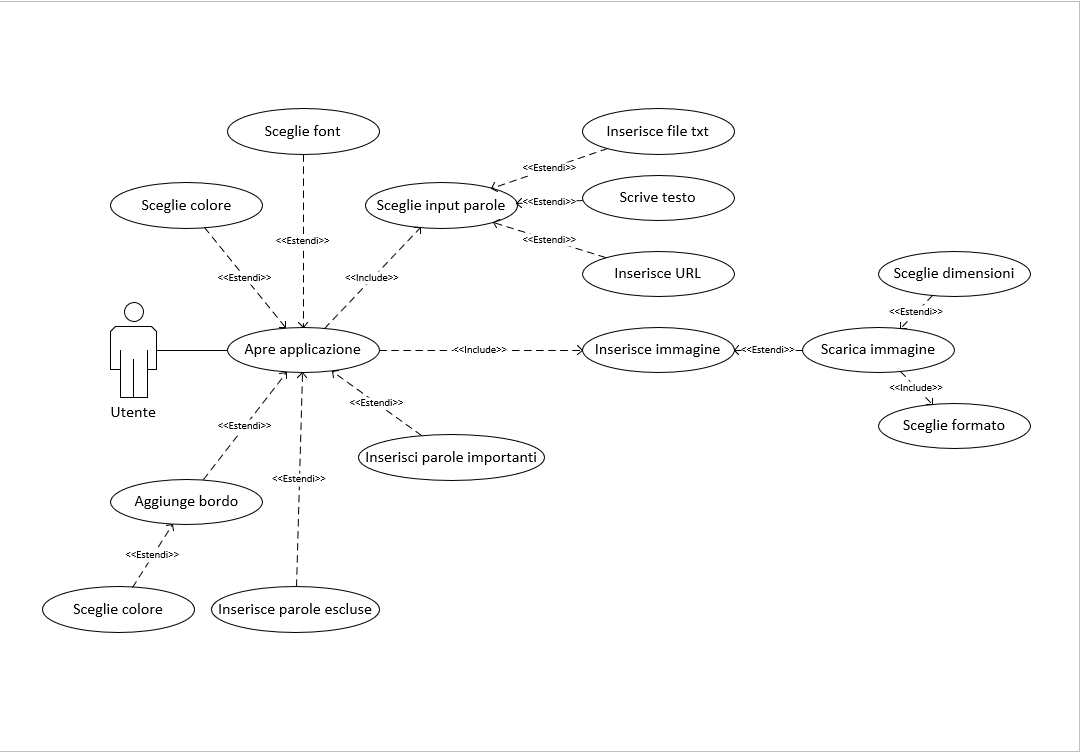


Figura UseCase

## Pianificazione

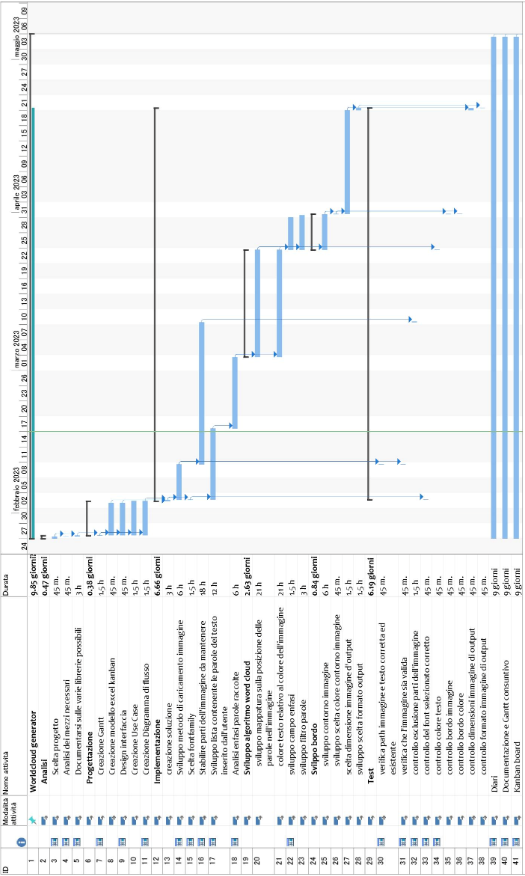


Figura Esempio di diagramma di Gantt

Se si usano altri metodi di pianificazione (p.es. scrum), dovranno apparire in questo capitolo.

