ImageDots

1 Introduzione 3

1.1 Informazioni sul progetto 3

1.2 Abstract 3

1.3 Scopo 3

2 Analisi 4

2.1 Analisi del dominio 4

2.2 Analisi e specifica dei requisiti 4

2.2.1 Spiegazione elementi tabella dei requisiti: 5

2.3 Use case 6

2.4 Pianificazione 6

2.5 Analisi dei mezzi 6

2.5.1 Software 6

2.5.2 Hardware 6

3 Progettazione 7

3.1 Design dell’architettura del sistema 7

3.1.1 Struttura dell’applicativo WEB 7

3.2 Design dei dati e database 8

3.3 Design delle interfacce 8

3.4 Design procedurale 9

4 Implementazione 10

4.1 Struttura base 10

4.2 main.js 10

4.3 layers.js 11

4.4 dots.js 13

4.5 dots\_manager.js 17

4.6 drawing.js 20

4.7 drawing\_manager.js 26

4.8 Funzionamento 28

4.8.1 Inizio 28

4.8.2 Puntini 28

4.8.3 Layer 31

4.8.4 Disegno 32

4.8.5 Salvataggio, stampa 33

5 Test 34

5.1 Protocollo di test 34

5.2 Risultati test 39

5.3 Mancanze/limitazioni conosciute 40

6 Consuntivo 40

7 Conclusioni 40

7.1 Sviluppi futuri 40

7.2 Considerazioni personali 40

8 Glossario 40

9 Bibliografia 41

9.1 Bibliografia per articoli di riviste: 41

9.2 Bibliografia per libri 41

9.3 Sitografia 41

10 Allegati 41

# Introduzione

## Informazioni sul progetto

## Abstract

*In an age when automation permeates almost every daily action, from turning on lights to driving cars, creating custom images with dots to connect has remained surprisingly manual. This project aims to change that by offering a creative alternative to searching for pre-existing images that may not match the user's original idea.*

*Users can add, delete, and move points, displaying the result of connected points in real time.*

*Finally, the ability to print the image created takes creativity off the screen and into the real world. This feature transforms the platform from a simple digital tool to a versatile tool for education and creativity that can be used in a variety of settings, from the classroom to the home.*

## Scopo

Lo scopo del progetto è offrire la possibilità di creare immagini personalizzate con punti da collegare, fornendo un’alternativa creativa alla ricerca di immagini preesistenti che potrebbero non corrispondere all’idea originale dell’utente.

Gli utenti possono aggiungere, eliminare e spostare i punti sul layer dei punti e vedere il risultato dei punti collegati. Inoltre, offre una serie di strumenti di disegno, tra cui penna, gomma, linee, rettangolo, cerchio e ovale, che possono essere utilizzati per creare disegni unici. Infine, offre la possibilità di stampare l’immagine creata, rendendola uno strumento versatile per l’educazione e la creatività.

# Analisi

## Analisi del dominio

L’applicativo verrà utilizzato principalmente da genitori, da maestri dell’asilo o delle scuole elementari.

Si vuole dare la possibilità di generare le proprie immagini con i puntini da collegare, per evitare la ricerca di immagini già con i puntini che potrebbero non rispettare l’idea originale dell’utente.

## Analisi e specifica dei requisiti

|  |  |
| --- | --- |
| **ID: REQ-001** | |
| **Nome** | Interfaccia |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Caricamento immagine |
| **Sotto requisiti** | |
| **001** | I formati dell’immagine caricata sono webp, jpg, jpeg, png |
| **002** | L’Immagine caricata deve essere ridimensionata |
| **003** | L’immagine creata deve essere salvabile |
| **004** | L’immagine creata deve essere stampabile |
| **005** | Si deve poter scartare l’immagine |

|  |  |
| --- | --- |
| **ID: REQ-002** | |
| **Nome** | Layer |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Ci devono essere più layer: disegni, puntini |
| **Sotto requisiti** | |
| **001** | Si può cambiare la visualizzazione dei layer |

|  |  |
| --- | --- |
| **ID: REQ-003** | |
| **Nome** | Puntini |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Bisogna essere in grado di aggiungere, eliminare, spostare i puntini |
| **Sotto requisiti** | |
| **001** | Il numero dei puntini aggiunti aumenta ogni volta che vengono aggiunti |
| **002** | Il numero dei puntini deve poter essere modifcato |
| **003** | Si può vedere il risultato dei puntini collegati |

|  |  |
| --- | --- |
| **ID: REQ-004** | |
| **Nome** | Disegno |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Ci devono essere i strumenti di disegno come penna, gomma, linee, rettangolo, cerchio, ovale |
| **Sotto requisiti** | |
| **001** | Unione del disegno insieme ai puntini |

|  |  |
| --- | --- |
| **ID: REQ-005** | |
| **Nome** | Stampa |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | L’immagine creata deve essere stampabile |
| **Sotto requisiti** | |
| **001** | Ricerca stampanti |

### 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

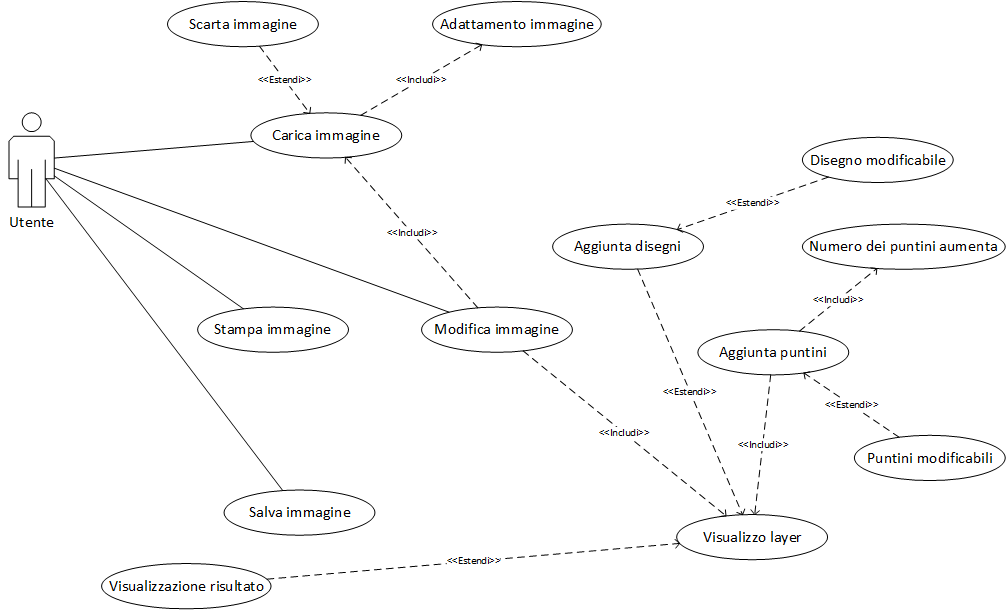


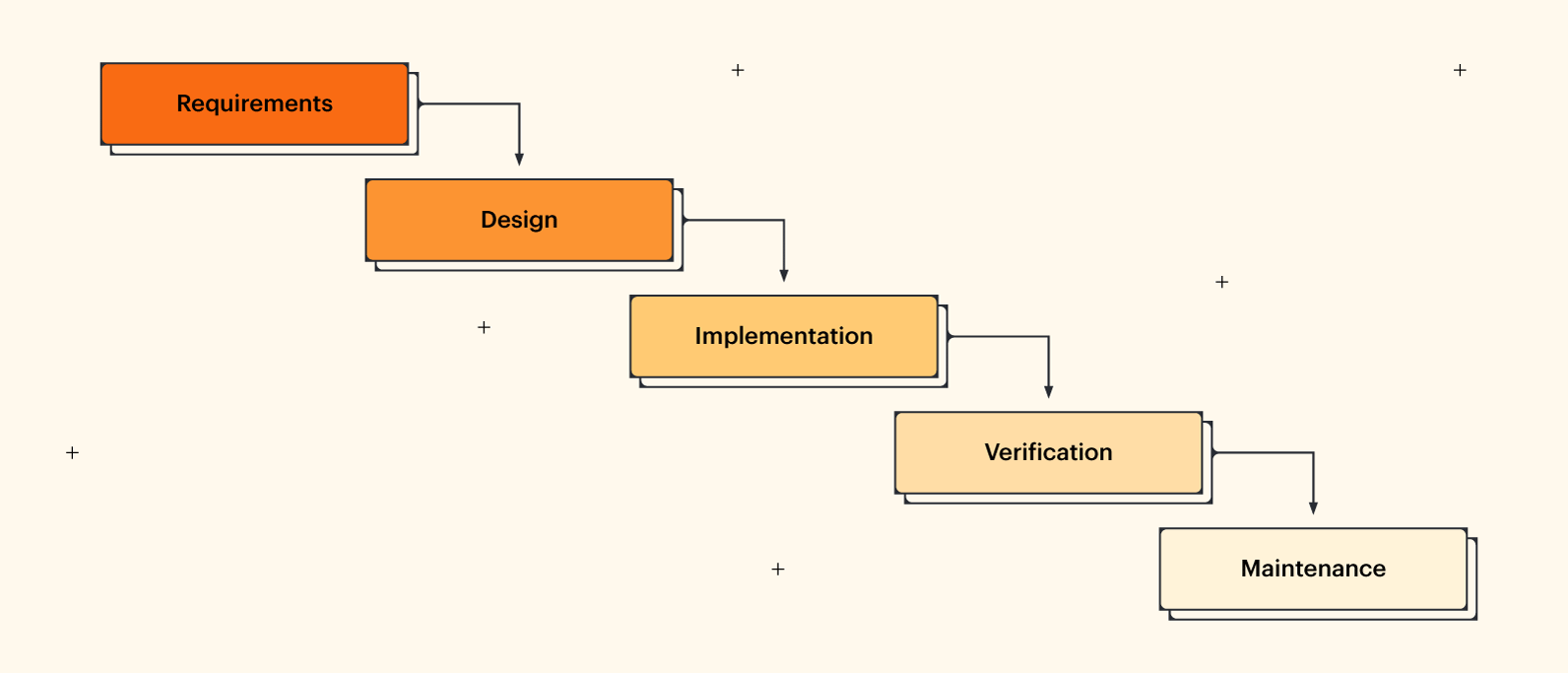
Figura 0 – Use case

## Pianificazione

Figura 1- Gantt

## Analisi dei mezzi

Ho optato per il modello a cascata poiché, a mio avviso, facilita l’elaborazione del progetto suddividendo le attività in maniera distinta.



