



ANDREA CAMPACI | IT ARCHITECT

# Micro Frontends

Web component approach con VUEJS

WORKSHOP





# Andrea Campaci

IT Architect @ Banca Mediolanum S.p.A.

[//andreacampaci.it](http://andreacampaci.it)

[info@andreacampaci.it](mailto:info@andreacampaci.it)

[GitHub](#)

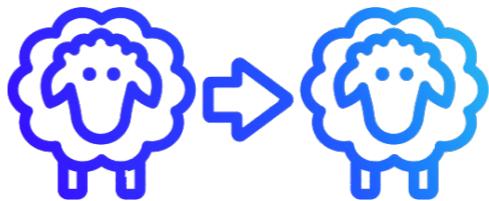


# Contenuti

## del workshop

- 01 Architettura
- 02 Web components
- 03 Vue CLI
- 04 Facciamo pratica
- 05 Q&A

# Repository di riferimento



[andreacampaci/micro-fe-vue2-wc](#)

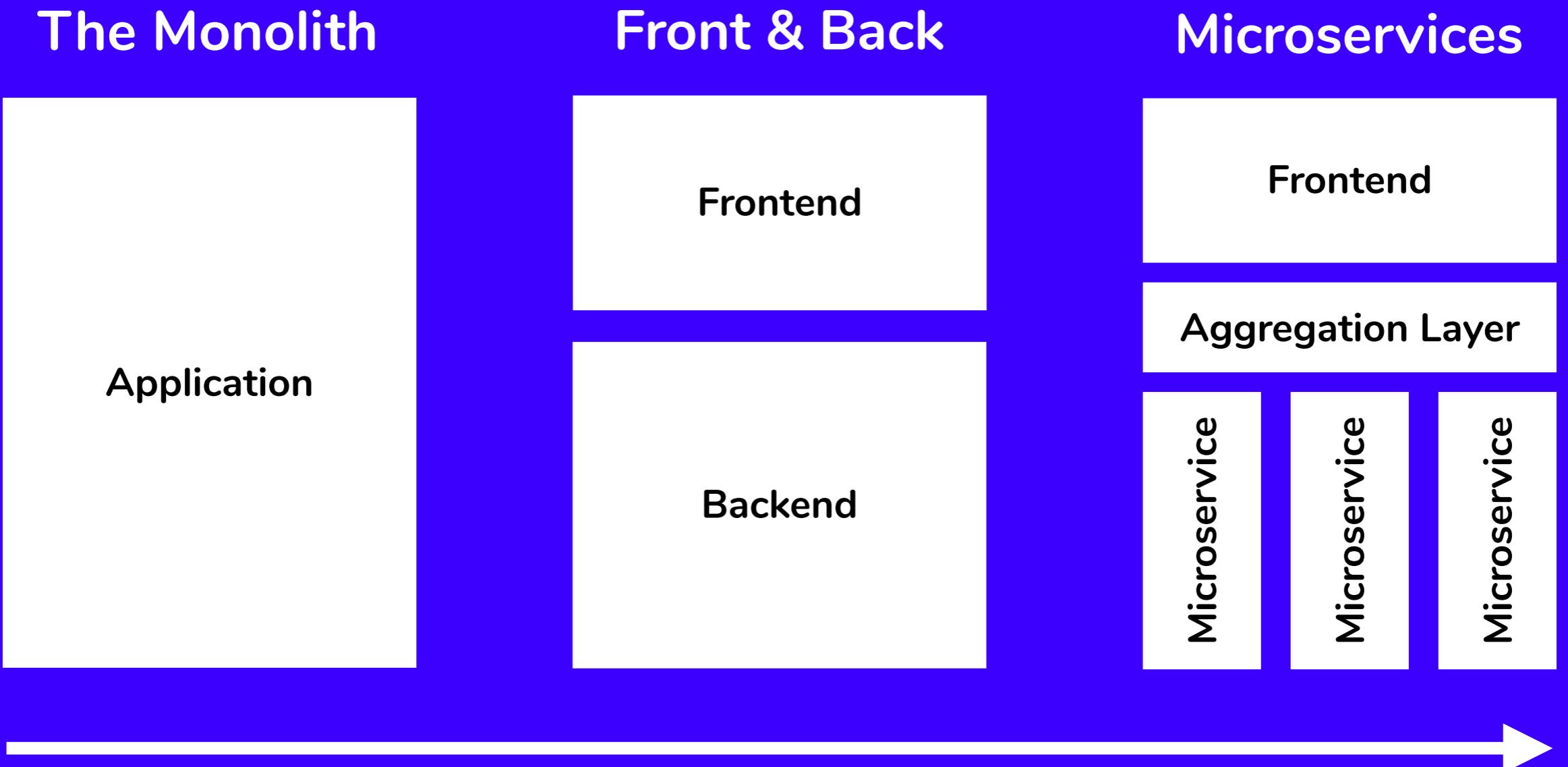
\$ git checkout workshop

\$ npm run bootstrap

Enjoy!!