KANBAN BOARD

Stile orizzontale



Figura Esempio di diagramma di Gantt

## Analisi dei mezzi

### Software

Python 3.10.9

Kivy 1.0.9

OpenCV version

Tkinter version

Maptolib

### Hardware

Su quale piattaforma dovrà essere eseguito il prodotto? Che hardware particolare è coinvolto nel progetto? Che particolarità e limitazioni presenta? Che HW sarà disponibile durante lo sviluppo?

Il nostro prodotto è pensato per essere utilizzato da un computer, infatti disponiamo di tre computer.

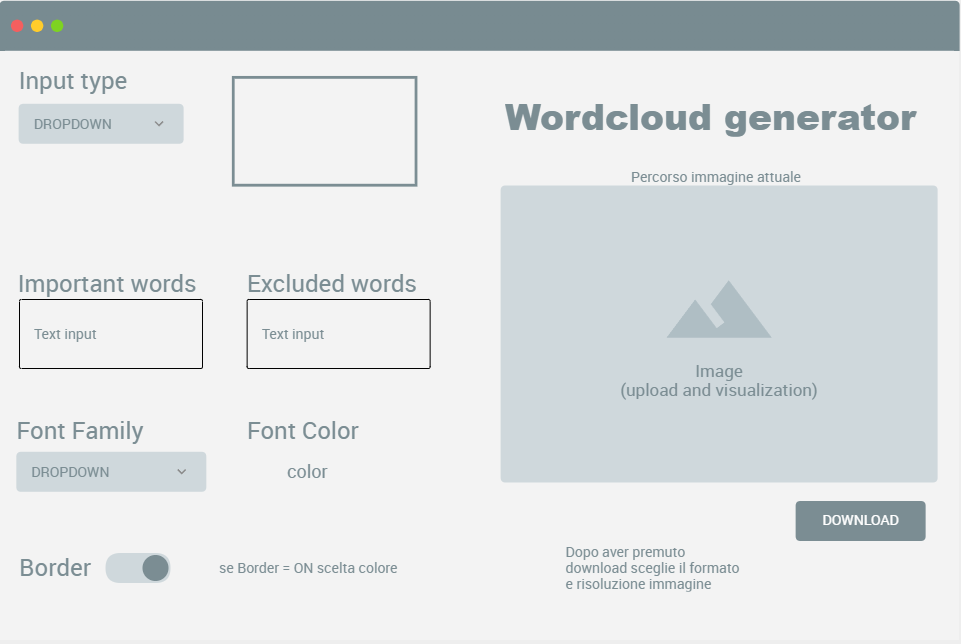
# Progettazione

## Design dell’architettura del sistema

Descrive:

* La struttura del programma/sistema lo schema di rete...
* Gli oggetti/moduli/componenti che lo compongono.
* I flussi di informazione in ingresso ed in uscita e le relative elaborazioni. Può utilizzare *diagrammi di flusso dei dati* (DFD).
* Eventuale sitemap

## Design delle interfacce



Questa è la nostra unica interfaccia del programma, da qui si potranno fare tutte le varie azioni disponibili.