### Software

* Visual Studio Code 1.78.2
* Html 5
* Css 3
* Javascript 1.5
* Print.js 1.5.0

### Hardware

* 1 PC: Le componenti del PC che verrà utilizzato per lo sviluppo dell’applicativo sono:
  + I7-9700
  + 32 GB RAM
  + SSD 512 GB
  + NVIDIA GeForce RTX 2060

# Progettazione

## Design dell’architettura del sistema

### Struttura dell’applicativo WEB

La struttura delle cartelle è la seguente e all’interno sono presenti i seguenti elementi:

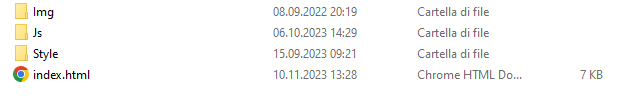


Figura 2 – Struttura cartelle

* **Img:** Dentro questa cartella sono presenti le immagini utilizzate dall’applicativo.
* **Js:** Contiene tutti gli script Javascript.
* **Style:** Contiene tutti i file CSS, che modificano la veste grafica dell’applicativo

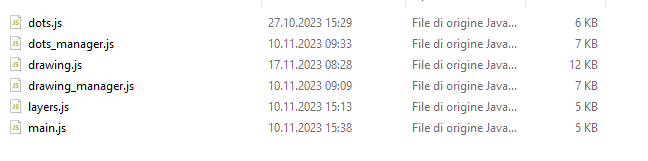
****

Figura 3 – File cartella Js

* **File dots.js e drawing.js:** contengono le funzioni essenziali per il disegno dei punti.
* **File dots\_manager.js e drawing\_manager.js**: fungono da intermediari tra l'HTML e il JS, e includono anche funzioni semplici e di dimensioni ridotte.
* **File layers.js:** gestisce il cambio della visualizzazione dei layer.
* **File main.js:** comprende il resto delle funzioni, come il salvataggio, stampa, inizializzazione di funzioni.

## Design dei dati e database

## Design delle interfacce

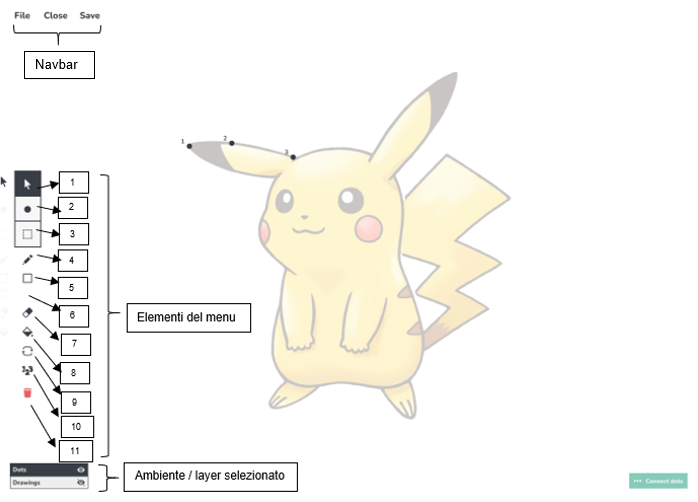


Figura 4 – Pagina principale

L’applicativo lavora principalmente su questa pagina, le differenze principali dal design al codice sono che:

* L’elemento del menu 1 è disponibile in entrambi gli ambienti.
* Gli elementi del menu 2, 3, 9, 10 sono disponibili solo quando siamo nell’ambiente dei puntini.
* Gli elementi del menu 4, 5, 6, 7, 8 sono disponibili solo quando siamo nell’ambiente dei disegni.
* L’elemento del menu 10 e 11 appare quando un puntino viene selezionato.
* L’elemento del menu 11 appare anche quando rettangoli ed elissi vengono selezionati.
* Il pulsante “Connect dots” è disponibile solo nell’ambiente dei puntini.
* Alla fine degli elementi c’è uno slider per cambiare le dimensioni di puntini o delle linee delle forme disegnate.
* Al posto dell’elemento “File” nella navbar c’è l’elemento

I vari elementi del menu rappresentano le seguenti funzionalità:

1. La selezione di un puntino o disegno.
2. L’aggiunta di un puntino.
3. La selezione di tutti i puntini.
4. Disegno di linee a mano libera.
5. Disegno di rettangoli.
6. Disegno di elissi.
7. Cancellazione di linee a mano libera.
8. Se selezionato le forme disegnate saranno riempite.
9. Inverte il numero dei puntini.
10. Cambia il numero del puntino selezionato.
11. Elimina il puntino o la forma selezionata.

Gli elementi della navbar rappresentano le seguenti funzionalità:

* File: diventano poi “Open”, prende l’immagine dall’utente.
* Close: chiude l’immagine e cancella tutte le cose che l’utente ha fatto.
* Save: apre il menu di salvataggio.

Pulsante “Connect dots”:

* Compare se siamo nell’ambiente dei puntini.
* Se schiacciato fa vedere tutti i puntini collegati.

Menu layer:

* Si seleziona l’ambiente che si vuole utilizzare.
* Si può visualizzare l’altro ambiente che non è selezionato. Es. l’utente è nell’ambiente puntini, vuole vedere come stanno i disegni insieme, per farlo schiaccia l’occhio accanto a “Drawings”.



Figura 5 – Menu di salvataggio

* **Name:** Rappresenta il nome del file scaricato, che verrà scaricato in formato png.
* **Include drawings:** Se schiacciato oltre ai puntini, saranno inclusi anche i disegni nell’immagine finale.
* **Connect dots (solution):** Stesso funzionamento del bottone “Connect dots”, per l’immagine scaricata.
* **Save:** Scarica l’immagine.
* **Print:** Apre il menu di stampa del browser.
* **Cancel:** Esce dal menu.

## Design procedurale

# Implementazione

## Struttura base

Ho diviso il codice nelle seguenti categorie:

* **File dots.js e drawing.js:** contengono le funzioni essenziali per il disegno dei punti.
* **File dots\_manager.js e drawing\_manager.js**: fungono da intermediari tra l'HTML e il JS, e includono anche funzioni semplici e di dimensioni ridotte.
* **File layers.js:** gestisce il cambio della visualizzazione dei layer.
* **File main.js:** comprende il resto delle funzioni, come il salvataggio, stampa, inizializzazione di funzioni.

## main.js

Come detto in precedenza, main.js comprende il resto delle funzioni, come il salvataggio, stampa, inizializzazione di funzioni.



Figura 6 – main.js, getImageData()

getImageData() serve a usare i dati dell'immagine presi dall'input, dopodiché modifica le proprietà dei vari canvas e l'immagine viene disegnata.

Questa parte inizialmente mi ha dato un po’ di problemi, soprattutto per il resize dell’immagine, ma “backgroundSize = ‘contain’” fa tutto il lavoro.



Figura 7 – main.js, printCanvas

Il metodo di stampa utilizza PrintJS che fa aprire il menu di stampa del browser con l’elemento HTML che vogliamo, si passa il workspace-canvas che racchiude gli elementi che l’utente vuole stampare, ogni volta dobbiamo deve essere pulito, altrimenti l’utente visualizza forme non interfacciabili.



Figura 8 – main.js, saveCanvas

Il metodo saveCanvas, serve a salvare in formato png l’immagine con gli elementi che l’utente vuole includere. Anche questo metodo utilizza il workspace-canvas.

## layers.js

Come detto in precedenza gestisce il cambio della visualizzazione dei layer.

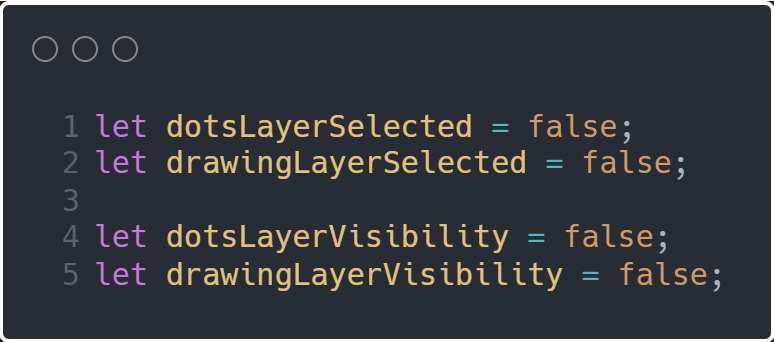


Figura 9 – layers.js, variabili globali

* **dotsLayerSelected e drawingLayerSelected:** servono dopo nella selezione della visibilità dei layer, perché se selezioni il layer dei puntini non puoi selezionare anche la visibiltà del layer dei puntini.
* **dotsLayerVisibility e drawingLayerVisibility:** se true usati in drawing.js e dots.js.



Figura 10 – layers.js, selezione layer

Quando viene selezionato un layer, in entrambi i casi si cambiano le variabili di prima, si fa il refresh del canvas e si cambiano gli elementi del menu per rispettare l’ambiente in cui siamo.

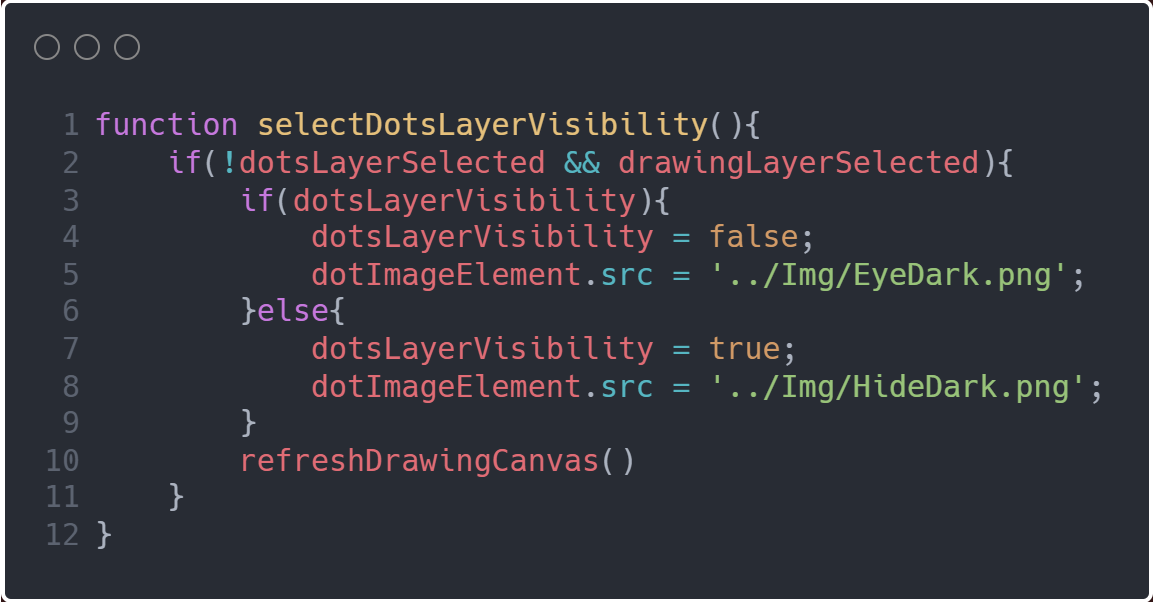


Figura 11 – layers.js, selezione visibilità

Quando viene selezionata la visibilità, in entrambi i casi le cose che succedono sono cambiare lo stile e le variabili.

## dots.js

Come detto in precedenza questo file serve contiene le funzioni di base dei puntini.

