

**SUPSI**

# Postman with React

---

Studente/i

Dominik Panzarella  
Alessandro Spagnuolo

---

Corso di laurea

Ingegneria Informatica

Modulo / Codice Progetto

Web Applications 2

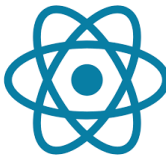
Anno

2024-2025

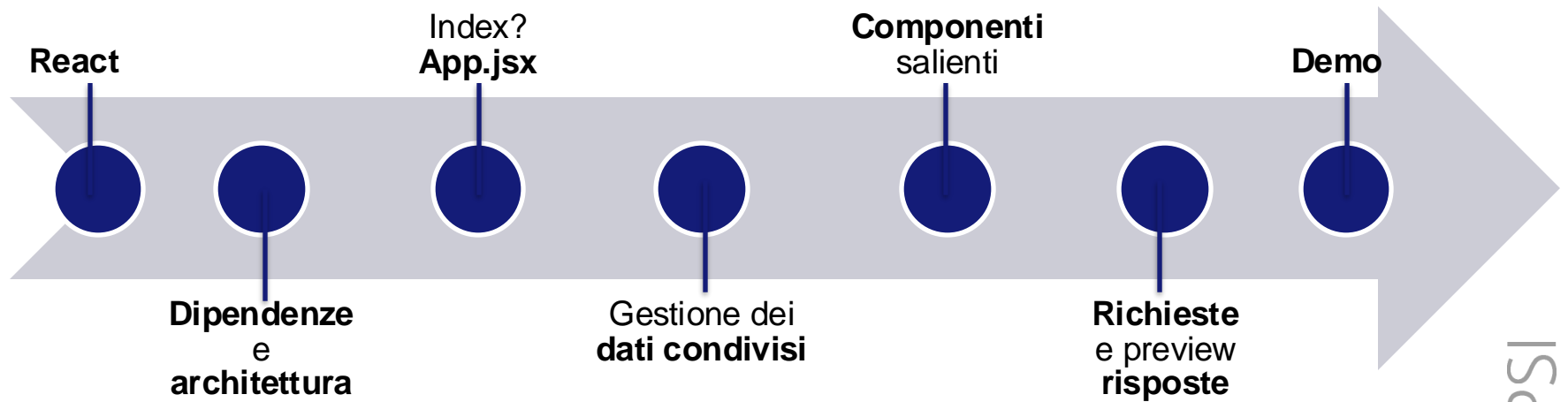
---