# 01 Architettura

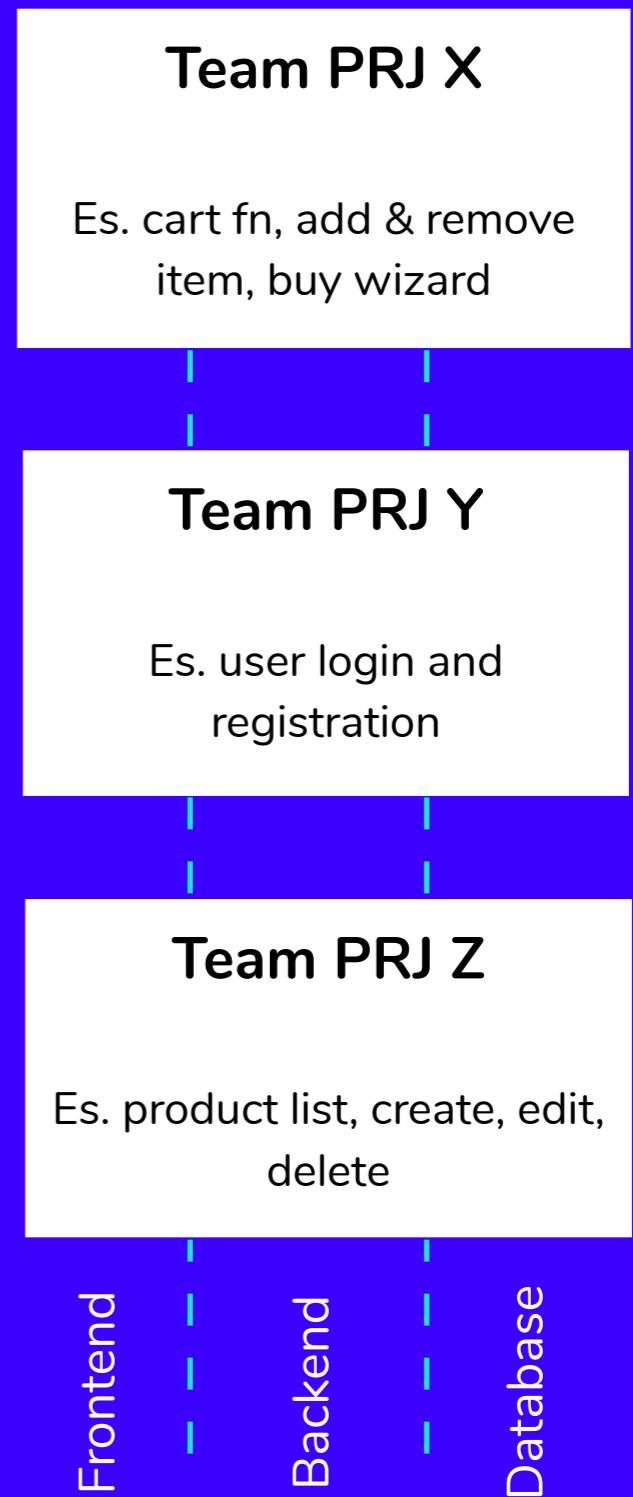
# 01 Architettura



# 01 Architettura

## Micro Frontends Overview

L'idea del pattern nasce dalla necessità di far realizzare a team indipendenti funzionalità business e2e.



# 01 Architettura

## Driver

## Principali



### Resilienza

- Seguendo il principio CCP (Common Closure Principle), ogni componente software è auto-consistente e responsabile di un solo dominio.
- Ogni micro applicazione gestisce il proprio error handling senza bloccare le altre in pagina.



### Performance

- Ogni micro applicazione FE ha un bundle estremamente piccolo, ottimizzabile ulteriormente utilizzando il code splitting.
- Alcuni file vendor possono essere esternalizzati per poi essere condivisi univocamente.



### Scalabilità

- Team con know how tecnologici differenti possono collaborare alla produzione di componenti software per un unico progetto.
- Parallelizzazione delle attività di sviluppo.
- Ogni micro app viene sviluppata, testata, rilasciata separatamente.



### Unlock

- Utilizzo di più framework.
- Ideale per chi vuole migrare la propria architettura verso un altro framework, prevenendo l'obsolescenza del software.