## Design procedurale

Descrive i concetti dettagliati dell’architettura/sviluppo utilizzando ad esempio:

* Diagrammi di flusso e Nassi.
* Tabelle.
* Classi e metodi.
* Tabelle di routing
* Diritti di accesso a condivisioni …

### Diagramma di flusso

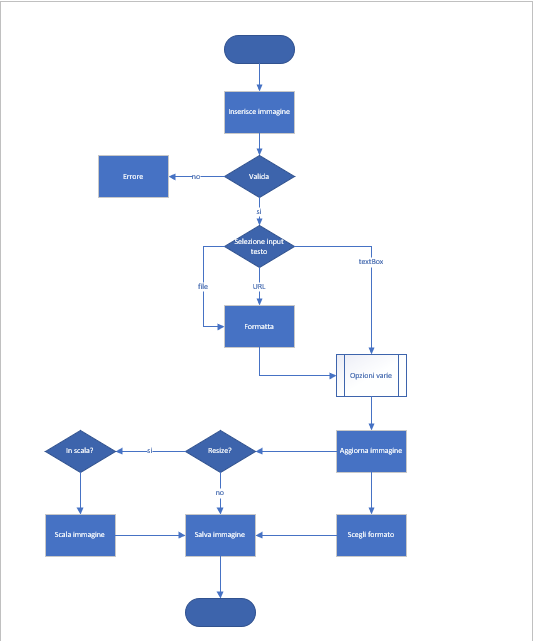


Figura Diagramma di flusso

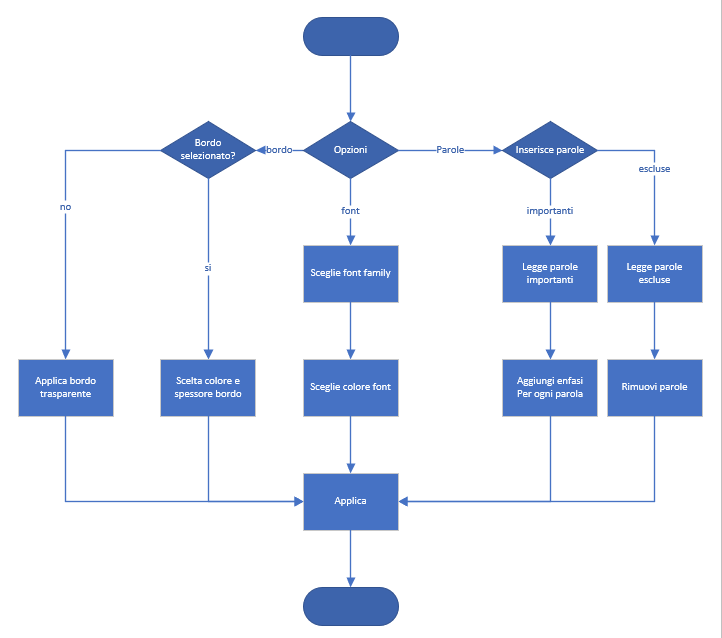


Figura Diagramma opzioni varie

# Implementazione

## Controllo path per immagine

L’utente può inserire la path per l’immagine da utilizzare e per questo c’è bisogno di controllare se questa path è effettivamente un’immagine o meno.

Noi abbiamo fatto anche in modo che, nello stesso momento in cui l’utente inserisce un nuovo carattere nel “TextInput” per la path, il programma ricerca automaticamente se esiste l’immagine e, quando la trova, la mostra a schermo.

Per fare ciò abbiamo utilizzato la proprietà “on\_text” che ci permette di richiamare un metodo ogni volta che viene inserito un nuovo carattere.

Questo metodo, da noi chiamato “visualizer”, richiama subito un nuovo metodo per poter leggere e memorizzare la path inserita dall’utente. Dopo averla memorizzata avviene un controllo di essa, verifichiamo che:

1. La path sia valida
2. La path sia un file
3. Il file sia un’immagine

Se questi tre controlli vanno a buon fine allora la path verrà utilizzata per mostrare l’immagine mentre se falliscono verrà mostrata un’immagine di default creata da noi che mostra semplicemente la scritta “Image not found”.