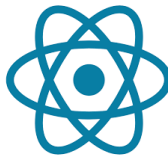
Data

08.04.2025

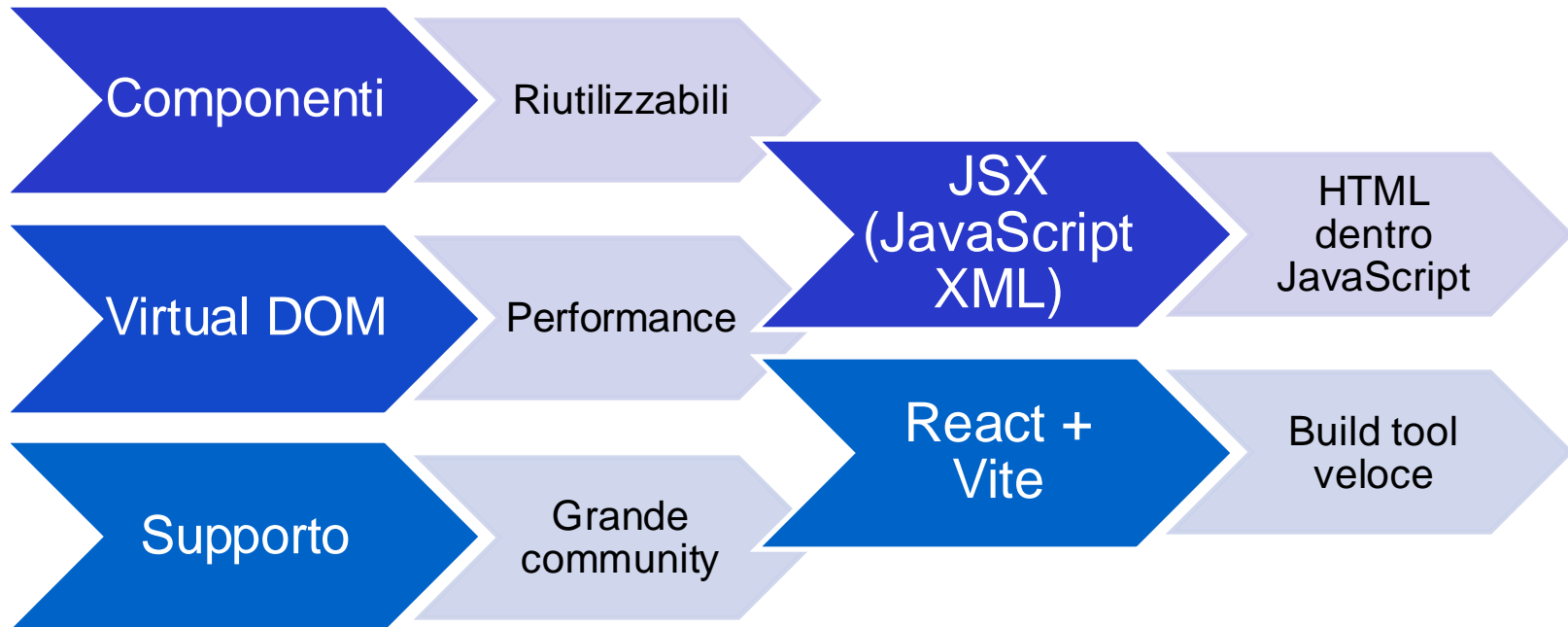


# Indice

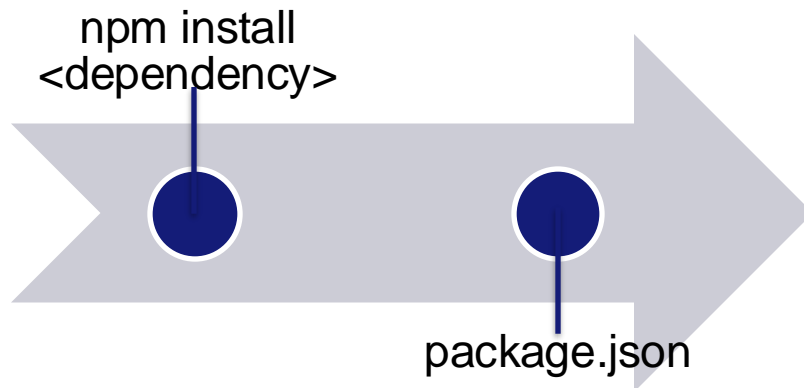




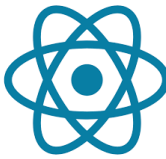
# React



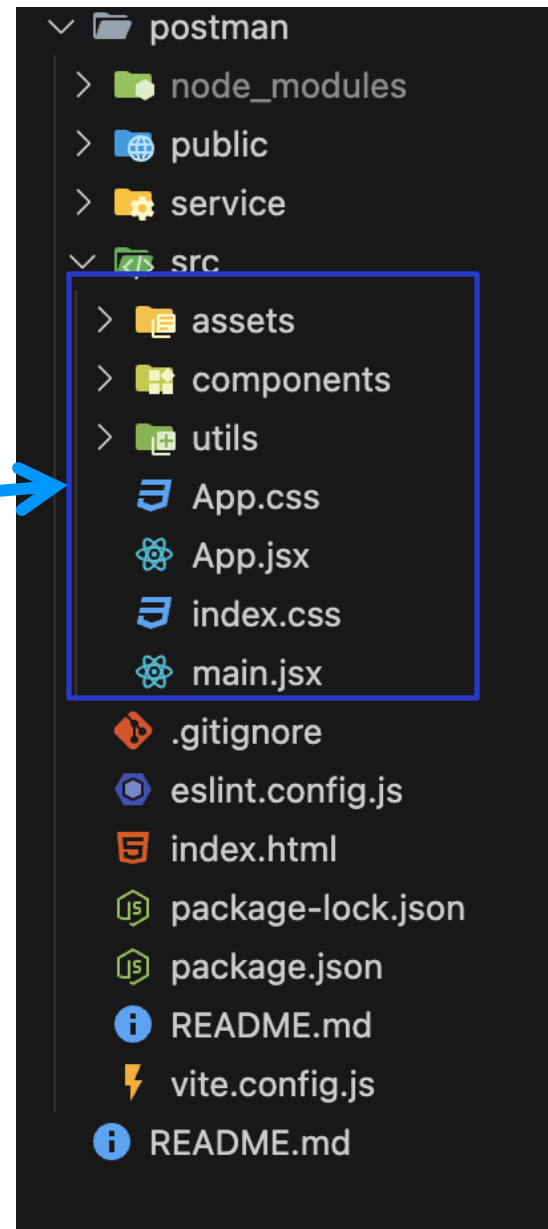
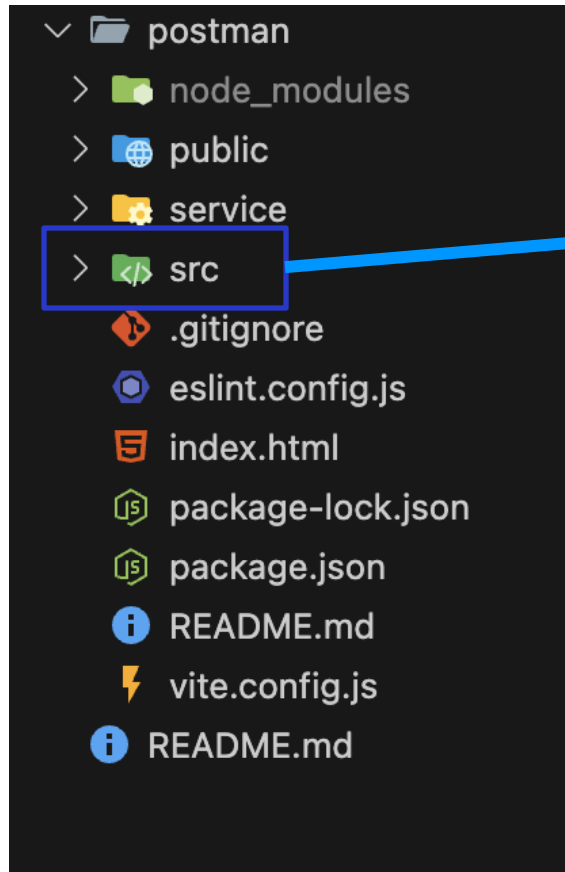
# Dipendenze