# Differenti approcci

## 01 Architettura

### WEB COMPONENT

- Standard HTML
- Reattività
- Isolamento CSS
- Necessita di polyfill in IE11

### IFRAME

- Forte isolamento
- Poca comunicazione
- Performance
- CORS issues

### SUB-DOMAIN

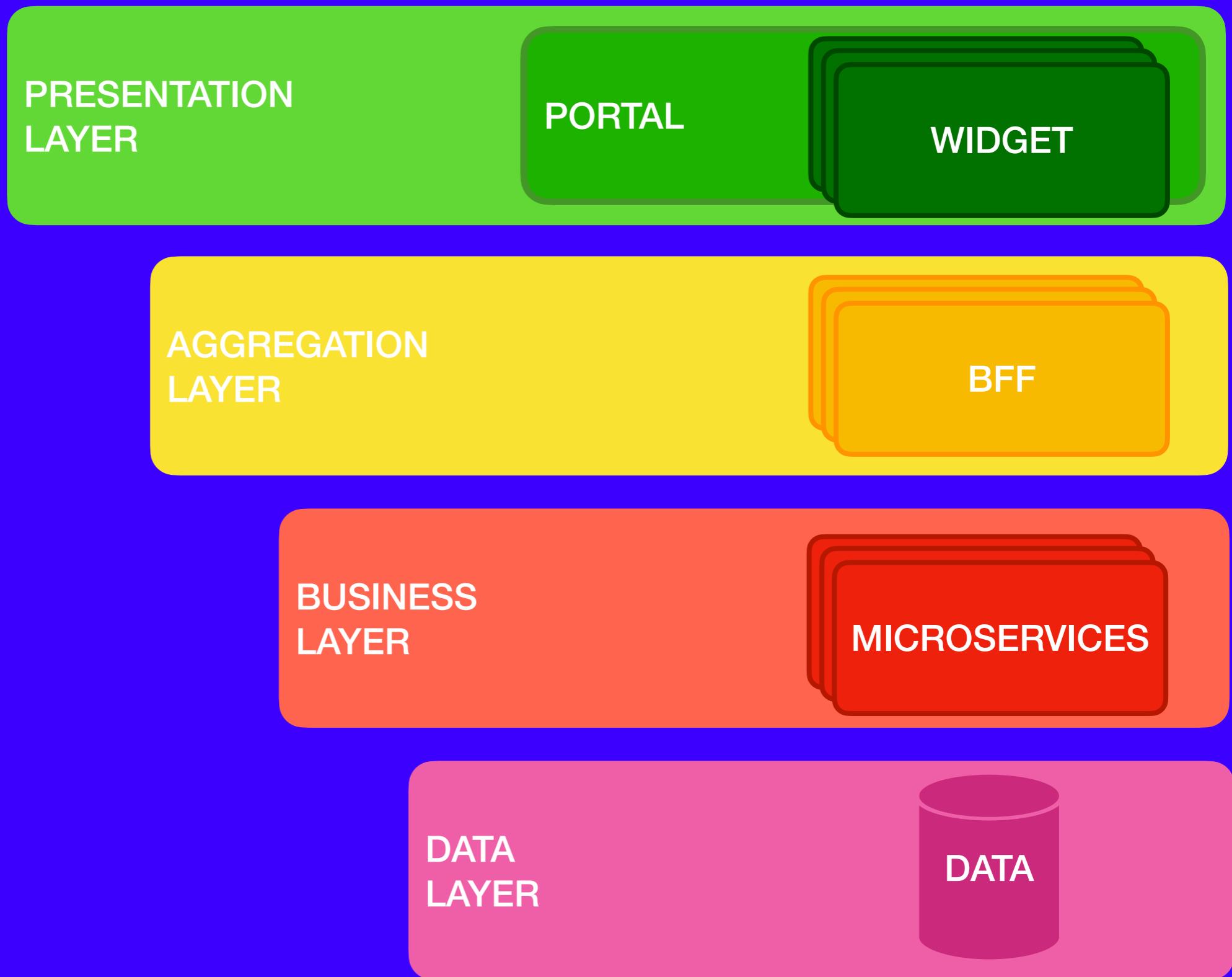
- Nessuna comunicazione
- Forte isolamento

### HTML FRAGMENTS

- Scomposizione
- Nessuno standard presente
- CSS presente in pagina

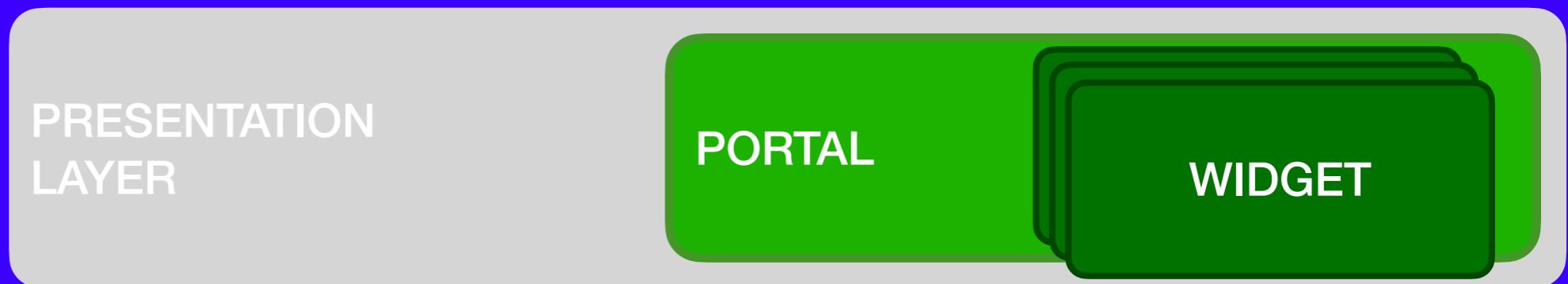
# Architettura logica

## 01 Architettura



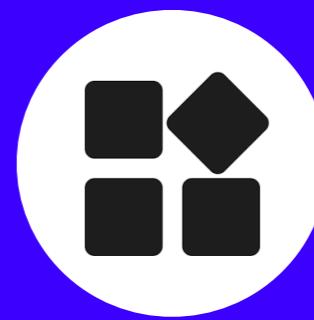
# 01 Architettura

## Principi di responsabilità



Portale (Shell)