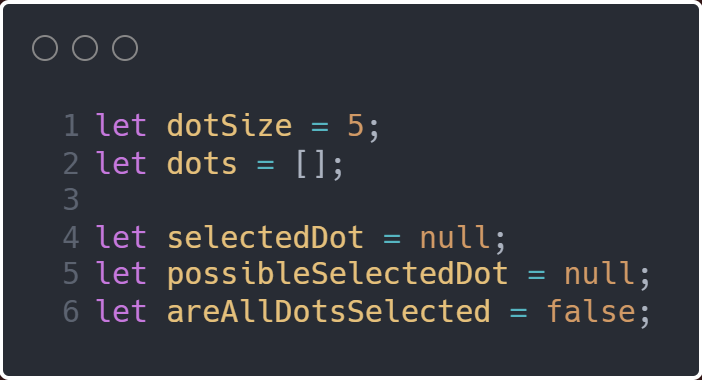


Figura 12 – dots.js variabili

Le funzioni dei puntini utilizzano principalmente queste variabili:

* **dotSize:** rappresenta il raggio del puntino.
* **dots:** contiene i puntini aggiunti dall’utente presenti nel canvas.
* **selectedDot:** il puntino selezionato dall’utente.
* **possibleSelectedDot:** il puntino che se l’utente schiaccia viene selezionato.



Figura 13 – dots.js, classe Dot

La classe Dot è la più usata ovviamente per il funzionamento dei puntini. Contiene solo un metodo oltre al costruttore che è draw(context). Le parti più interessanti sono:

* **Riga 23:** qua viene disegnato concretamente il puntino:
  + **pointX:** la coordinata X di dove viene disegnato il puntino.
  + **pointY:** la coordinata Y di dove viene disegnato il puntino.
  + **this.size:** rappresenta il raggio del puntino.
  + **Gli ultimi due parametri:** rappresentano l'angolo di inizio e l'angolo di fine dell'arco. In questo caso, l'arco si estende da 0 a 360 gradi, ovvero un giro completo.
* **Riga 30:** viene calcolato quanto deve essere la grandezza del testo del label, il numero del puntino. Ho deciso che la grandezza che più sta bene è 16px / 5 px del raggio del puntino.

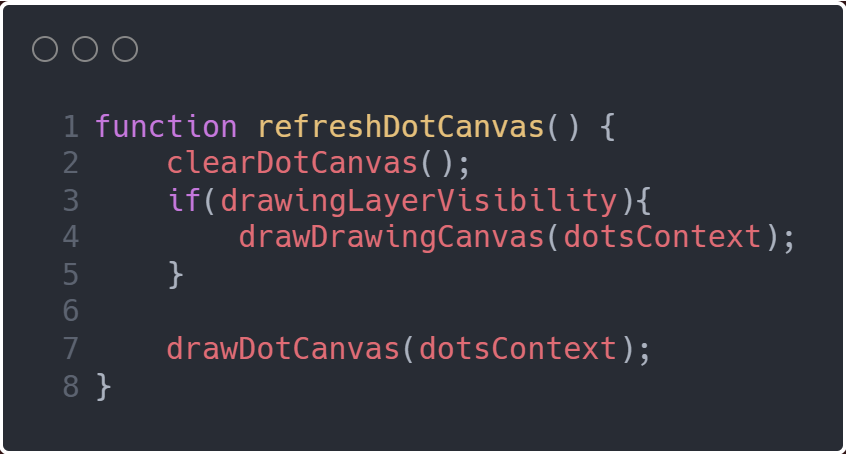


Figura 14 – dots.js, refreshDotCanvas()

Un metodo molto usato che vale la pena far vedere è refreshDotCanvas(), è usato continuamente nel codice. Prima viene pulito il canvas, poi se la visibilità del layer dei disegni, descritta precedentemente, è attiva aggiunge i disegni, infine usa drawDotCanvas(dotsContext) che disegnerà nel context dei puntini i puntini.

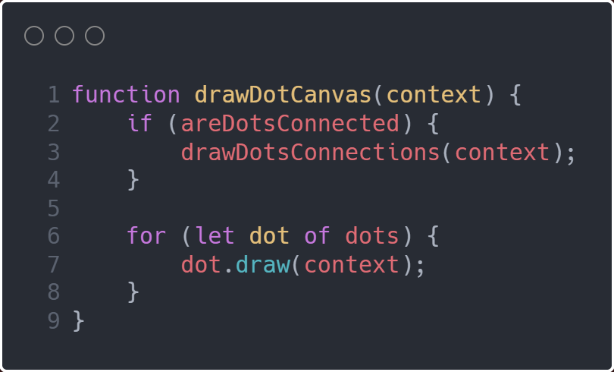


Figura 15 – dots.js, drawDotCanvas(context)

Come metodo drawDotCanvas(context) è molto semplice, la maggior parte del lavoro la fa la nostra classe Dot, se areDotsConnected è true, saranno disegnate le linee tra i puntini.

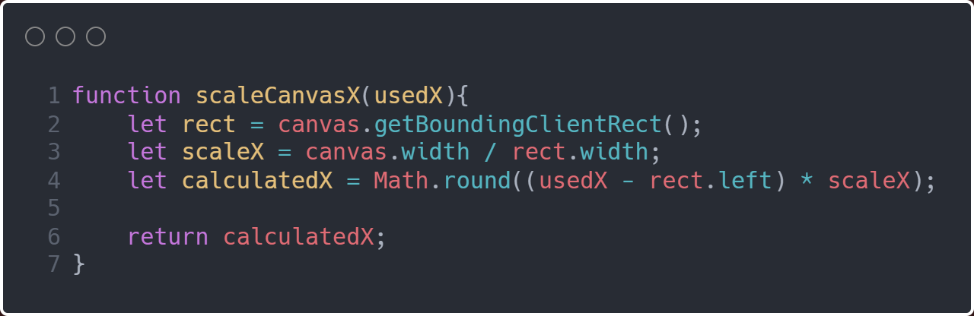


Figura 16 – dots.js, scaleCanvasX(usedX)

Il metodo scaleCanvasX(usedX) e scaleCanvasY(usedY) vengono usati spesso, senza di essi, le funzioni di aggiunta, spostamento e selezione dei puntini non sarebbero precise e potrebbero non funzionare.

* **usedX:** la X data dagli eventi, perché gli eventi prendono la X e la Y dello schermo e non del canvas.
* **getBoundingClientRect:** per ottenere il rettangolo che rappresenta l'elemento canvas nell'area di visualizzazione del browser. Questo rettangolo fornisce informazioni come l'altezza, la larghezza, la posizione sinistra e la posizione superiore dell'elemento canvas rispetto all'area di visualizzazione.
* **scaleX:** che rappresenta la scala orizzontale del canvas. Viene calcolata dividendo la larghezza del canvas per la larghezza del rettangolo ottenuto in precedenza.
* **calculatedX:** utilizzando la formula (usedX - rect.left) \* scaleX. Questo calcola la posizione orizzontale corrispondente all'input usedX rispetto all'area di visualizzazione del browser.

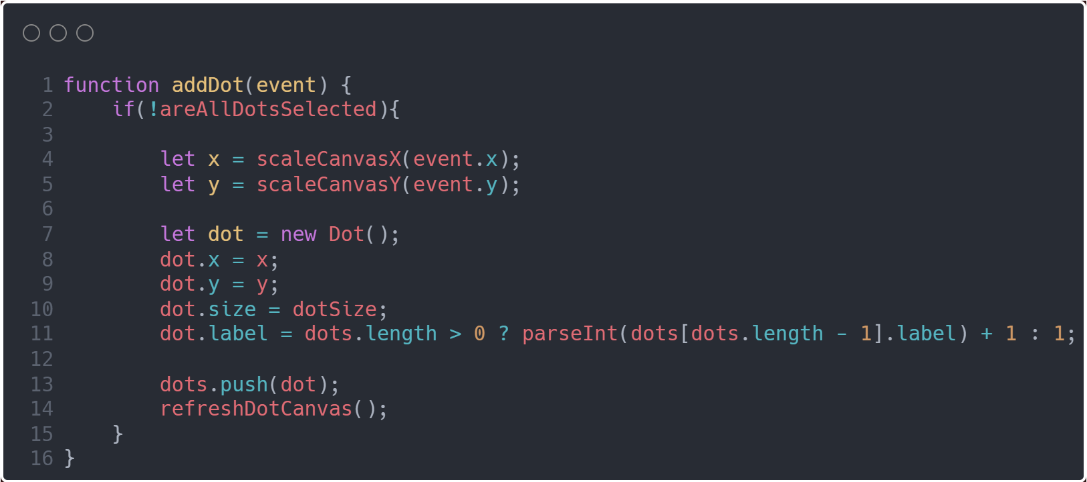


Figura 17 – dots.js, addDot()

addDot() viene usato come il nome fa intuire, per aggiungere i puntini. Il funzionamento è molto semplice, quando si schiaccia sul canvas, viene creato un nuovo puntino ed aggiunto all’array dei puntini, per il numero del puntino si prende il numero dell’ultimo puntino creato, si aggiunge 1.



Figura 18 – dots.js, selectDot

selectDot() viene usato per la selezione di un puntino, che potrà dopo essere cancellato, spostato, modificato, ecc.

Quando viene schiacciato il canvas, si controllano tutti puntini, per vedere se la posizione schiacciata corrisponde alla posizione di un puntino, se corrisponde si cambia il colore del puntino, viene salvato come puntino selezionato. Infine vengono visualizzati gli elementi del menu che servono al puntino selezionato.

Se la posizione schiacciata non corrisponde alla posizione di un puntino, vengono deselezionati tutti i puntini e spariscono gli elementi del menu che servirebbero al singolo puntino.



Figura 19 – dots.js, moveDot

Per muovere il puntino ci sono tre metodi, stopMovingDot, startMovingDot, moveDot.

* **stopMovingDot:** imposta la variabile isMouseDown a false, facendo smettere lo spostamento di un puntino.
* **moveDot:** sposta il puntino, per lo spostamento di un puntino ovviamente almeno un puntino deve essere selezionato, cambia la posizione del/dei puntino/i con quella del mouse.

## dots\_manager.js

Come detto in precedenza questo file funge da intermediario tra l'HTML e dots.js, include anche funzioni semplici e di dimensioni ridotte.



Figura 20 – dots\_manager.js, removeAllDotsEvents e resetOptionElements

removeAllDotsEvents e resetOptionElements servono ad evitare la ripetizione di codice, quando viene selezionata una modalità.



Figura 21 – dots\_manager.js, selectAddMode

Quando c’è un cambio di modalità il codice si struttura più o meno nella stessa maniera, prendendo d’esempio selectAddMode(), che serve ad aggiungere nuovi puntini:

* Si tolgono gli eventi delle altre modalità.
* Si modifica lo stile degli elementi del menu.
* Si usano funzioni di dots.js.



Figura 22 – dots\_manager.js, selectDeleteMode

selectDeleteMode() è la funzione utilizzata dal cestino, se tutti i puntini sono selezionati, rimuove tutti i puntini, altrimenti:

* Scorre l’array e rimuove il punto selezionato.
* Controlla se c’è una lacuna nel numero dei punti.
* Per ogni puntino controlla se è stata trovata una lacuna, decrementa il numero del punto. Altrimenti, controlla se c'è una lacuna tra i numeri dei punti.