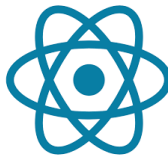


```
1  "dependencies": {  
2    "@emotion/react": "^11.14.0",  
3    "@emotion/styled": "^11.14.0",  
4    "@mui/icons-material": "^6.4.6",  
5    "@mui/material": "^6.4.7",  
6    "@mui/styles": "^6.4.6",  
7    "axios": "^1.8.2",  
8    "react": "^19.0.0",  
9    "react-dom": "^19.0.0",  
10   "react-icons": "^5.5.0",  
11   "react-router-dom": "^7.2.0"  
12 },
```

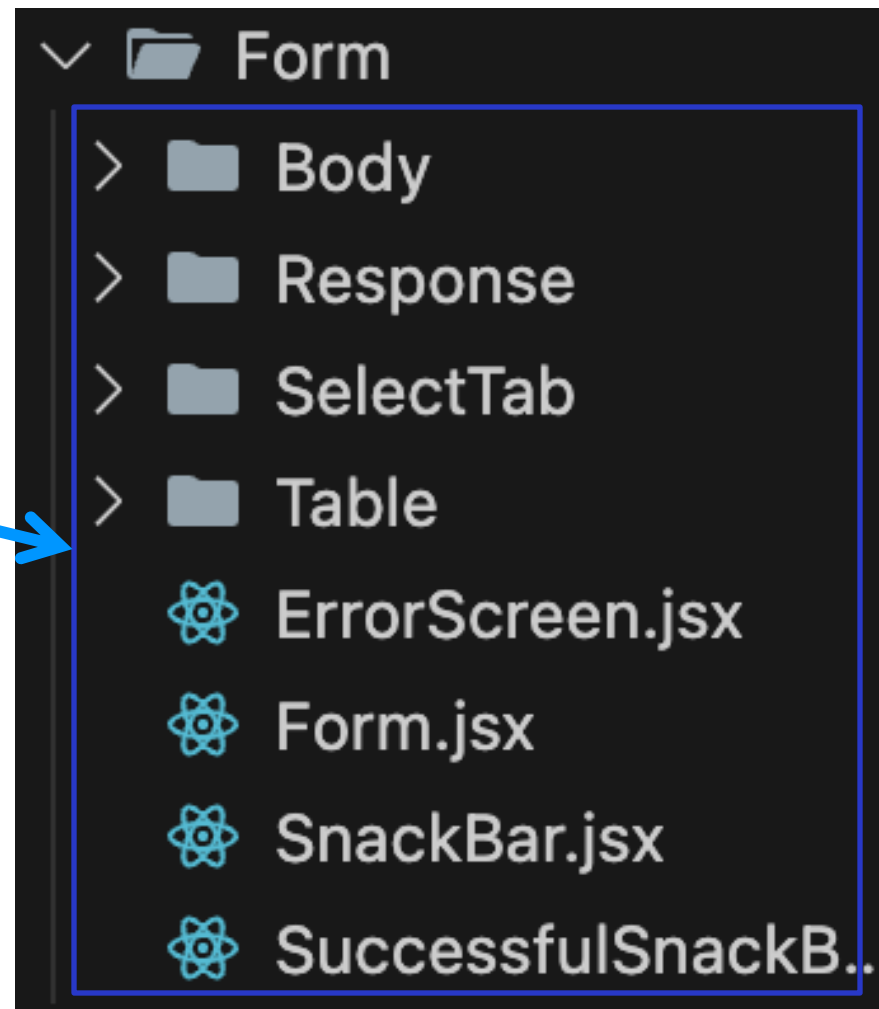
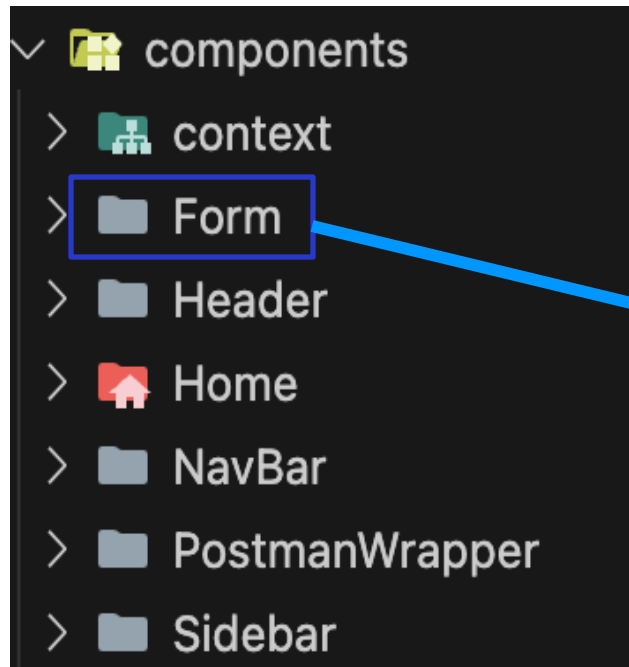