## Algoritmo di floading

La classe “imagepartselector” possiede 6 variabili globali, 3 sono dedicate alle path, due hai colori e una per raggiungere il pannello download da ovunque.

BORDER\_COLOR = (100,100,100)

FLOAD\_COLOR = (0,0,0)

DOWNLOAD = -1

START\_PATH = "./pictures/Immagine.png"

MASK\_PATH = "./pictures/ImmagineBN.png"

TEMP\_PATH = "./pictures/CopiaImmagineEdo.png"

“Build image” è il metodo che si occupa di preparare l’immagine alla selezione da parte dell’utente delle aree sulla quale si desidera applicare l’effetto world cloud.

Esso parte creando l’immagine sulla quale il programma lavorerà e in seguito quella temporanea vuota dedicata alla maschera, ovvero il prodotto della classe.

Poi si occuperà di tracciare un primo bordo, in modo tale da preparare il lavoro al metodo “updateMask”

Il metodo “updateMask” si occupa di: tracciare nuovamente il bordo, calcolare le coordinate dell’immagine dato che l’evento “touch\_down” recupera le coordinate sull’asse sbagliato (piano cartesiano), infine di ricaricare l’immagine dopo che è stata evidenziata.

“HighlightArea”, questo metodo iterativo sostituisce il metodo ricorsivo classico di floading data una limitazione di python con il numero massimo di ripetizioni ricorsive.

Esso occupa evidenziare la zona desiderata dall’utente con l’aiuto dei 3 metodi dall’intuitivo scopo: “isHighlighedPixel”, “isBorder” e “isInArea”

Questi esattamente come dal nome si occupano di controllare se:

1. Un determinato pixel dell’immagine è colorato
2. Una coordinata si trova nel bordo della figura tracciata
3. Una coordinata si trova nello spazio dell’immagine

## Proprietà del bordo

Ogni volta che viene chiamato il metodo “updateImage” l’immagine a sua volta viene ricaricata modificando il colore e la tolleranza in base alle esigenze dell’utente.

## Metodo di input per le parole

L’utente può inserire le parole che dovranno venir rappresentate nell’immagine in tre modi:

* Tramite testo semplice
* Tramite file
* Tramite URL

Per questa scelta abbiamo implementato uno “Spinner” con queste tre opzioni, ogni volta che l’utente seleziona un tipo di input viene richiamato un metodo che prende come argomento il tipo di input selezionato.

In questo metodo viene controllato inizialmente quale tipo di input è stato selezionato e in base a questo viene chiamato il metodo apposito per poter memorizzare il testo.

Per memorizzare il testo da un file abbiamo semplicemente usato dei metodi standard di python per aprire il file, leggerlo e salvarlo all’interno di una variabile.

Per memorizzare il testo da una pagina web, invece, abbiamo installato BeautifulSoup che ci permette di aprire la pagina e di salvare tutto il contenuto, evitando i vari tag HTML e Javascript, in modo piuttosto semplice.

Una volta memorizzate le parole abbiamo fatto in modo che, tramite un ciclo, tutti i caratteri speciali (caratteri che non sono lettere) vengano esclusi e sostituiti con degli spazi. Dopo aver fatto questa modifica tutte le varie parole vengono separate e controllate singolarmente. Questo controllo consiste nel vedere se le parole presenti sono delle parole da escludere o meno, se risultano come parole “vietate” esse verranno semplicemente cancellate.

Per finire bisogna calcolare l’enfasi di queste parole prima di poterle inserire nell’immagine. Per farlo abbiamo creato un nuovo metodo che, tramite un array, memorizza la parola e quante volte è presente all’interno del testo. Una volta riempito questo array esso viene ordinato in modo crescente così da avere alla prima posizione la parola più presente.

## Proxy per la richiesta via URL

Visto che questo progetto lo abbiamo svolto a scuola abbiamo dovuto gestire il proxy.

## Schermata per il download

Al posto di mostrare le varie opzioni per il download direttamente sull’interfaccia principale si può creare una nuova schermata apposita.