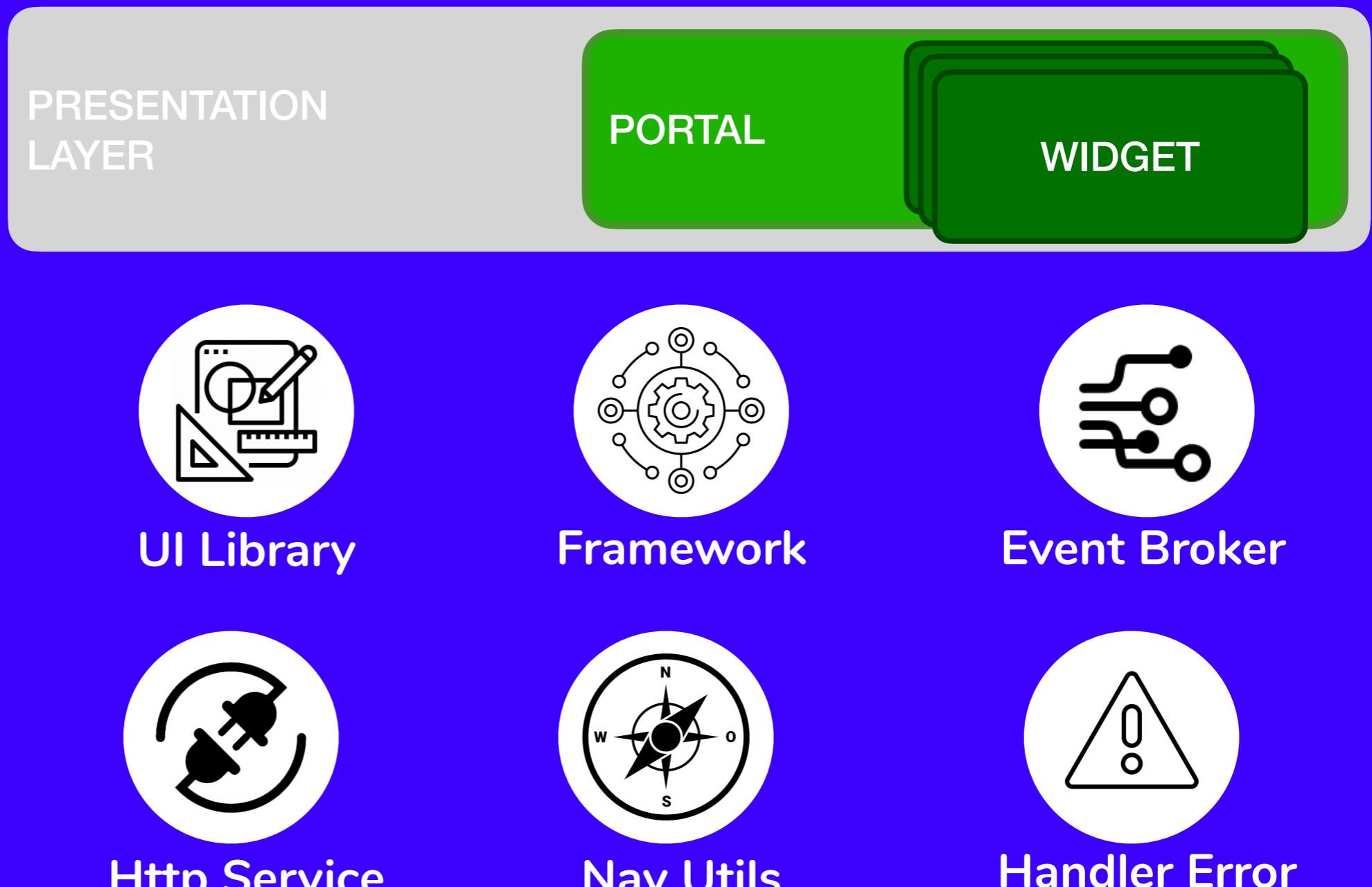
- Carica
  - Orchestra
  - Compone
  - Naviga
  - Autentica
- 
- Business
  - Business
  - Business