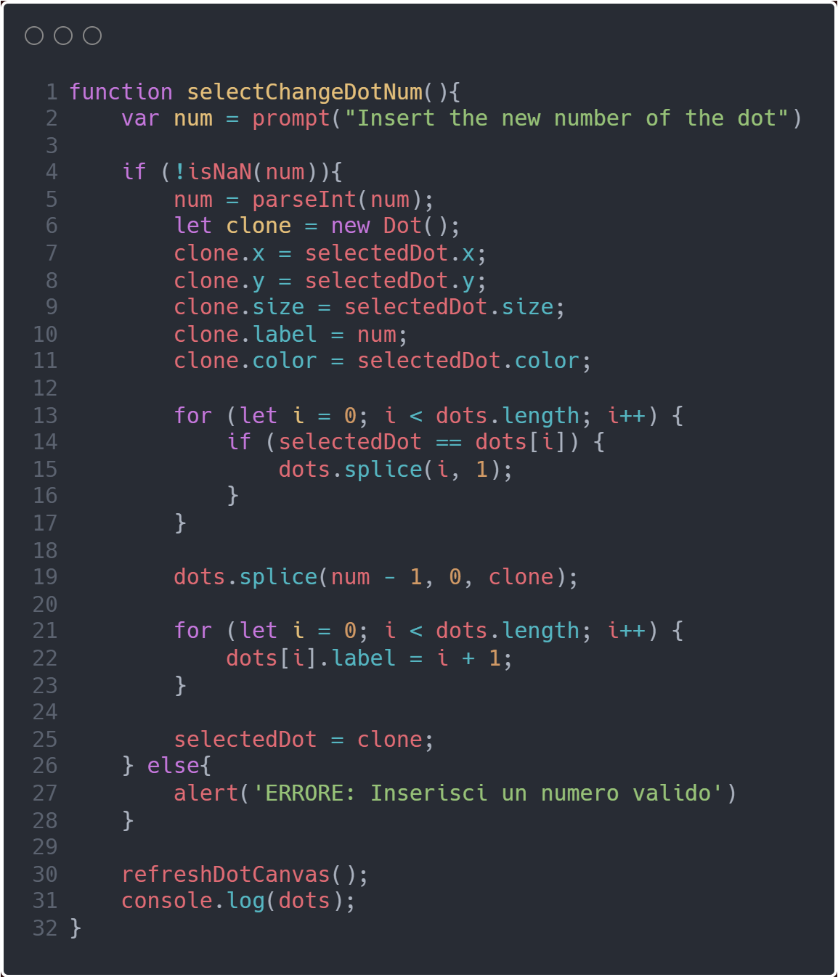


Figura 23 – dots\_manager.js, selectChangeDotNum

selectChangeDotNum chiede all’utente di inserire il nuovo numero del punto, verifica se il numero inserito è valido, crea un nuovo punto con le stesse proprietà del punto selezionato ma con il nuovo numero, rimuove il punto selezionato dall’array, inserisce il nuovo punto nell’array nella posizione specificata dal nuovo numero, aggiorna i numeri di tutti i punti nell’array, e infine aggiorna il canvas e stampa l’array dots sulla console.

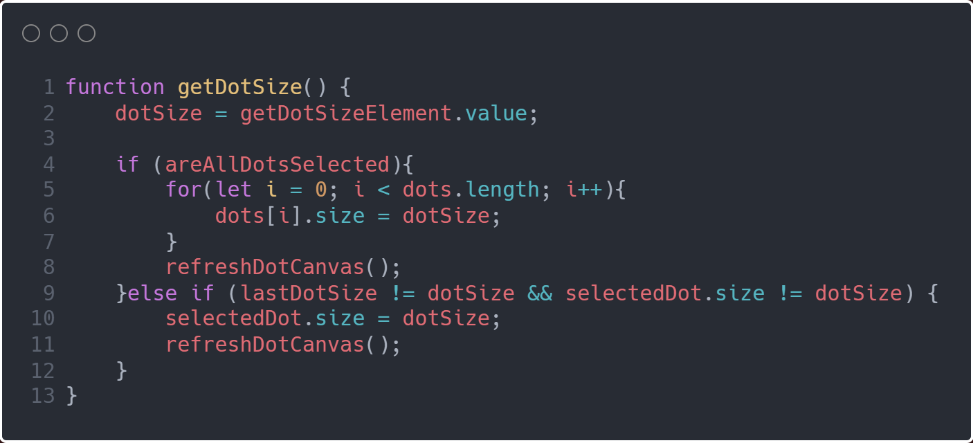


Figura 24 – dots\_manager.js, getDotSize

getDotSize() è il metodo utilizzato quando si usa lo slider, per cambiare la grandezza di un puntino. Se uno più puntini sono selezionati cambia la loro grandezza, se non ci sono puntini selezionati cambia la grandezza dei prossimi puntini che verranno aggiunti.

## drawing.js

Come detto in precedenza drawing.js contiene la logica del funzionamento dei disegni, contiene classi ed eventi che vengono usati in drawing\_manager.js e gli altri file.



Figura 25 – drawing.js, variabili

Le variabili più utilizzate sono le seguenti:

* **drawingSize:** spessore linee rettangoli, elissi e linee.
* **drawingColor:** colore usato nelle forme disegnate.
* **areShapesFilled:** se i rettangoli ed ellissi sono riempite con il colore selezionato.
* **lines:** tutte le linee disegnate nel drawingContext, create con la classe Line.
* **rects:** tutti i rettangoli disegnati nel drawingContext, creati con la classe Rect.
* **ellipses:** tutte le elissi disegnate nel drawingContext, create con la classe Ellipse.
* **selectedDrawing:** la forma selezionata che verrà eliminata, spostata, modificata.



Figura 26 – drawing.js, Line

La classe Line è utilizzata sia nel disegno delle linee a mano libera che quelle tra i puntini. I parametri necessari al funzionamento sono:

* **fromX:** coordinata X nel canvas da dove parte la linea.
* **fromY:** coordinata Y nel canvas da dove parte la linea.
* **toX:** coordinata X nel canvas dove arriva la linea.
* **toY:** coordinata Y nel canvas dove arriva la linea.

È importante la riga 12, altrimenti le linee vengono disegnate come se fossero un codice a barre.



Figura 27 – drawing.js, classe Rect

La classe Rect serve a disegnare i rettangoli, funziona principalmente come le altre classi per disegnare, i parametri sono:

* **x:** coordinata X nel canvas dove inizia la forma.
* **y:** coordinata Y nel canvas dove inizia la forma
* **width:** larghezza della forma.
* **height:** altezza della forma.
* **filled:** per riempire il rettangolo quando viene disegnato.
* **color:** colore della linea e del riempimento del rettangolo disegnato.
* **size:** spessore della linea del rettangolo disegnato.

Principalmente tutte le forme disegnate hanno questo tipo di parametri.



Figura 28 – drawing.js, drawDrawingCanvas

In drawDrawingCanvas il funzionamento è uguale a drawDotCanvas, ma ovviamente qua disegniamo le forme e non i puntini.



Figura 29 – drawing.js, stopDrawingRect, startDrawingRect, drawRect

Tutti i metodi di disegno hanno sempre lo stesso funzionamento, tranne che per le righe a mano libera che sono ancora più semplici:

* Presenza di uno stopDrawing.
* Presenza di uno StartDrawing.
* Controllo se la forma disegnata è la prima forma disegnata, altrimenti c’è un effetto visivo un po’ strano.
* lastForma, es. lastRect, serve a mantenere solo l’ultima forma disegnata, perché vediamo più forme quando trasciniamo, ma deve rimanerne solo una alla fine.
* Impostamento di lastForma a null, altrimenti a volte la forma viene duplicata quando usciamo dal canvas, perché stopDrawing risponde anche all’evento ‘mouseout’.



Figura 30 – drawing.js, stopDrawingLine, startDrawingLine, drawLine

Come detto in precedenza i metodi per disegnare le linee a mano libera sono più semplici, si crea una nuova linea in base dalle coordinate precedenti a quelle attuali.



Figura 31 – drawing.js, stopErasingDrawing, startErasingDrawing, eraseDrawing

Per cancellare le linee create a mano libera, abbiamo 3 metodi come per il disegno, stopErasingDrawing, startErasingDrawing ed eraseDrawing. Per controllare se una linea è da cancellare, controlliamo se la differenza tra le coordinate di dove parte la linea o dove arriva, è minore di drawingSize, il +1 è messo perché produce risultati leggermente più soddisfacenti.



Figura 32 – drawing.js, selectDrawings

Il metodo selectDrawings viene utilizzata per selezionare un disegno (rettangolo o ellisse) su un canvas quando l’utente fa clic su di esso. Se un disegno viene selezionato, viene visualizzata l’opzione per eliminarlo. Se l’utente fa clic su un’area del canvas dove non ci sono disegni, qualsiasi disegno precedentemente selezionato viene deselezionato.



Figura 33 – drawing.js, stopMovingDrawing, startMovingDrawing, moveDrawing

Come per i metodi per spostare i puntini, anche qua il funzionamento è semplice, si prende la forma selezionata (rettangoli/ovali) e le si cambiano le coordinate per spostarla.

## drawing\_manager.js

Come detto in precedenza questo file funge da intermediario tra l'HTML e drawing.js, include anche funzioni semplici e di dimensioni ridotte.



Figura 34 – drawing\_manager.js, removeAllDrawingEvents

Il metodo removeAllDrawingEvents, è simile all’altro metodo in dots\_manager.js, vengono tolti tutti gli eventi dal drawingCanvas per evitare codice ripetuto in seguito.



Figura 35 – drawing\_manager.js, selectFill

Come gli altri metodi, che interagiscono con un elemento del menu, ci sono dei cambiamenti di stile, oltre a ciò si controlla se un disegno è selezionato, in tal modo viene cambiato davanti agli occhi dell’utente il colore della forma selezionata.



Figura 36 – drawing\_manager.js, getDrawingSize

Come nel metodo simile in dots\_manager.js prendiamo il valore dal ranger, si controlla anche se una forma disegnata è selezionata, in tal modo da cambiare lo spessore in tempo reale.



Figura 37 – drawing\_manager.js, selectDeleteDrawingMode

Per eliminare le forme disegnate scorriamo gli array, controllando se l’elemento corrisponde alla forma selezionata, in tal caso si rimuove e si refresha il canvas.

## Funzionamento

### Inizio



Figura 38 – Inizio

Quando accediamo all’applicativo si presenta così, prima di fare qualsiasi altra cosa, bisogna schiacciare open e passare una immagine da usare.