Per accedere a questa schermata si deve utilizzare il pulsante “download” presente nella schermata principale e, una volta premuto, si accederà a quella per il download.

Per fare questo collegamento si deve utilizzare la libreria “screenmanager”.

Inizialmente si deve creare una nuova classe che rappresenterà la nuova schermata: “DownloadScreen”. Una volta creata bisogna andare nel metodo “build” della classe “WordCloudApp” per poter creare lo “ScreenManager”. All’interno di esso si devono aggiungere le classi che rappresentano delle interfacce, dandogli un nome.

A questo punto, nel file wordcloud.kv, bisogna far sì che, tramite la proprietà “on\_release”, quando si premeranno i pulsanti corretti la scena si sposterà sullo “Screen” selezionato.

## Schermata per la modifica dell’immagine

In questa schermata l’utente potrà scegliere le parti dell’immagine da mantenere e quelle da modificare.

Per crearla abbiamo creato una nuova classe chiamata “ImageModifier”, questa classe contiene tutti i metodi necessari per la realizzazione della nuova schermata.

Nel costruttore della classe abbiamo inserito il codice per leggere l’immagine e convertirla in scala di grigi, una volta convertita abbiamo creato un’immagine binaria e grazie ad essa abbiamo trovato i vari contorni presenti all’interno dell’immagine.  
A questo punto il programma prende l’immagine originale di base, ci disegna sopra i vari bordi e crea una nuova immagine, che verrà salvata come “imageMod.png”. Questa immagine verrà poi mostrata a schermo.

All’interno del file “wordcloud.kv” abbiamo aggiunto una nuova sezione chiamata “ImageModifier” che conterrà i vari componenti per la modifica dell’immagine. Dentro a questa sezione abbiamo inserito un “Image”, uno “Slider” e un “Button”.

Lo slider permette di cambiare la tolleranza per la selezione dei bordi dell’immagine mentre il bottone permette di ritornare all’interfaccia principale.