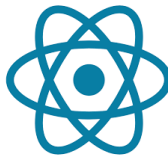
## Architettura





## Architettura





# App.jsx:

## Utilizzo di PostmanWrapper

### Configure PostmanWrapper

API URL

<https://supsi-ticket.cloudns.org/supsi-http-client/bff>

- ☒ Enable Search bar
- ☒ Enable Sidebar with Collections

Action on Response Click

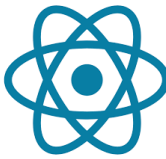
Show alert with the status code



```
1 <Box sx={{ mt: 4, width: '100%' }}>
2   <PostmanWrapper
3     url={config.url}
4     search={config.search}
5     collections={config.collections}
6     onResponseMessageClick={responseHandlers[config.responseHandler]}
7   />
8 </Box>
```

props





## Componente PostmanWrapper

```
1 import DataProvider from '../context/DataProvider';
2 import Home from '../Home/Home';
3
4 function PostmanWrapper({url, search, collections, onResponseMessageClick}) {
5   return (
6     <DataProvider>
7       <Home url={url}
8         search={search}
9         collections={collections}
10        onResponseMessageClick={onResponseMessageClick}>
11     </Home>
12   </DataProvider>
13 );
14 }
15 export default PostmanWrapper;
```

Componenti  
da importare

Componente  
da esportare

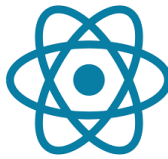




# DataProvider



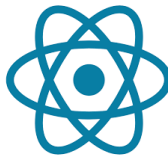
```
1 import React, { useState, createContext } from 'react';  
2 export const DataContext = createContext(null);
```