### Puntini

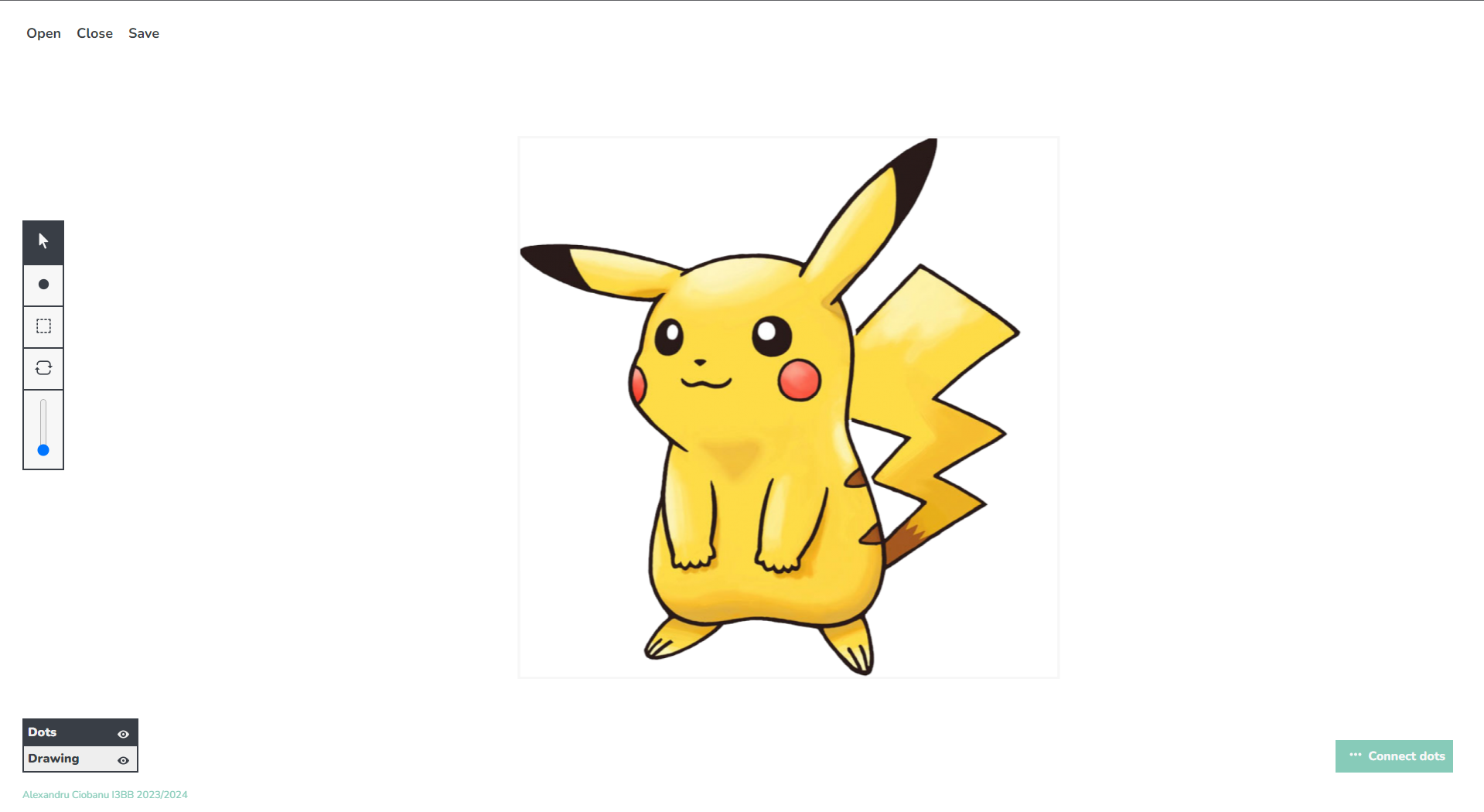


Figura 39 – Puntini

Ora c’è una immagine, per aggiungere puntini bisogna selezionare dal menu il puntino e schiacciare nel canvas.

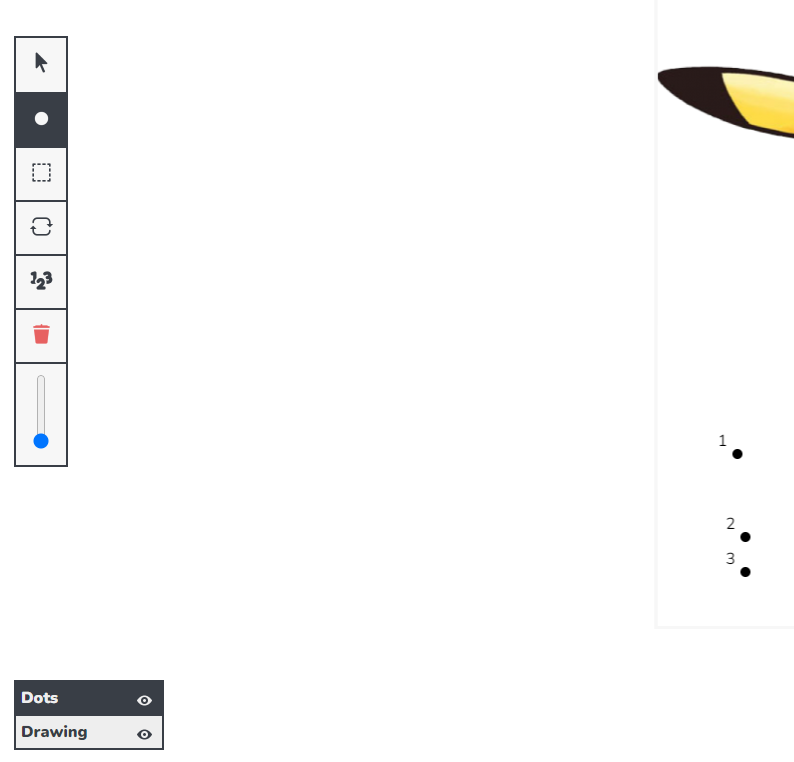


Figura 40 – Aggiunta puntini

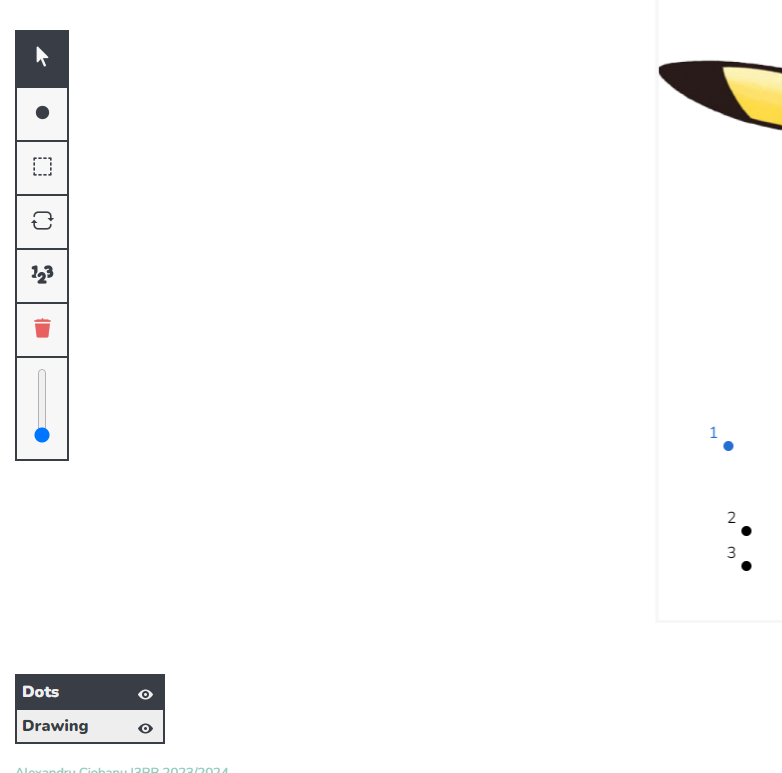


Figura 41 – Spostamento puntini

Se selezioniamo la freccia e si schiaccia un puntino, viene selezionato, da qua si può eliminarlo con l’elemento con l’icona del cestino, cambiare la grandezza con lo slider, cambiare il numero con l’elemento con l’icona dei numerini.



Figura 42 – Cambiamento numero puntino

Da qua scegliamo un numero che sia minore del numero dei puntini e schiacciamo OK.

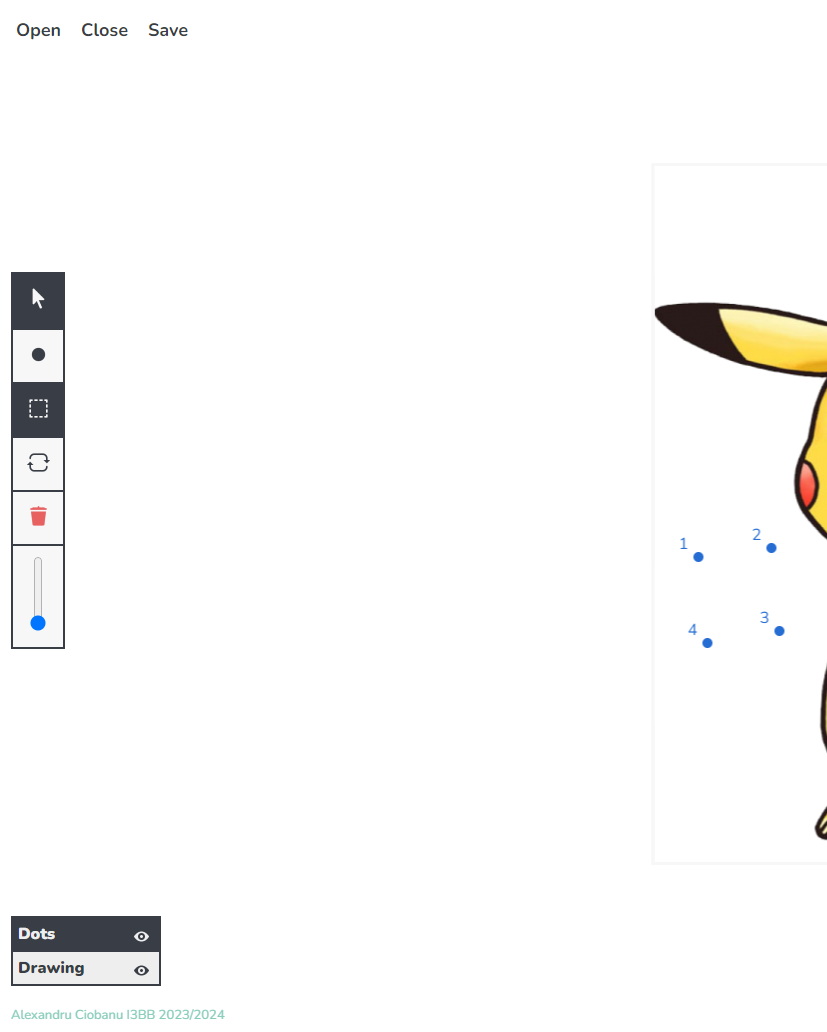


Figura 43 – Selezione di tutti i puntini

Per selezionare tutti i puntini, si schiaccia l’elemento con l’icona del quadrato tratteggiato, i puntini ora possono essere eliminati, spostati tutti insieme oppure possiamo modificare la loro grandezza.