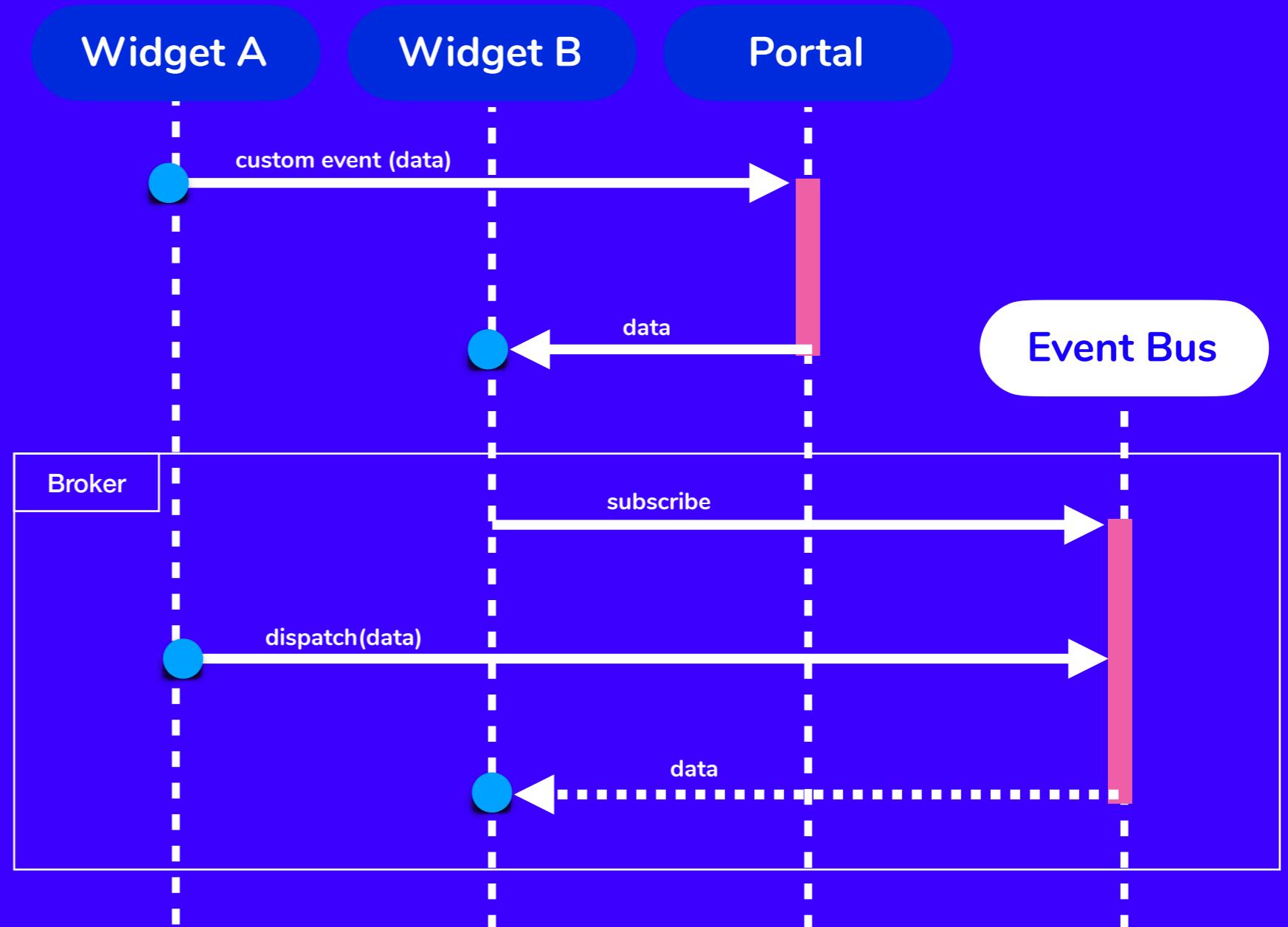
Widget

# 01 Architettura

Componenti a fattor comune



# Comunicazione tra widget

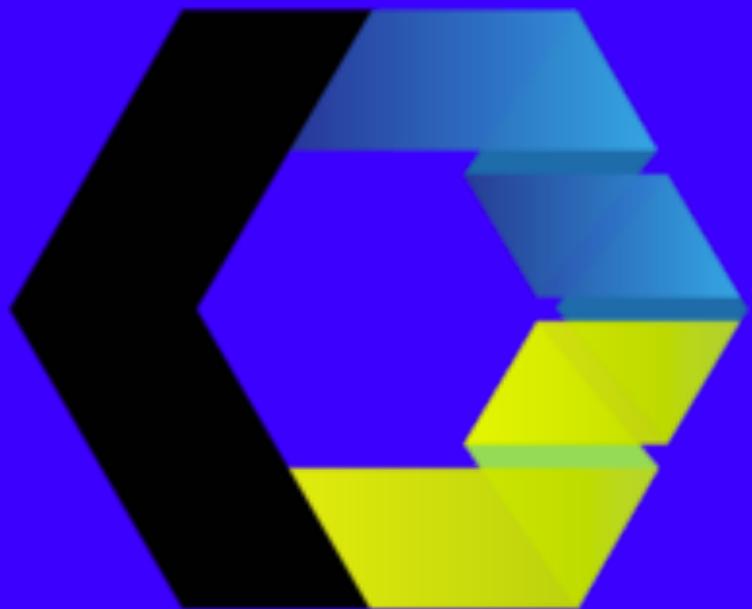


# 02 Web components

## 02 Web components

### Cosa sono?

Componenti create con diverse librerie/framework, le cui funzionalità sono incapsulate rispetto al codice presente in pagina e possono essere riutilizzate in differenti applicazioni web.