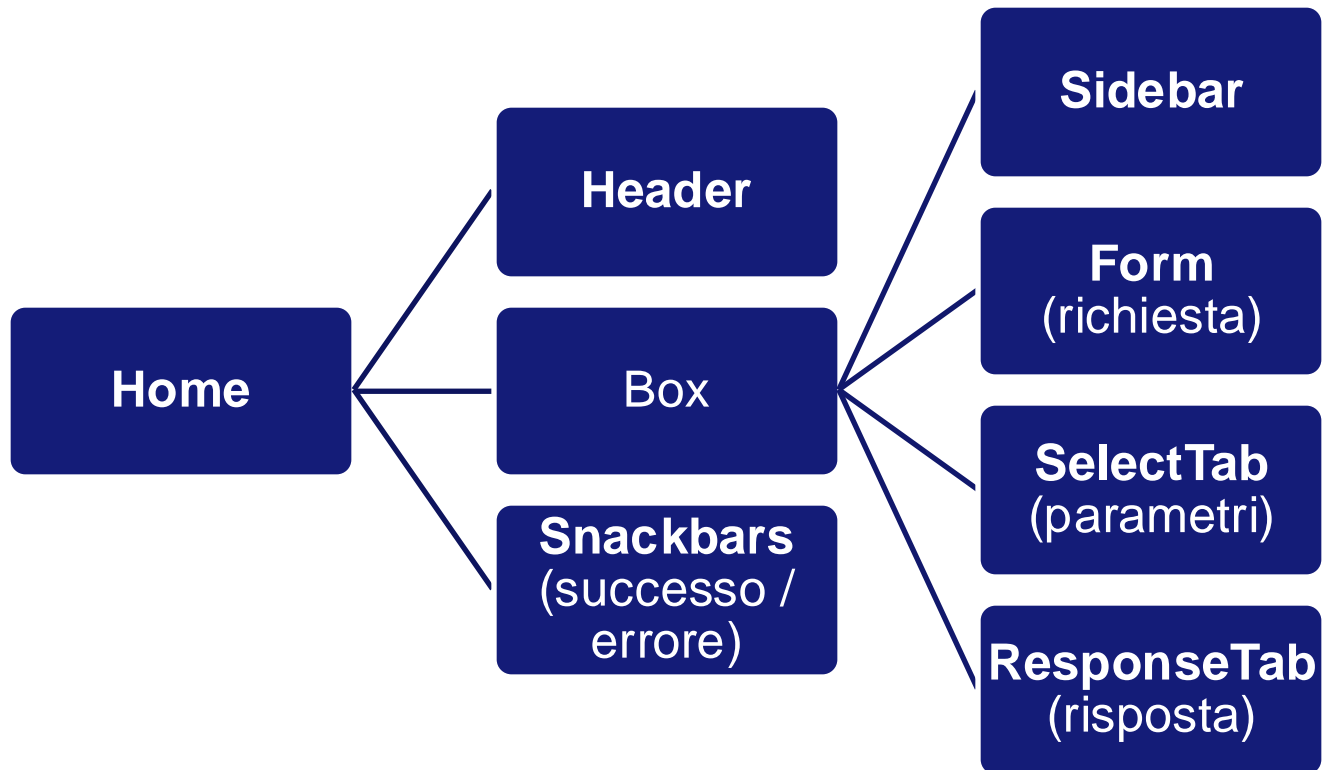
## DataProvider - Example

**state management**  
per la **condivisione**  
di stato tra  
componenti.

```
1 function PostmanWrapper({url, search, collections, onResponseMessageClick}) {  
2   return (  
3     <DataProvider>  
4       <Home  
5         url={url} search={search}  
6         collections={collections}  
7         onResponseMessageClick={onResponseMessageClick}>  
8       </Home>  
9     </DataProvider>  
10  );  
11 }
```



## Componente Home



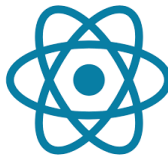
## Componente Home - sezione con la sidebar

```
1 {collections && (  
2   <Box sx={{...}}>  
3     <Box sx={{ width: '30%', minWidth: '250px', flexShrink: 0 }}>  
4       <Sidebar url={url} search={search}/>  
5     </Box>  
6     <Box sx={{...}}>  
7       <Form onSendClick={onSendClick} showSnackbar={showSnackbar}  
8         setError={setError} setSuccess={setSuccess}/>  
9       <SelectTab />  
10      {errorResponse ? <ErrorScreen /> :  
11        <ResponseTab apiResponse={apiResponse}  
12          onResponseMessageClick={onResponseMessageClick}/>  
13      </Box>  
14    </Box>  
15  )}
```

Check se mostrare la sidebar

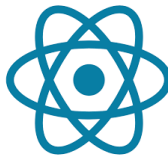
Manda avanti se mostrare la search bar o no

Manda avanti la funzione da chiamare al click sulla risposta



## Componente Home - sezione senza la sidebar

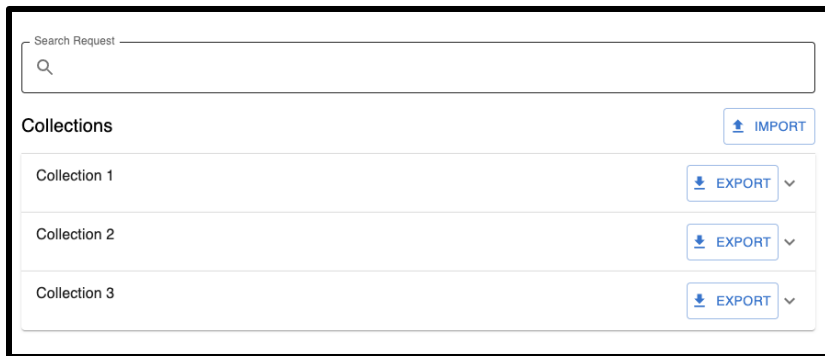
```
1  {!collections && (  
2    <Box sx={{...}}>  
3      <Form onSendClick={onSendClick} showSnackbar={showSnackbar}  
4        setError={setError} setSuccess={setSuccess}/>  
5      <SelectTab />  
6      {errorResponse ? <ErrorScreen /> :  
7        <ResponseTab apiResponse={apiResponse}  
8          onResponseMessageClick={onResponseMessageClick}/>}  
9    </Box>  
10 )}
```