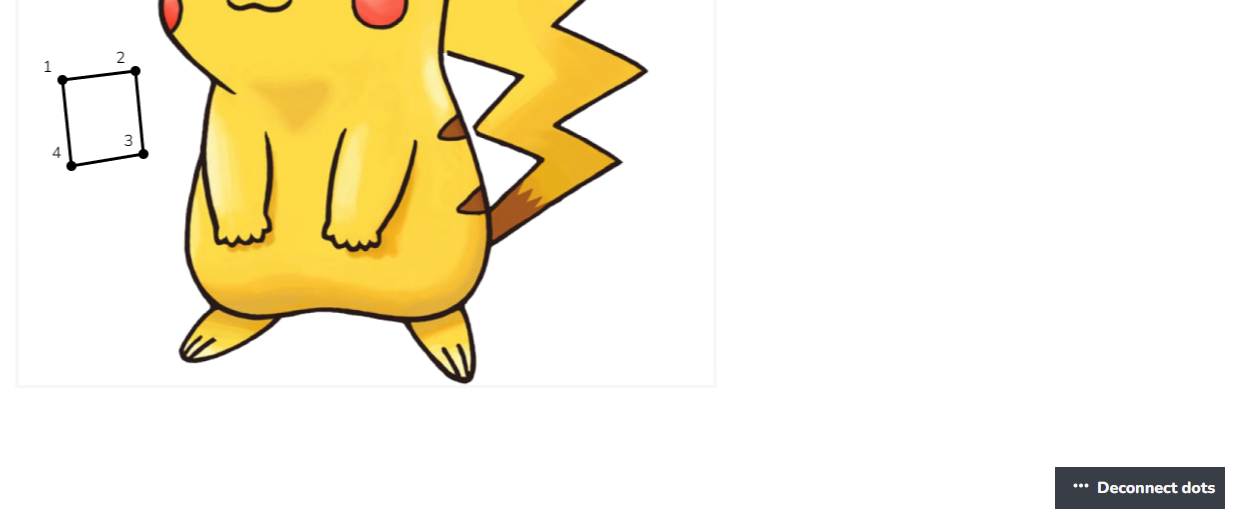


Figura 44 – Visualizzazione soluzione, puntini collegati

Per visualizzare tutti i puntini collegati ed avere una idea della soluzione, si schiaccia l’elemento “Connect dots”.

### Layer



Figura 45 – Layer

Per cambiare layer e avere la possibilità di creare anche disegni schiacciamo “Drawing”.

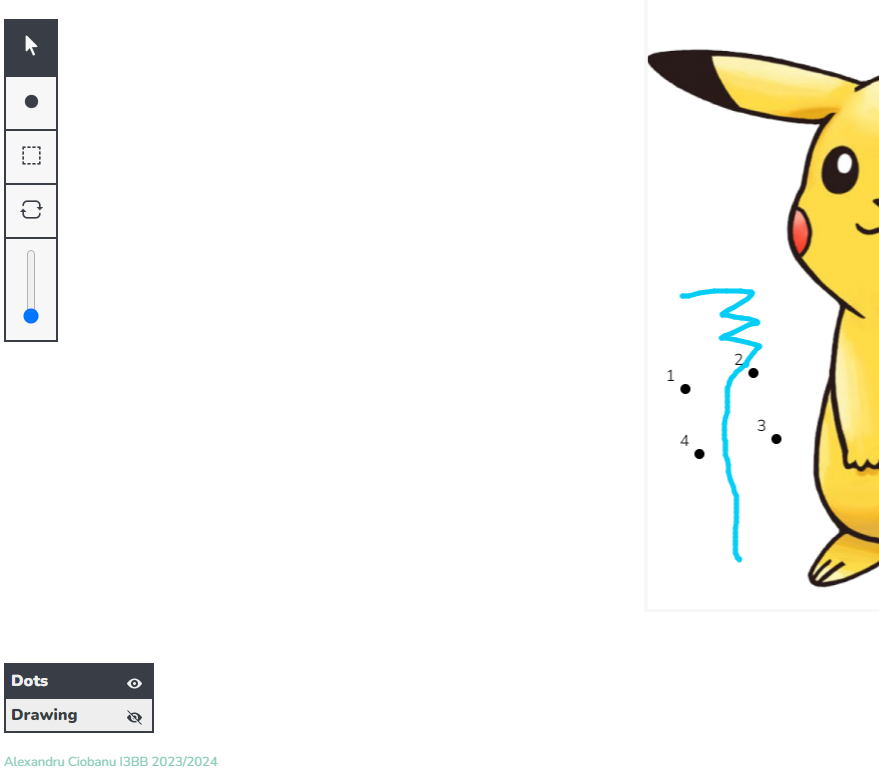


Figura 46 – Visualizzazione dei layer insieme

Se si vuole vedere come vengono combinati insieme i due layer nel lavoro finale, si schiaccia l’occhio del layer non selezionato.

### Disegno

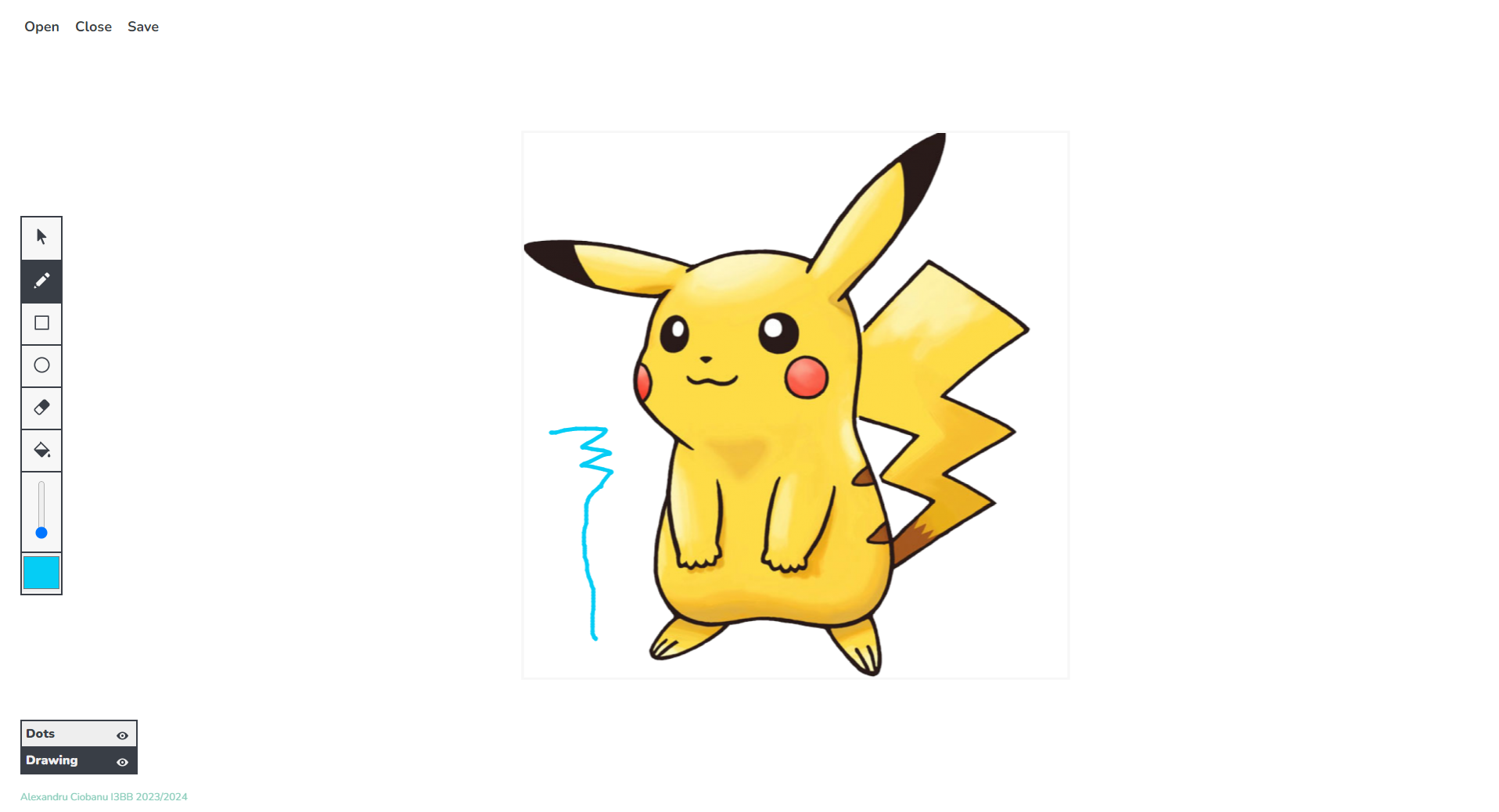


Figura 47 – Disegno a mano libera

Per disegnare a mano libera si seleziona l’elemento con l’icona della penna, volendo si può cambiare spessore con lo slider e colore.



Figura 48 – Gomma, cancellazione linee a mano libera

Selezionando la gomma e regolando la grandezza del raggio d’azione, dopodiché si possono cancellare le linee disegnate a mano libera andandoci sopra e schiacciando.

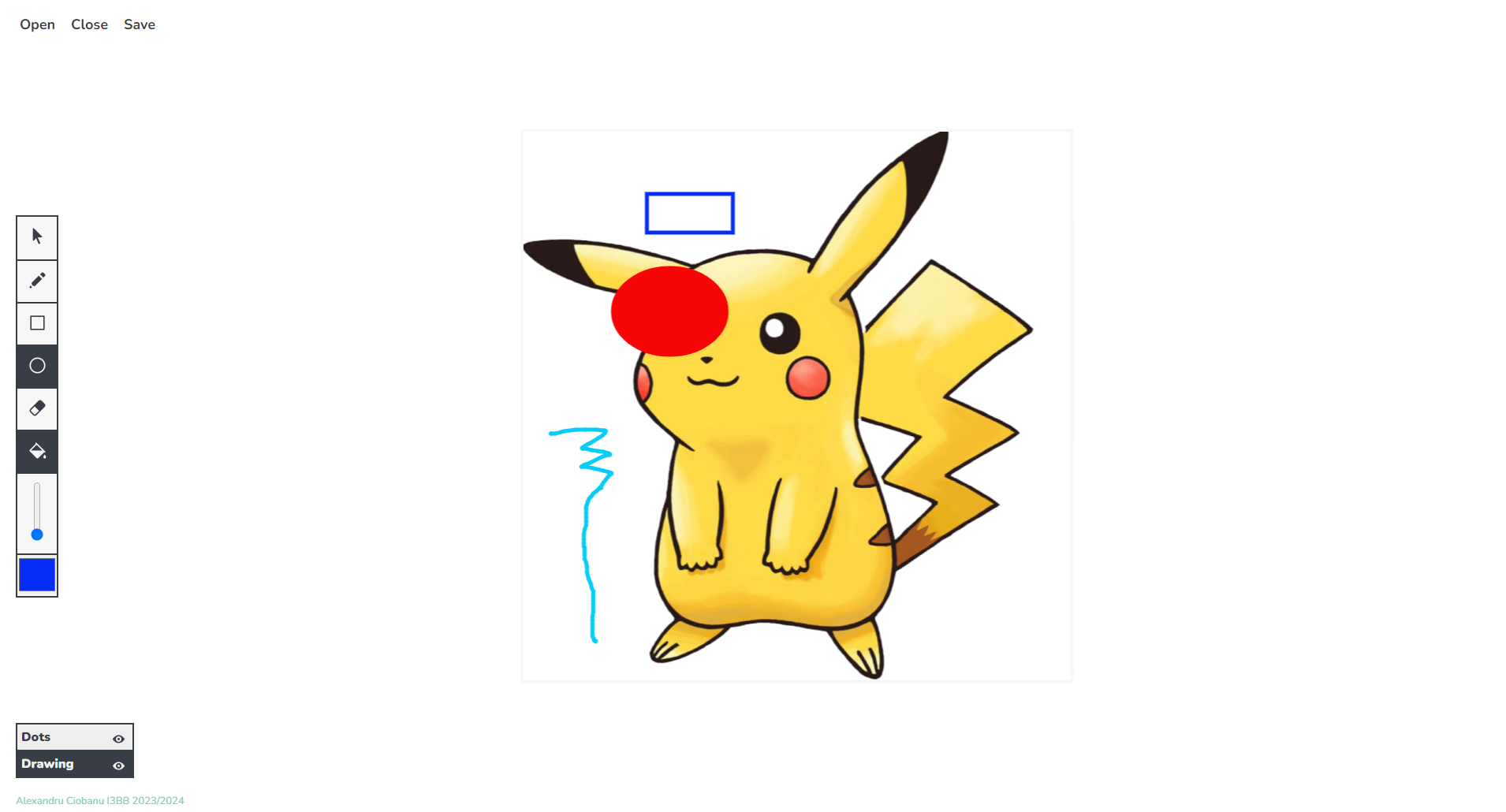


Figura 49 – Forme, rettangoli ed ellissi

Possiamo disegnare delle forme come elissi o rettangoli, selezionando l’elemento con l’icona che le rappresenta, dopodiché basterà schiacciare e trascinare per crearle. Prima di creare una forma si può cambiare colore, decidere di far riempire le forme e cambiare lo spessore delle linee, schiacciando l’elemento con l’icona del secchiello.