# Liberie

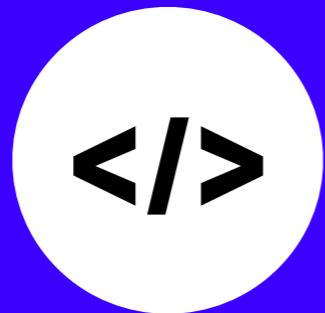
## e framework

# 02 Web components



# 02 Web components

## Principali proprietà



### Custom elements

I Custom Elements permettono di creare elementi HTML (Tags) nativi che possono implementare da una singola funzionalità ad un'intera applicazione.

Lo scopo di tali elementi è essere riutilizzabili.



### ESM (Moduli)

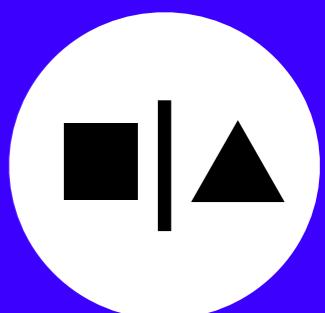
I moduli JS sono la base della tecnologia web component.

Questa funzionalità introdotta da JS permette di importare ed esportare codice JS in moduli rendendo riutilizzabile il codice in altre applicazioni.



### Shadow DOM

Un aspetto importante dei Web Component è l'incapsulamento: essere in grado di mantenere la struttura, lo stile e il comportamento separato da altro codice nella pagina.