# Test

## Protocollo di test

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-001  REQ-001 | **Nome** | Test upload dell’immagine |
| **Descrizione** | Caricare un’immagine sull’applicazione | | |
| **Prerequisiti** | Poter avviare l’applicazione | | |
| **Procedura** | 1. Avviare l’applicazione 2. Trascinare e rilasciare nello spazio apposito l’immagine 3. Controllare se l’immagine viene visualizzata a schermo | | |
| **Risultati attesi** | L’immagine viene visualizzata a schermo | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-002  REQ-002 | **Nome** | Test delle parti d’immagine da non cambiare |
| **Descrizione** | Testare che l’utente possa scegliere le parti da non cambiare all’interno dell’immagine | | |
| **Prerequisiti** | Aver verificato che funzioni l’inserimento dell’immagine | | |
| **Procedura** | ……. | | |
| **Risultati attesi** | L’utente può scegliere le parti che non vengono modificate ed esse non verranno effettivamente toccate alla generazione della nuova immagine con le parole | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-003  REQ-003 | **Nome** | Test delle parole inserite tramite input di testo |
| **Descrizione** | Testare che funzioni il metodo di inserimento manuale da parte dell’utente delle parole | | |
| **Prerequisiti** | Aver verificato che funzioni l’inserimento delle immagini.  Aver implementato il drop down di scelta del metodo di input delle parole | | |
| **Procedura** | 1. Scegliere l’opzione di input tramite testo inserito dall’utente 2. Scrivere all’interno del text box apposito 3. Controllare che l’immagine contenga tutte le parole 4. Scrivere anche parole bloccate e controllare che non vengano mostrate | | |
| **Risultati attesi** | L’utente inserisce un testo all’interno dello spazio apposito e nell’immagine compaiono tutte le parole inserite tranne quelle bloccate | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-004  REQ-003 | **Nome** | Test delle parole inserite tramite file di testo |
| **Descrizione** | Testare che funzioni il metodo di inserimento delle parole tramite un file di testo | | |
| **Prerequisiti** | Aver verificato che funzioni l’inserimento delle immagini.  Aver implementato il drop down di scelta del metodo di input delle parole | | |
| **Procedura** | 1. Scegliere l’opzione di input con l’upload del file di testo 2. Rilasciare nello spazio apposito un file di testo contenente delle parole 3. Controllare che l’immagine contenga tutte le parole presenti all’interno del file tranne quelle bloccate | | |
| **Risultati attesi** | L’utente inserisce un file di testo all’interno dello spazio apposito e nell’immagine compaiono tutte le parole inserite tranne quelle bloccate | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-005  REQ-003 | **Nome** | Test delle parole inserite tramite URL di una pagina da scaricare |
| **Descrizione** | Testare che funzioni il metodo di inserimento delle parole tranne | | |
| **Prerequisiti** | Aver verificato che funzioni l’inserimento delle immagini.  Aver implementato il drop down di scelta del metodo di input delle parole | | |
| **Procedura** | 1. Scegliere l’opzione di input tramite URL di una pagina 2. Inserire un URL all’interno dello spazio apposito 3. Controllare che l’immagine contenga tutte le parole tranne i vari tag HTML e le parole bloccate | | |
| **Risultati attesi** | L’utente inserisce un URL all’interno dello spazio apposito e nell’immagine compaiono tutte le parole presenti in quella pagina tranne quelle bloccate e i tag HTML | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-006  REQ-004 | **Nome** | Test del font family corretto |
| **Descrizione** | Testare che una volta selezionato un tipo di font family esso venga effettivamente utilizzato | | |
| **Prerequisiti** | Aver verificato che funzioni l’inserimento delle immagini.  Aver verificato il funzionamento dei vari input delle parole | | |
| **Procedura** | 1. Inserire un’immagine 2. Inserire del testo 3. Selezionare un font family 4. Controllare che sull’immagine venga effettivamente utilizzato il font family selezionato | | |
| **Risultati attesi** | L’utente sceglie il font e quest’ultimo viene effettivamente utilizzato nell’immagine creata | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-007  REQ-005 | **Nome** | Test del colore delle parole |
| **Descrizione** | Testare che il colore delle parole rispecchi quello dell’immagine iniziale | | |
| **Prerequisiti** | Aver verificato che funzioni l’inserimento delle immagini.  Aver verificato che le parole vengano scritte nell’immagine | | |
| **Procedura** | 1. Inserire delle parole nell’input di testo 2. Controllare con l’immagine iniziale che i colori siano stati mantenuti | | |