Infine le forme sono selezionabili e spostabili, quando selezionate si può cambiare il colore, riempimento e spessore della linea in tempo reale.

A differenza delle linee a mano libera, per eliminare queste forme bisogna schiacciarle e schiacciare l’elemento con l’icona del cestino, non si può usare la gomma.

### Salvataggio, stampa

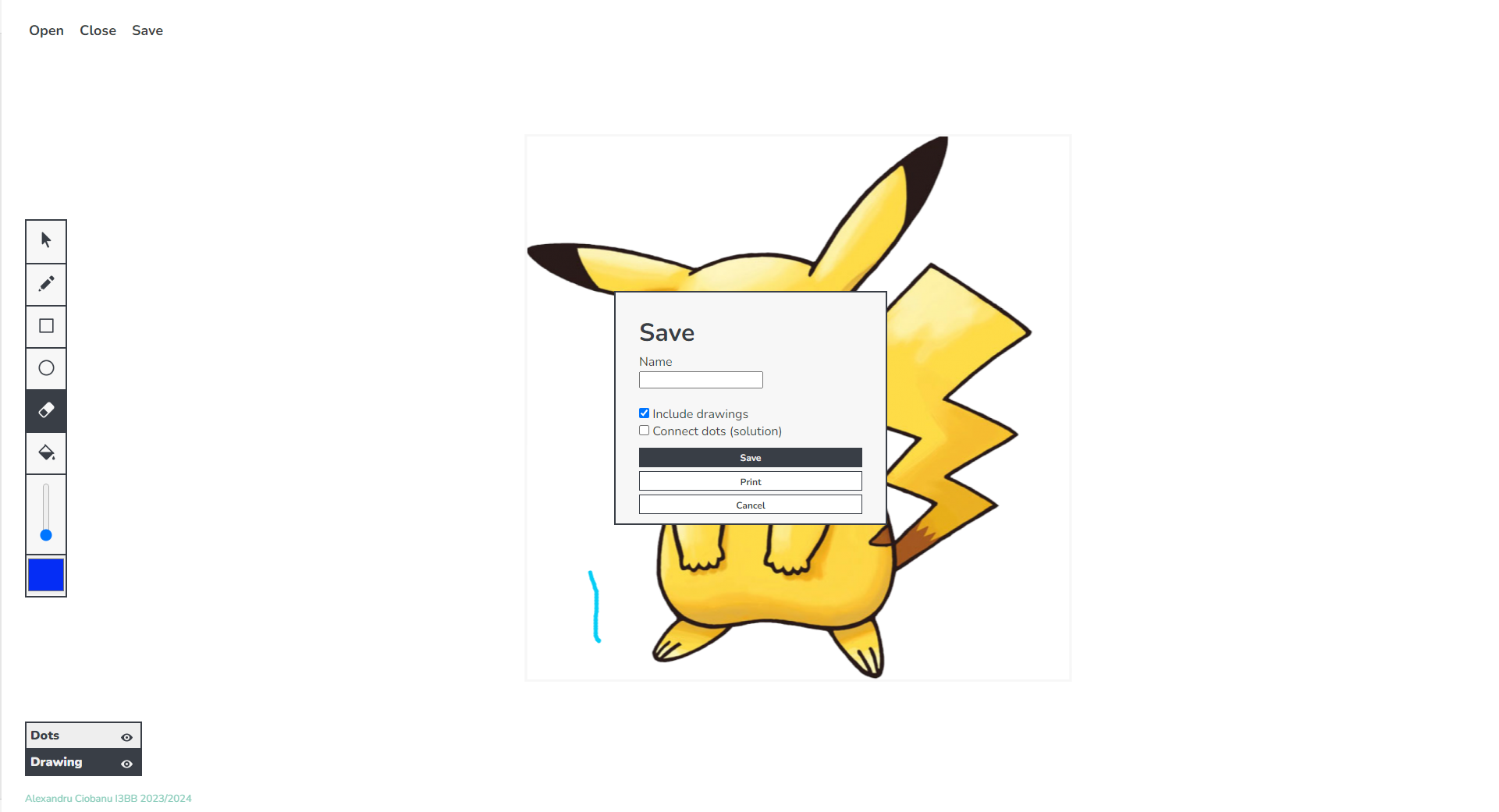


Figura 50 – Salvataggio

Per salvare l’immagine o stamparla, si schiaccia “Save”, qua si ha la possibilità di salvare o stampare l’immagine modificata. Se si seleziona “Include drawings”, i disegni verranno aggiunti al risultato finale, se “Connect dots” i puntini verranno collegati insieme, facendo vedere la soluzione.

# Test

## Protocollo di test

Definire in modo accurato tutti i test che devono essere realizzati per garantire l’adempimento delle richieste formulate nei requisiti. I test fungono da garanzia di qualità del prodotto. Ogni test deve essere ripetibile alle stesse condizioni.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-001  REQ-001 | **Nome:** | Caricamento immagine |
| **Descrizione:** | Caricamento di una immagine webp, jpg, jpeg, png | | |
| **Prerequisiti:** | Pagina iniziale completata e caricamento immagine. | | |
| **Procedura:** | 1. Cliccare “Open” 2. Selezionare una immagine | | |
| **Risultati attesi:** | Immagine caricata | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-002  REQ-001 | **Nome:** | Ridimensionamento immagine |
| **Descrizione:** | Se una immagine troppo grande viene caricata, sarà ridimensionata per non uscire al di fuori dello spazio dell’immagine. | | |
| **Prerequisiti:** | Pagina iniziale completata e caricamento immagine. | | |
| **Procedura:** | 1. Cliccare “Open” 2. Selezionare un’immagine di grandi dimensioni. | | |
| **Risultati attesi:** | L’immagine caricata non esce fuori dallo spazio riservato dell’immagine. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-003  REQ-001 | **Nome:** | Salvataggio immagine |
| **Descrizione:** | Una immagine dopo essere stata modificata deve poter essere salvata. | | |
| **Prerequisiti:** | Pagina iniziale completata, caricamento immagine e menu di salvataggio. | | |
| **Procedura:** | 1. Cliccare “Open” 2. Selezionare una immagine 3. Aggiungere qualche puntino a caso 4. Cliccare “Save” 5. Aggiungiamo un titolo all’immagine 6. Cliccare “Save” | | |
| **Risultati attesi:** | Una immagine png verrà scaricata, sarà trasparente e conterrà i puntini. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-004  REQ-001 REQ-005 | **Nome:** | Stampa immagine |
| **Descrizione:** | Una immagine dopo essere stata modificata deve poter essere stampata. | | |
| **Prerequisiti:** | Pagina iniziale completata, caricamento immagine e menu di salvataggio. | | |
| **Procedura:** | 1. Cliccare “Open” 2. Selezionare una immagine 3. Aggiungere qualche puntino a caso 4. Cliccare “Save” 5. Aggiungiamo un titolo all’immagine 6. Cliccare “Print” | | |
| **Risultati attesi:** | Il menu di stampa del browser verrà aperto. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-005  REQ-001 | **Nome:** | Scartare un’immagine |
| **Descrizione:** | Si deve poter scartare l’immagine dove si sta lavorando. | | |
| **Prerequisiti:** | Pagina iniziale completata, caricamento immagine. | | |
| **Procedura:** | 1. Cliccare “Open” 2. Selezionare una immagine 3. Cliccare “Close” | | |
| **Risultati attesi:** | L’immagine verrà scartata. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-006  REQ-002 | **Nome:** | Cambiamento layer |
| **Descrizione:** | Si deve poter cambiare tra layer del disegno e layer dei puntini. | | |
| **Prerequisiti:** | Pagina iniziale completata, caricamento immagine, funzioni puntini e disegno, funzioni layer. | | |
| **Procedura:** | 1. Cliccare “Open” 2. Selezionare una immagine 3. Mettere qualche puntino a caso 4. Cliccare per andare nel layer dei disegni “Drawing” | | |
| **Risultati attesi:** | I puntini non saranno più visibili e si vedranno gli strumenti di disegno nel menu. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-007  REQ-002  REQ-004 | **Nome:** | Visualizzazione sovrapposta dei layer |
| **Descrizione:** | Si devono poter vedere i due layer insieme. | | |
| **Prerequisiti:** | Pagina iniziale completata, caricamento immagine, funzioni puntini e disegno, funzioni layer. | | |
| **Procedura:** | 1. Cliccare “Open” 2. Selezionare una immagine 3. Mettere qualche puntino a caso 4. Cliccare per andare nel layer dei disegni “Drawing” 5. Aggiungere qualche disegno a caso 6. Schiacciare l’occhio accanto a “Dots” | | |
| **Risultati attesi:** | Si vedranno sia i disegni che i puntini nella stessa pagina. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-008  REQ-003 | **Nome:** | Aggiunta puntini |
| **Descrizione:** | Aggiunta di puntini sul canvas nel layer dei puntini. | | |
| **Prerequisiti:** | Pagina iniziale completata, caricamento immagine, funzioni puntini. | | |
| **Procedura:** | 1. Cliccare “Open” 2. Selezionare una immagine 3. Selezionare l’elemento con l’icona dei puntini 4. Schiacciare nel canvas | | |
| **Risultati attesi:** | Il puntino sarà aggiunto | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-009  REQ-003 | **Nome:** | Eliminazione puntini |
| **Descrizione:** | Eliminazione di un puntino. | | |
| **Prerequisiti:** | Pagina iniziale completata, caricamento immagine, funzioni puntini | | |
| **Procedura:** | 1. Cliccare “Open” 2. Selezionare una immagine 3. Selezionare l’elemento con l’icona dei puntini 4. Schiacciare nel canvas 5. Selezionare l’elemento con l’icona della freccia 6. Schiacciare il puntino 7. Selezionare l’elemento con l’icona del cestino | | |
| **Risultati attesi:** | Il puntino selezionato sarà eliminato. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-010  REQ-003 | **Nome:** | Spostamento puntini |
| **Descrizione:** | I puntini selezionati possono essere spostati. | | |
| **Prerequisiti:** | Pagina iniziale completata, caricamento immagine, funzioni puntini. | | |
| **Procedura:** | 1. Cliccare “Open” 2. Selezionare una immagine 3. Selezionare l’elemento con l’icona dei puntini 4. Schiacciare nel canvas 5. Selezionare l’elemento con l’icona della freccia 6. Schiacciare il puntino 7. Tenere premuto e spostare il mouse | | |
| **Risultati attesi:** | Spostamento del puntino. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-011  REQ-003 | **Nome:** | Numero dei puntini cambiabile |
| **Descrizione:** | Il numero di un puntino può essere cambiato con un altro. | | |
| **Prerequisiti:** | Pagina iniziale completata, caricamento immagine, funzioni puntini. | | |
| **Procedura:** | 1. Cliccare “Open” 2. Selezionare una immagine 3. Selezionare l’elemento con l’icona dei puntini 4. Schiacciare nel canvas 5. Selezionare l’elemento con l’icona della freccia 6. Schiacciare il puntino 7. Selezionare l’elemento con l’icona dei numeri 8. Inserire il numero di un puntino esistente | | |
| **Risultati attesi:** | Il numero di tutti i puntini verrà cambiato per far rispettare la modifica avvenuta | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-012  REQ-003 | **Nome:** | Soluzione puntini |
| **Descrizione:** | È possibile vedere i puntini collegati. | | |
| **Prerequisiti:** | Pagina iniziale completata, caricamento immagine, funzioni puntini. | | |
| **Procedura:** | 1. Cliccare “Open” 2. Selezionare una immagine 3. Selezionare l’elemento con l’icona dei puntini 4. Schiacciare nel canvas un po’ di volte in posti diversi 5. Schiacciare “Connect dots” | | |
| **Risultati attesi:** | Delle linee appariranno tra i puntini per vedere la soluzione | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-013  REQ-004 | **Nome:** | Disegno a mano libera |
| **Descrizione:** | Possibilità di disegnare linee a mano libera | | |
| **Prerequisiti:** | Pagina iniziale completata, caricamento immagine, funzioni disegno. | | |
| **Procedura:** | 1. Cliccare “Open” 2. Selezionare una immagine 3. Selezionare il layer dei disegni 4. Selezionare la penna e disegnare. | | |
| **Risultati attesi:** | Le linee a mano libera saranno disegnate. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-014  REQ-004 | **Nome:** | Cancellazione disegni a mano libera |
| **Descrizione:** | Usando la gomma sarà possibile cancellare i disegni fatti a mano libera. | | |
| **Prerequisiti:** | Pagina iniziale completata, caricamento immagine, funzioni disegno. | | |
| **Procedura:** | 1. Cliccare “Open” 2. Selezionare una immagine 3. Selezionare il layer dei disegni 4. Selezionare la penna e disegnare 5. Selezionare la gomma e cancellare il disegno | | |
| **Risultati attesi:** | Il disegno sarà cancellato mano a mano che ci andiamo sopra. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-015  REQ-004 | **Nome:** | Aggiunta forme disegnate. |
| **Descrizione:** | Aggiunta di rettangoli e/o elissi. | | |
| **Prerequisiti:** | Pagina iniziale completata, caricamento immagine, funzioni disegno. | | |
| **Procedura:** | 1. Cliccare “Open” 2. Selezionare una immagine 3. Selezionare il layer dei disegni 4. Selezionare il rettangolo o l’ellissi 5. Disegnare | | |
| **Risultati attesi:** | La forma selezionata sarà disegnata. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-016  REQ-004 | **Nome:** | Spostamento forma disegnata |
| **Descrizione:** | Spostamento di rettangoli e/o elissi. | | |
| **Prerequisiti:** | Pagina iniziale completata, caricamento immagine, funzioni disegno. | | |
| **Procedura:** | 1. Cliccare “Open” 2. Selezionare una immagine 3. Cliccare “Open” 4. Selezionare una immagine 5. Selezionare il layer dei disegni 6. Selezionare il rettangolo o l’ellissi 7. Disegnare 8. Selezionare la freccia 9. Cliccare la forma 10. Spostare | | |
| **Risultati attesi:** | La forma selezionata sarà spostata | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-017  REQ-004 | **Nome:** | Cancellazione forma |
| **Descrizione:** | Cancellazione di rettangoli e/o elissi. | | |
| **Prerequisiti:** | Pagina iniziale completata, caricamento immagine, funzioni disegno. | | |
| **Procedura:** | 1. Cliccare “Open” 2. Selezionare una immagine 3. Cliccare “Open” 4. Selezionare una immagine 5. Selezionare il layer dei disegni 6. Selezionare il rettangolo o l’ellissi 7. Disegnare 8. Selezionare la freccia 9. Cliccare la forma 10. Schiacciare il cestino | | |
| **Risultati attesi:** | La forma selezionata verrà cancellata | | |