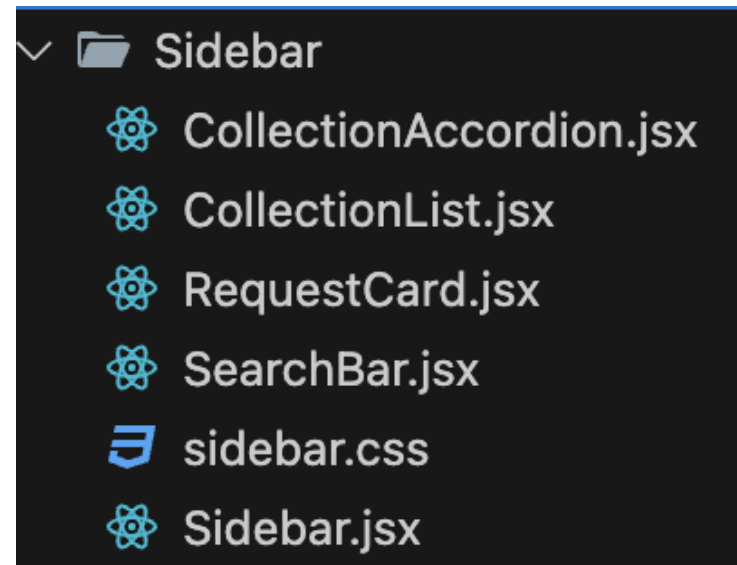
# Componente Sidebar



PostmanWrapper



→ Sidebar



# Componente Sidebar



PostmanWrapper

Search Request

Collections

IMPORT

Collection 1

EXPORT

Collection 2

EXPORT

PUT | Bag-End

PUT | Rivendell

DELETE | Last Desert

PUT | unknown

GET | unknown

Collection 3

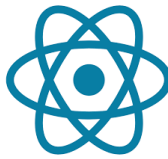
EXPORT

Searchbar

```

1  return (
2    <div className="http-client">
3      {collections && (
4        <div className="collections-sidebar">
5
6          {search && (
7            <SearchBar
8              onSearch={handleSearchChange}
9              searchTerm={searchTerm}></SearchBar>
10          )}
11
12          <input
13            type="file"
14            ref={fileInputRef}
15            style={{ display: 'none' }}
16            accept=".json"
17            onChange={processImportedFile}
18          />
19
20          <CollectionList
21            collectionsList={collectionsList}
22            expandedCollection={expandedCollection}
23            handleAccordionChange={handleAccordionChange}
24            handleExport={handleExport}
25            handleImport={handleImport}
26            filteredRequests={filterRequests()}
27            handleCardClick={handleCardClick}
28            classes={classes}
29          />
30        </div>
31      )}
32    </div>
33  );

```



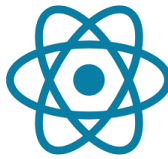
# Componente Sidebar

## CollectionList

CollectionAccordion ←

```
24
25 {collectionsList.map((collection) => (
26   <CollectionAccordion
27     key={collection.id}
28     collection={collection}
29     expanded={expandedCollection === collection.id}
30     onChange={handleAccordionChange(collection.id)}
31     onExport={() => handleExport(collection)}
32     requests={filteredRequests[collection.id] || []}
33     onRequestClick={handleCardClick}
34     classes={classes}
35   />
36 ))}
37 </>
38 );
39 };
```



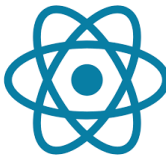


# Componente Sidebar

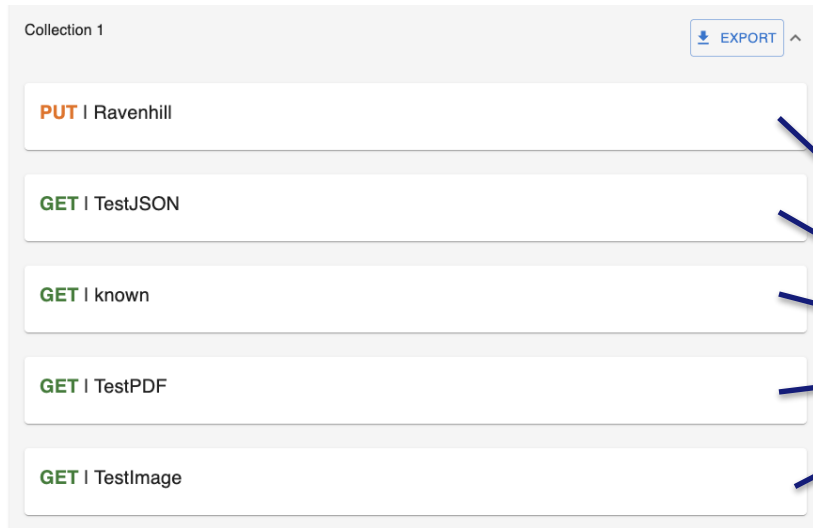
## CollectionAccordion

```
41     <AccordionDetails>
42       <Grid2 container direction="column" spacing={2}>
43         {requests.map((request) => (
44           <Grid2 item key={request.id}>
45             <RequestCard request={request} onClick={() => onRequestClick(request)} />
46           </Grid2>
47         ))}
48       </Grid2>
49     </AccordionDetails>
50   </Accordion>
51 );
52 };
```

RequestCard ←



# Componente Sidebar



RequestCard

# Componente Form



Request's name


POST

SEND >

Headers

Body

Headers

	KEY	VALUE	
<input type="checkbox"/>	<input type="text"/>	<input type="text"/>	
<input type="checkbox"/>	<input type="text"/>	<input type="text"/>	
<input type="checkbox"/>	<input type="text"/>	<input type="text"/>	
<input type="checkbox"/>	<input type="text"/>	<input type="text"/>	

+


Response



Enter the URL and click **SEND** to get a response.

# Componente Form

Richiesta esiste in  
una collection\*

 **PostmanWrapper**

Search Request

Collections

Collection 1

**PUT | Ravenhill**

**GET | TestJSON**

Headers Body

Headers

	KEY	VALUE	
<input type="checkbox"/>	Content-Length		<input type="button" value="X"/>
<input type="checkbox"/>	Content-Type	application/json	<input type="button" value="X"/>

# Componente Form

Richiesta NON esiste  
in una collection



Search Request

Collections

- Collection 1
- Collection 2
- Collection 3

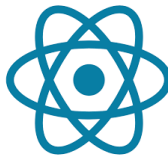
Request's name: rerererere

PUT

Headers

KEY	VALUE

Response



# Componente Form

Richiesta  
»pendente« in una  
collection

Search Request

Request's name  
NotAdded [+ CREATE](#)

Collections

Collection 1

[EXPORT](#)

**PUT** | Ravenhill

**GET** | TestJSON

**GET** | known

**GET** | TestPDF

**GET** | TestImage

**GET** | TestHTML

**POST** | TestX

**POST** | unknown

[IMPORT](#) **PUT** [SEND](#)


Headers Body

Headers

KEY	VALUE

[+](#)

Response



Enter the URL and click **SEND** to get a response.