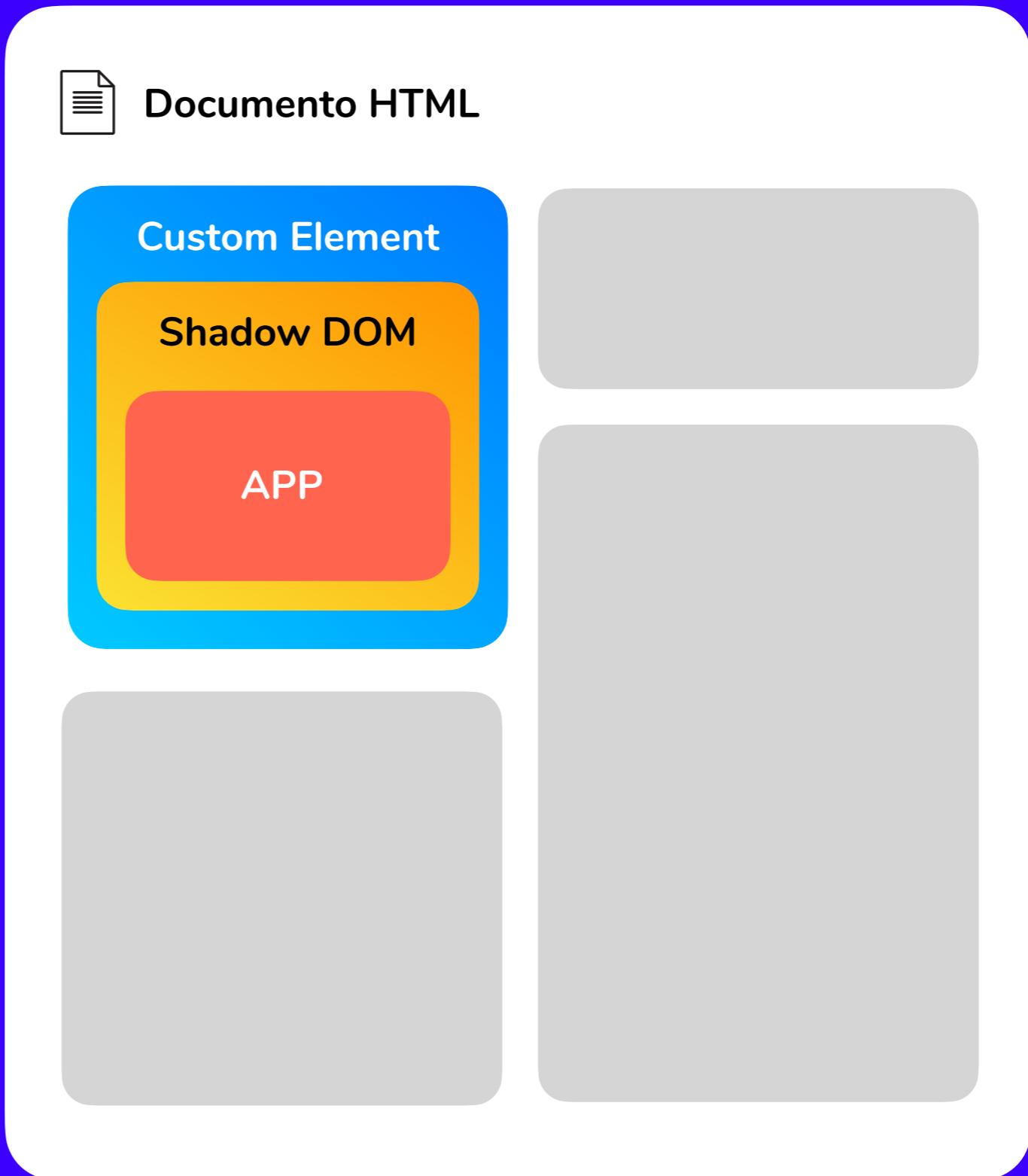


### CSS Scope

Il CSS viene isolato dall'esterno in modo che l'applicazione resti consistente indipendente dal contesto.

# Struttura logica

## 02 Web components



## 02 Web components

Supporto  
Browsers



# 03 Vue CLI

# 03 Vue CLI

## Creazione del componente

La creazione di un nuovo progetto avviene tramite **CLI** con il seguente comando:

```
$ vue create [widget-name]
```

Una volta che l'applicazione è completata, accodare al comando di build l'attributo **“target”** dal valore Web Component

```
$ vue-cli-service build --target wc
```

Proprietà  
input  
.in

L'jection di dati dall'esterno avvengono tramite **data-binding**; Sfruttando quindi le proprietà del **componente principale** Vue (App.vue) possiamo definire quali sono i dati che possiamo passare in input al nostro Web Component.

# 03 Vue CLI

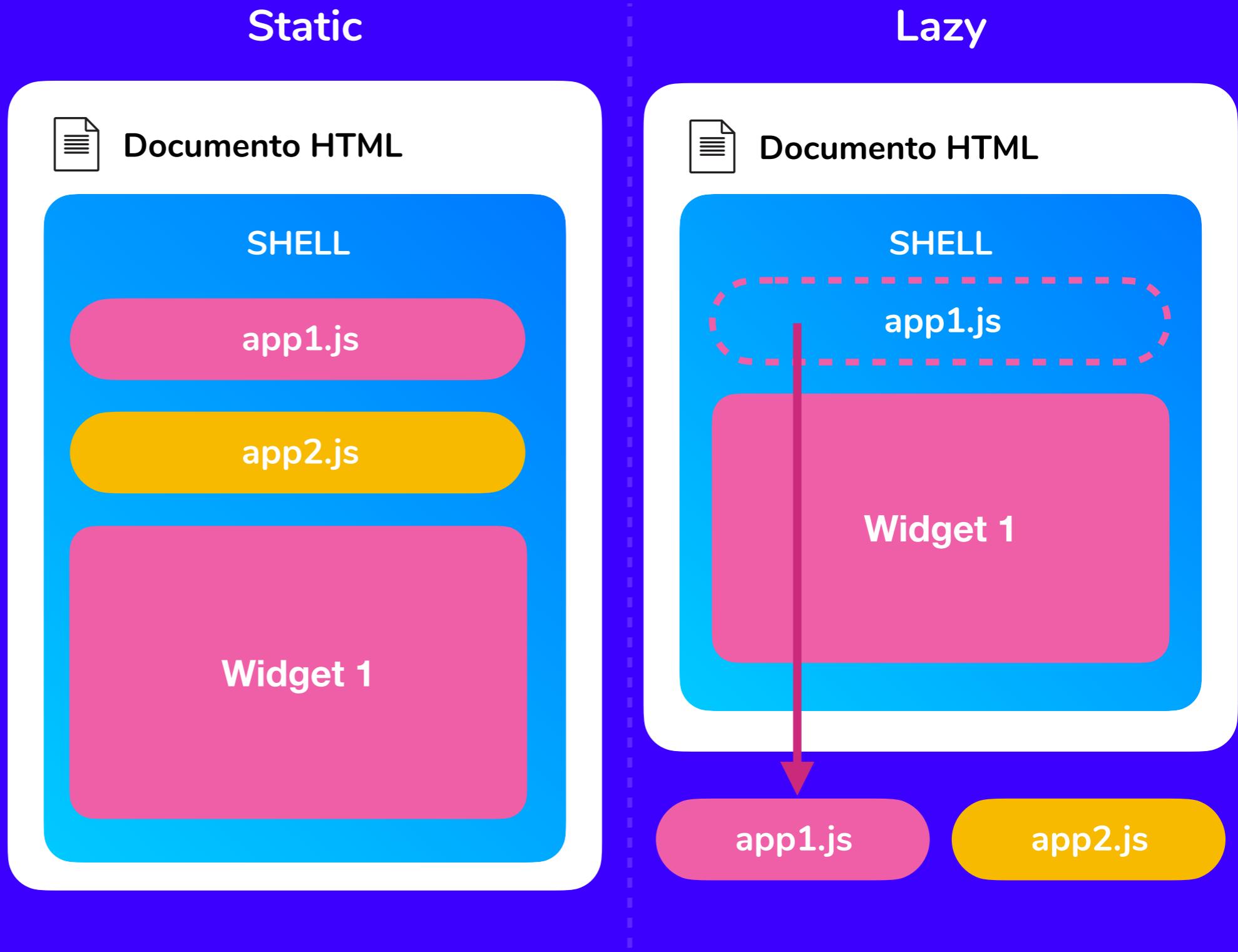
Dispatch  
di eventi  
in

Allo stesso modo delle proprietà, ogni evento lanciato dal **componente principale** (App.vue) viene automaticamente trasformato in un **Custom Event**. L'evento pubblicato potrà essere ascoltato sul **Custom Element**.

```
this.$emit("event-name", data)
```

