# Progetto Farfalle Interattive - Relazione Tecnica

# Autore: Alessio Prete – Matricola 533010

# 1. Introduzione e Obiettivi

Il progetto *Farfalle Interattive* è una visualizzazione dinamica e interattiva di dati multivariati realizzata con HTML, CSS e JavaScript tramite la libreria **D3.js**.  
L’obiettivo è rappresentare **8 farfalle**, ciascuna caratterizzata da 5 variabili quantitative (tutte positive), che controllano:

* Posizione X
* Posizione Y
* Dimensione delle ali
* Dimensione della testa
* Dimensione dell’addome

L’utente può interagire cliccando su una farfalla per **ruotare le variabili** (in senso orario) di tutte le farfalle contemporaneamente. Tenendo premuto il tasto **"R"** durante il click, la rotazione avviene in senso inverso.  
Le modifiche sono accompagnate da **transizioni animate fluide** e ogni farfalla mantiene un **colore fisso** per essere riconoscibile nelle trasformazioni.

# 2. Struttura del Progetto

Il progetto è composto da:

**index.html**: struttura della pagina, logica interattiva e stile CSS.

**dataset.json**: file esterno contenente i dati multivariati delle farfalle.

## 2.1. Componente HTML

La struttura della pagina include:

* <div class="container"> contenitore principale.
* **Titolo** e **box istruzioni** (.instructions) per l’uso dell’applicazione.
* **Legenda** che associa ciascuna variabile alla parte corrispondente della farfalla.
* *<div id="visualization">* dove D3.js disegna l’*<svg>* con tutte le farfalle.
* **Sezione statistiche** (.stats) con contatori di click e rotazioni.
* **Pannello variabili** (.variable-display) che mostra **tutte le farfalle** con i valori aggiornati in tempo reale e i loro **nomi in base al colore** (es. “Farfalla blu”, “Farfalla verde”).
* *<footer>* con crediti.

## 2.2. Componente CSS

### L'aspetto visivo è stato curato per offrire un'interfaccia moderna e reattiva. Le tecniche principali utilizzate includono:

* **Layout responsive** con Flexbox e CSS Grid.
* **Animazioni fluide** (**@keyframes**, **transition**) per hover, click e aggiornamenti.
* **Tema dinamico**: in modalità inversa (**reverse-mode**) cambia lo schema cromatico.
* **Animazione dei valori numerici**: la classe **.changed** colora e ingrandisce temporaneamente i numeri quando cambiano

# 3. Logica Applicativa (JavaScript e D3.js)

Questa è la componente più complessa del progetto, dove risiede tutta la logica dell'applicazione. Il datasetviene caricato direttamente da un file **dataset.json**.

## 3.1. Inizializzazione e Scale di D3.js

Un concetto fondamentale di D3 è quello delle **scale**. Nel progetto, sono state usate scale lineari (d3.scaleLinear) per mappare in modo proporzionale i valori dei dati (il **dominio** o *domain*) a valori di pixel per il posizionamento e le dimensioni (il **range**).

La logica chiave per soddisfare i requisiti del progetto risiede nella gestione sincrona della rotazione dei **dati** e dei **domini** delle scale. Ad ogni click:

1. Viene ruotato l'array vars di ogni farfalla.
2. Viene ruotato in modo parallelo un array di supporto chiamato domains, che contiene gli intervalli di input di ogni variabile.
3. Prima di ogni disegno, la funzione updateScaleDomains() riassegna a ogni scala il suo nuovo dominio.

Questo approccio garantisce che, anche se i valori cambiano posizione (ad esempio, un valore di Posizione Y finisce nello slot delle Ali), la scala corrispondente viene aggiornata per "capire" quel nuovo tipo di valore, evitando errori di rendering e mantenendo le proporzioni corrette.

## 3.2. Il "General Update Pattern" di D3

Per il rendering degli elementi SVG, è stato impiegato il **General Update Pattern** di D3, un meccanismo potente ed efficiente basato sulla sequenza di metodi .data(), .enter() e .merge():

* **.data(dataset, keyFunction):** Associa gli elementi del dataset agli elementi SVG sulla pagina. La funzione chiave (d => d.id) è fondamentale per permettere a D3 di riconoscere quale dato corrisponde a quale farfalla tra un aggiornamento e l'altro.
* **.enter():** Questa selezione contiene i "segnaposto" per i dati che non hanno ancora un elemento grafico corrispondente. Qui vengono creati i nuovi gruppi <g> per ogni farfalla al primo caricamento.
* **.merge():** Unisce la selezione enter (nuovi elementi) con la selezione update (elementi già esistenti), permettendo di applicare le stesse trasformazioni e stili a tutte le farfalle presenti sulla pagina in un colpo solo.

## 3.3. Gestione dell'Interattività e delle Transizioni

La funzione handleButterflyClick è il cuore dell'interazione. Al verificarsi di un click, essa esegue le seguenti operazioni in sequenza:

1. **Aggiorna le statistiche** di interazione.
2. **Ruota i dati:** Modifica fisicamente gli array vars di ogni oggetto nel dataset. La direzione della rotazione dipende dallo stato della variabile booleana isRPressed.
3. **Ruota i domini:** Aggiorna l'array domains in modo speculare alla rotazione dei dati.
4. **Innesca il ridisegno:** Chiama la funzione drawButterflies(dataset), che applica il General Update Pattern e aggiorna la visualizzazione.
5. **Anima le transizioni:** Utilizzando il metodo .transition() di D3, ogni cambiamento di attributo (posizione, forma, dimensione) viene animato in modo fluido nell'arco di circa 1/2 secondi, anziché avvenire istantaneamente.

## Gestione dell'Interattività e delle Transizioni

L’applicazione finale:

* Carica i dati da un file JSON esterno.
* Visualizza 8 farfalle colorate con proprietà legate a variabili quantitative.
* Aggiorna in tempo reale posizione e forma al click, con transizioni fluide.
* Mostra un pannello informativo con tutti i valori aggiornati e nomi in base al colore.