# Componente Form

**PostmanWrapper**

Search Request  
Q

Collections

- Collection 1 **EXPORT**
- Collection 2 **EXPORT**
- Collection 3 **EXPORT**

Collection 1 **EXPORT**

- PUT** | Ravenhill
- GET** | TestJSON
- GET** | known
- GET** | TestPDF
- GET** | TestImage
- GET** | TestHTML
- POST** | TestX
- POST** | unknown


Request's name  
Ravenhill

**PUT** <https://supsi-bucket.cloudns.org/supsi-http-client/test-endpoints/lotr/random-character> **SEND**

**Headers** **Body**


	KEY	VALUE
<input type="checkbox"/>	Content-Length	
<input type="checkbox"/>	Content-Type	application/json

**Response**

  
Enter the URL and click SEND to get a response.

Collection importata

# Componente Form





 Request's name

POST

Headers

Body

Headers

	KEY	VALUE	
<input type="checkbox"/>	<input type="text"/>	<input type="text"/>	
<input type="checkbox"/>	<input type="text"/>	<input type="text"/>	
<input type="checkbox"/>	<input type="text"/>	<input type="text"/>	
<input type="checkbox"/>	<input type="text"/>	<input type="text"/>	

Response


SelectTab component



Enter the URL and click **SEND** to get a response.



# Componente Form

 Request's name


POST

Headers Body

Headers

	KEY	VALUE	
<input type="checkbox"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="X"/>
<input type="checkbox"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="X"/>
<input type="checkbox"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="X"/>
<input type="checkbox"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="X"/>

Response



Enter the URL and click **SEND** to get a response.

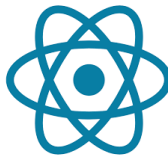
ResponseHandler  
component

# Gestione delle richieste

Incapsulamento  
Dati

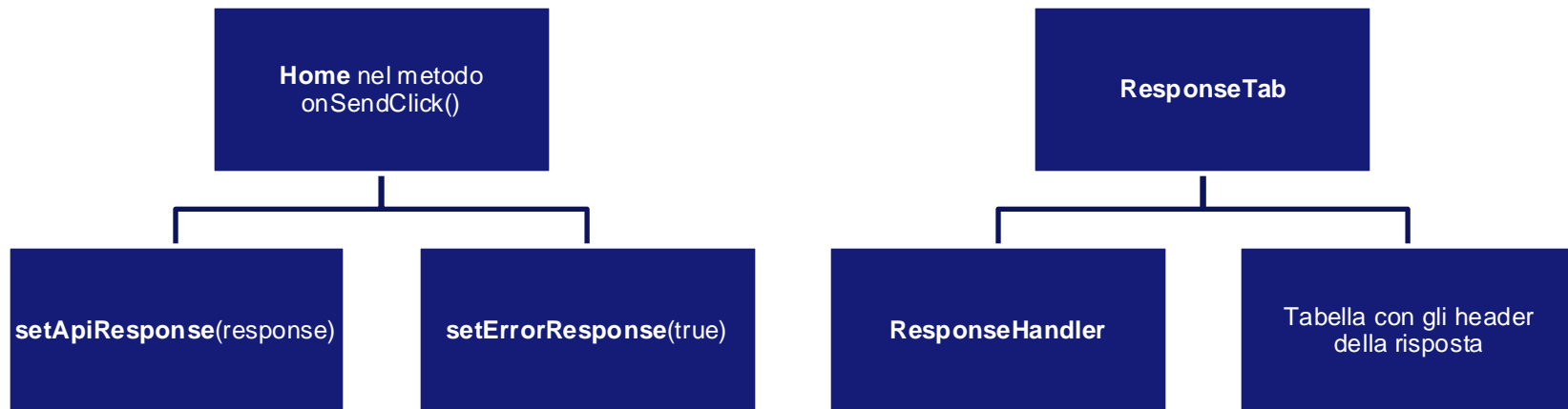
Proxy SUPSI

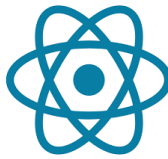
```
1 export const getData = async (formData, jsonText, paramData, headerData) => {
2   const apiType = formData.type;
3   const apiUrl = formData.url;
4   const apiHeaders = getHeadersAndParams(headerData);
5   const apiParams = getHeadersAndParams(paramData);
6   const data = apiType === 'GET' ? '' : jsonText;
7
8   const transformedHeaders = Object.entries(apiHeaders).reduce((acc, [key, value]) => {
9     acc[key] = Array.isArray(value) ? value : [String(value)];
10    return acc;
11  }, {});
12
13  const supsiData = {
14    method: apiType,
15    uri: apiUrl,
16    headers: transformedHeaders,
17    body: data
18  };
19
20  try {
21    const response = await axios({
22      method: 'POST',
23      url: 'https://supsi-ticket.cloudns.org/supsi-http-client/proxy/execute',
24      data: supsiData,
25      headers: {'Content-Type': ['application/json'], 'Accept': ['application/json']},
26    });
27
28    return response;
29  } catch (error) {
```