# 03 Vue CLI

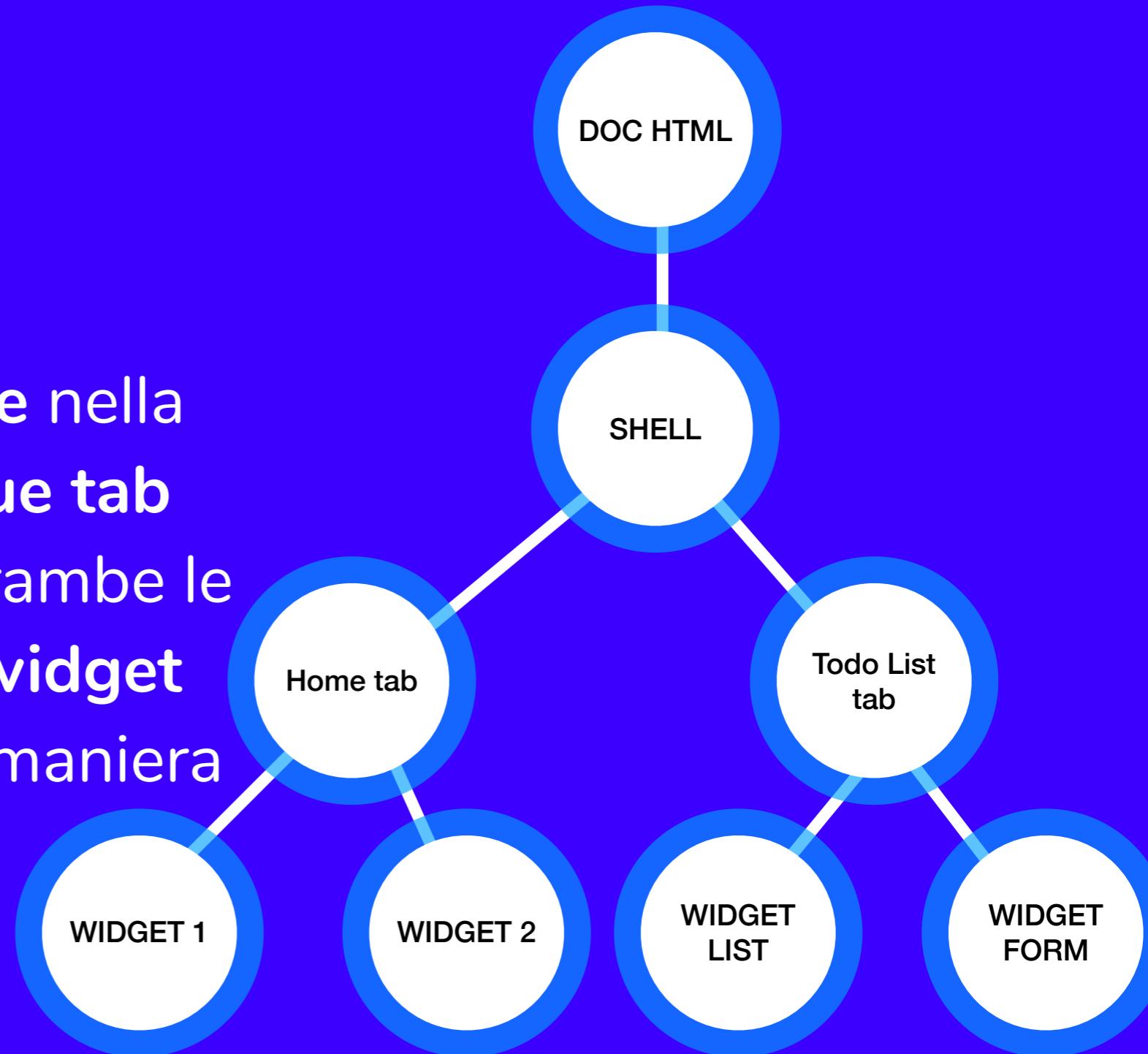
Caricamento  
da una Shell



# 04 Facciamo pratica

## Struttura del progetto

Creeremo un **applicazione** nella quale saranno presenti **due tab** “Home” e “Todo list”. Entrambe le pagine ospiteranno **due widget** diversi tra loro caricati in maniera **Lazy**.



# Q&A

# Thank 🙏

[info@andreacampaci.it](mailto:info@andreacampaci.it)