| **Risultati attesi** | I colori delle parole rispecchiano i colori iniziali dell’immagine | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-008  REQ-006 | **Nome** | Test delle parole bloccate |
| **Descrizione** | Testare che le parole bloccate non vengano mostrate sull’immagine | | |
| **Prerequisiti** | Aver verificato che le parole vengano visualizzate nell’immagine | | |
| **Procedura** | 1. Inserire nell’input di testo delle parole bloccate 2. Controllare che non vengano mostrate sull’immagine | | |
| **Risultati attesi** | Dopo aver inserito le parole bloccate esse non verranno mostrate sull’immagine | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-009  REQ-007 | **Nome** | Test delle parole con importanza maggiore |
| **Descrizione** | Testare che le parole con importanza maggiore definite dall’utente vengano effettivamente mostrate più grandi delle altre | | |
| **Prerequisiti** | Aver verificato che le parole vengano visualizzate nell’immagine | | |
| **Procedura** | 1. Inserire nel campo di testo delle parole enfatizzate del testo 2. Controllare che nell’immagine la parola inserita all’interno del campo di testo venga visualizzata più in grande | | |
| **Risultati attesi** | La parola inserita verrà mostrata con un font di dimensioni maggiori rispetto alle altre parole | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-010  REQ-008 | **Nome** | Test del bordo |
| **Descrizione** | Testare che quando viene abilitato il bordo esso appaia in modo corretto | | |
| **Prerequisiti** | Aver verificato che le parole vengano visualizzate nell’immagine | | |
| **Procedura** | 1. Abilitare il bordo dell’immagine 2. Controllare che sull’immagine sia apparso un bordo nero 3. Scegliere un colore per il bordo 4. Controllare che il colore venga applicato in modo corretto 5. Impostare uno spessore al bordo 6. Controllare che lo spessore venga applicato correttamente all’immagine | | |
| **Risultati attesi** | Una volta abilitato il bordo all’immagine esso verrà automaticamente mostrato in modo corretto, quando si cambierà il colore, esso si aggiornerà automaticamente al colore selezionato e infine quando si imposterà uno spessore il bordo si aggiornerà automaticamente ingrandendosi o rimpicciolendosi | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-011  REQ-009 | **Nome** | Test dell’aggiornamento in tempo reale dell’immagine |
| **Descrizione** | Testare che l’immagine si aggiorni automaticamente ad ogni cambiamento effettuato dall’utente | | |
| **Prerequisiti** | Aver verificato che tutti gli input a disposizione dell’utente funzionino | | |
| **Procedura** | 1. Caricare una nuova immagine 2. Inserire del testo e controllare che ad ogni parola l’aggiornamento dell’immagine 3. Cambiare font e controllare che venga cambiato in tempo reale anche nell’immagine 4. Abilitare il bordo e controllare che si aggiorni automaticamente mostrandolo di colore nero 5. Cambiare il colore al bordo e controllare che si aggiorni automaticamente usando il colore selezionato | | |
| **Risultati attesi** | L’immagine, qualunque cosa faccia l’utente, si aggiorna sempre in tempo reale | | |
| **Test Case**  **Riferimento** | TC-012  REQ-010 | **Nome** | Test della risoluzione a scelta dell’immagine |
| **Descrizione** | Testare che l’utente possa scegliere una risoluzione a suo piacimento e che questa venga effettivamente applicata | | |
| **Prerequisiti** | Aver verificato che l’applicazione funziona correttamente | | |
| **Procedura** | 1. Caricare una nuova immagine 2. Inserire del testo per creare una nuova immagine 3. Premere il pulsante download 4. Scegliere vari tipi di risoluzione e scaricare l’immagine 5. Controllare che ogni immagine rispecchi la risoluzione data | | |
| **Risultati attesi** | L’immagine sarà sempre della risoluzione data dall’utente al momento del download | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-013  REQ-011 | **Nome** | Test del formato dell’immagine una volta scaricata |
| **Descrizione** | Testare che l’immagine, una volta scaricata, sia del formato scelto precedentemente dall’utente | | |
| **Prerequisiti** | Aver verificato che l’applicazione funzioni correttamente e in modo completo | | |
| **Procedura** | 1. Caricare una nuova immagine 2. Inserire del testo per creare una nuova immagine 3. Premere il tasto download 4. Scegliere una risoluzione base dell’immagine 5. Scegliere il formato 6. Controllare che una volta scaricata sia effettivamente di quel formato | | |
| **Risultati attesi** | L’immagine, una volta scaricata, sarà del formato scelto al momento del download da parte dell’utente | | |