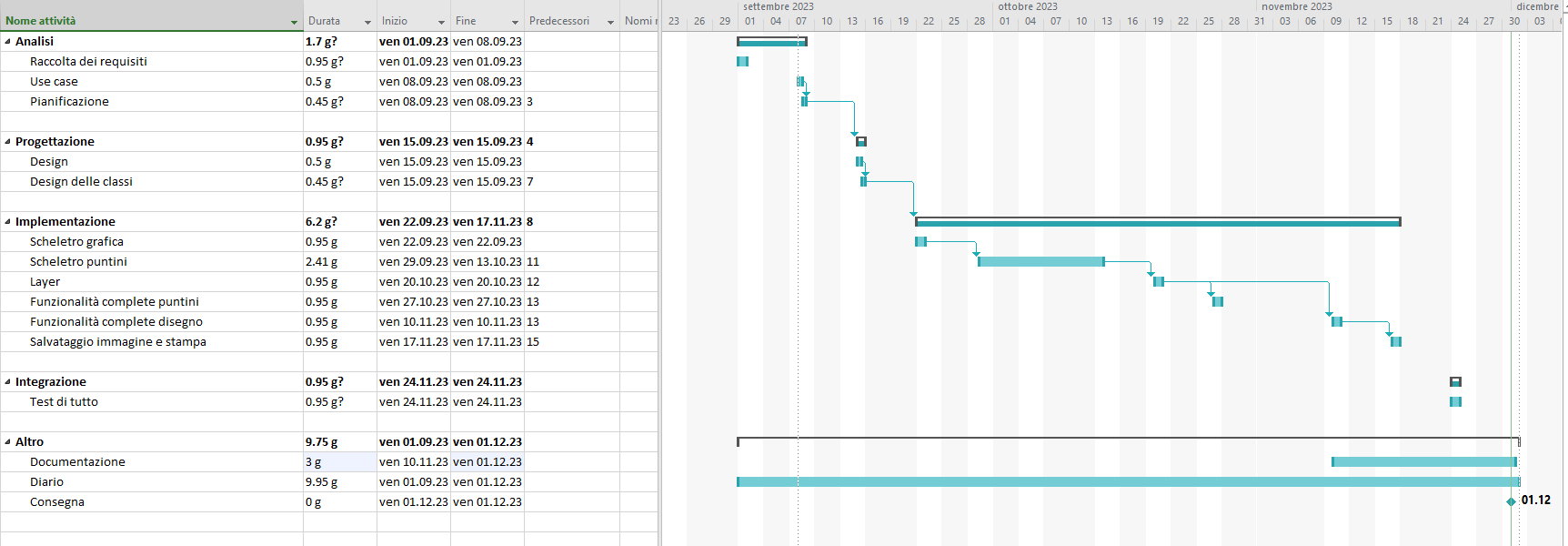
## Risultati test

|  |  |  |
| --- | --- | --- |
| Test case | Risultato | Stato |
| TC-001 | Immagine caricata | Passato |
| TC-002 | L’immagine caricata non esce fuori dallo spazio riservato dell’immagine. | Passato |
| TC-003 | Una immagine png è stata scaricata, trasparente e contiene i puntini | Passato |
| TC-004 | Il menu di stampa del browser è stato aperto. | Passato |
| TC-005 | L’immagine è stata scartata. | Passato |
| TC-006 | I puntini non sono più visibili e si vedono gli strumenti di disegno nel menu. | Passato |
| TC-007 | Si vedono sia i disegni che i puntini nella stessa pagina. | Passato |
| TC-008 | Il puntino è stato aggiunto | Passato |
| TC-009 | Il puntino selezionato è stato eliminato. | Passato |
| TC-010 | Il puntino è stato spostato. | Passato |
| TC-011 | Il numero di tutti i puntini è stato cambiato per far rispettare la modifica avvenuta. | Passato |
| TC-012 | Delle linee sono apparse tra i puntini per vedere la soluzione. | Passato |
| TC-013 | Le linee a mano libera sono state disegnate. | Passato |
| TC-014 | Il disegno è stato cancellato mano a mano che ci andiamo sopra. | Passato |
| TC-015 | La forma selezionata è stata disegnata. | Passato |
| TC-016 | La forma selezionata è stata spostata. | Passato |
| TC-017 | La forma selezionata è stata cancellata. | Passato |

## Mancanze/limitazioni conosciute

CSS problematico su Firefox.

# Consuntivo



# Conclusioni

## Sviluppi futuri

Funzioni di login, di salvataggio dei lavori recenti.

## Considerazioni personali

In conclusione posso dire che il progetto mi ha insegnato a lavorare in maniera un po’ più ordinata, anche se penso che per il prossimo progetto documenterò ogni lezione invece di lasciare tutto all’ultimo. Sono molto soddisfatto della grafica dell’applicativo e dell’usabilità che ho creato anche se ci sono piccolezze migliorabili. All’inizio l’idea di lavorare a questo tipo di progetto mi ha un po’ spaventato, ma me la sono gestita bene ed ho risolto la maggior parte delle tematiche nelle prime settimane.

# Glossario

|  |  |
| --- | --- |
| **Termine** | **Descrizione** |
| HTML | Hypertext markup language, usata in informatica, e in modo particolare nella tecnica della rete Internet, per indicare un linguaggio descrittore delle pagine web. |
| CSS | Cascading Style Sheets, in italiano fogli di stile. |
| JS | JavaScript è un linguaggio orientato a oggetti basato su prototipi. |
| Applicativo WEB | Si intende il sito web che verrà visualizzato a progetto completato. |

# Bibliografia

## Bibliografia per articoli di riviste:

## Bibliografia per libri

## Sitografia

* <https://codepen.io/JWLD/pen/JrayLd>, 15-09-2023
* <https://dirask.com/posts/JavaScript-draw-point-on-canvas-element-PpOBLD>, *Javascript draw point on canvas element – Dirask, 22-09-2023*
* <http://www.java2s.com/ref/javascript/html-canvas-remove-object-from-canvas.html>, HTML Canvas Remove object from canvas, 22-09-2023
* <https://codepen.io/zsolt555/pen/rpPXOB>, Drawing pen canvas, 22-09-2023