## Risultati test

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Risultato ottenuto** | **Stato** |
| TC-001 |  |  |
| TC-002 |  |  |
| TC-003 |  |  |
| TC-004 |  |  |
| TC-005 |  |  |
| TC-006 |  |  |
| TC-007 |  |  |
| TC-008 |  |  |
| TC-009 |  |  |
| TC-010 |  |  |
| TC-011 |  |  |
| TC-012 |  |  |
| TC-013 |  |  |

## Mancanze/limitazioni conosciute

Descrizione con motivazione di eventuali elementi mancanti o non completamente implementati, al di fuori dei test case. Non devono essere riportati gli errori e i problemi riscontrati e poi risolti durante il progetto.

# Consuntivo

Consuntivo del tempo di lavoro effettivo e considerazioni riguardo le differenze rispetto alla pianificazione (cap. 1.7) (ad esempio Gantt consuntivo).

Stile orizzontale



CONSUNTIVO

Figura Esempio di diagramma di Gantt consuntivo

# Conclusioni

Siamo molto contenti del risultato della nostra soluzione, non pensiamo cambi il mondo, ma nemmeno che sia marginale. Anche se già presente nel web siamo convinti che averne una versione applicativa sia molto comoda, infatti questa è anche molto facilmente modificarla per possibili sviluppi futuri

## Sviluppi futuri

Abbiamo ipotizzato molti possibili sviluppi futuri, tra questi abbiamo l’idea di aumentare la gamma delle immagini sulla quale è possibile applicare l’effetto worldcloud, che può sembrare una cosa banale, ma richiede diverso tempo per trovare o sviluppare un ricercatore di bordi più avanzato.

Durante l’implementazione avevamo anche pensato all’inserimento delle parole in obliquo, anche di permettere all’utente di decidere il colore delle parole oppure di scrivere le parole all’interno di altre.

## Considerazioni personali

*Alessandro:*

Ho messo dedizione, impegno e disciplina in questo lavoro, ho lasciato la mia anima.

Durante la realizzazione di questo progetto mi sono reso utile sia creando che supportando i miei compagni e ho conquistato la loro fiducia. Ho imparato a lavorare in python, ho ampliato le mie conoscenze di opencv e soprattutto non sono impazzito ad utilizzare kivy.

Credo di aver raggiunto tutti i traguardi che bramavo, oramai mi sento completo, grazie a questo progetto mi sono formato moltissimo e per questa volta sono contento di averne preso parte.

*Christian:*

*Edoardo:*

La risoluzione di questo progetto mi ha insegnato diverse cose, tra le quali sicuramente è l’utilizzo del linguaggio python. L’ho trovato interessante non avendolo mai utilizzato prima, anche se ha portato alcune considerazioni negative dal mio punto di vista.

Ma tutto sommato sono contento della mia esperienza con esso.

Il progetto in sè essendo il primo di gruppo mi ha tranquillizzato molto, infatti vedendo i requisiti con la certezza di dover fare tutto da solo è spaventosa, soprattutto considerano la sua grandezza.

# Bibliografia

## Sitografia

1. URL del sito (se troppo lungo solo dominio, evt completo nel diario),
2. Eventuale titolo della pagina (in italico),
3. Data di consultazione (GG-MM-AAAA).

**Esempio:**

* http://standards.ieee.org/guides/style/section7.html, *IEEE Standards Style Manual*, 07-06-2008.

# Glossario

|  |  |
| --- | --- |
| **Termine** | **Significato** |
| ASCII | American Standard Code for Information Interchange, codice per la codifica di caratteri. |
| World cloud | Stile di immagine dove un soggetto piene riempito di parole in base alla loro presenza in un testo |
|  |  |
|  |  |
|  |  |
|  |  |

# Indice delle figure

[Figura 1 UseCase 10](#_Toc132978875)

[Figura 4 Esempio di diagramma di Gantt 12](#_Toc132978876)

[Figura 5 Esempio di diagramma di Gantt 13](#_Toc132978877)

[Figura 2 Diagramma di flusso 16](#_Toc132978878)

[Figura 3 Diagramma opzioni varie 17](#_Toc132978879)

[Figura 8 Esempio di diagramma di Gantt consuntivo 29](#_Toc132978880)

# Allegati

* Diari di lavoro
* Codici sorgente/documentazione macchine virtuali
* Documentazione di prodotti di terzi 🡪 algoritmo wc
* Manuale di utilizzo
* QdC
* Prodotto