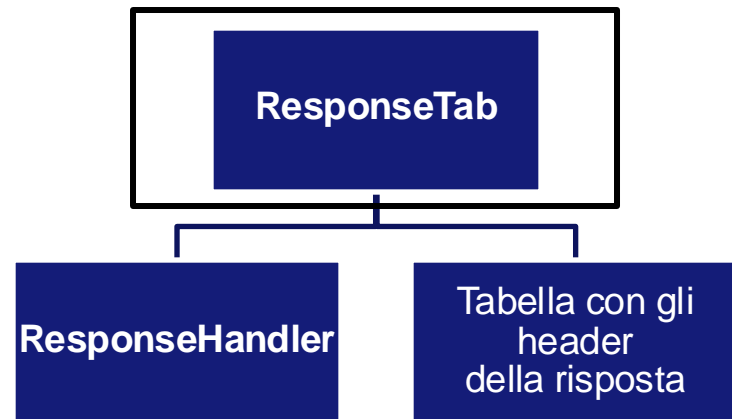
# Gestione delle risposte

```
{errorResponse ? <ErrorScreen /> :  
  <ResponseTab apiResponse={apiResponse}  
    onResponseMessageClick={onResponseMessageClick}/>}
```





# Gestione delle risposte



BODY

HEADERS

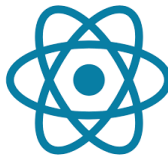
200 OK

98 MS

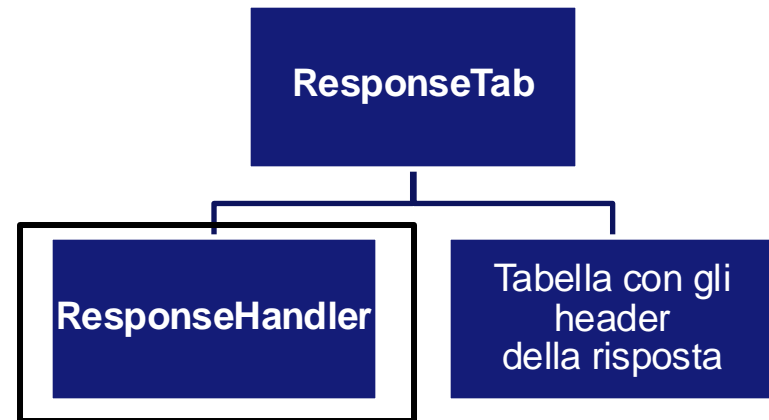
23.47 KB



```
1 <Box role="tabpanel" hidden={value !== 0}
2   id={`simple-tabpanel-${0}`}
3   aria-labelledby={`simple-tab-${0}`}>
4   <ResponseHandler apiResponse={apiResponse} onResponseMessageClick={onResponseMessageClick}></ResponseHandler>
5 </Box>
6 <Box role="tabpanel" hidden={value !== 1}
7   id={`simple-tabpanel-${1}`}
8   aria-labelledby={`simple-tab-${1}`}>
9   <CreateTable text="" data={convertArray(apiResponse.data.headers)} setData={() => {}} modifiable={false} />
10 </Box>
```



# Gestione delle risposte

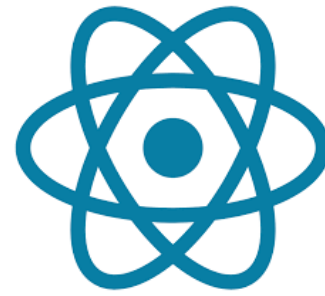


```

1 import HTMLHandler from "../HTML/HTMLHandler";
2 import IMAGEHandler from "../IMAGE/IMAGEHandler";
3 import JSONHandler from "../JSON/JSONHandler";
4 import PDFHandler from "../PDF/PDFHandler";
5
6 const ResponseHandler = ({apiResponse, onResponseMessageClick}) => {
7   const responseHeader = apiResponse.data.headers || '';
8   const contentTypeArray = responseHeader['Content-Type'] || '';
9   const contentTypeHeader = contentTypeArray[0];
10  const body = apiResponse.data.body || '';
11
12  if(contentTypeHeader.includes('json'))
13  {
14    return (<div onClick={()=>onResponseMessageClick(apiResponse)}><JSONHandler apiResponse={apiResponse}></JSONHandler></div>);
15  }else if(contentTypeHeader.includes('text/html') || (typeof data === 'string' && data.trim().startsWith('<'))){
16    return (<div onClick={()=>onResponseMessageClick(apiResponse)}><HTMLHandler apiResponse={apiResponse}></HTMLHandler></div>);
17  }else if (contentTypeHeader.includes('image/')) {
18    return (<div onClick={()=>onResponseMessageClick(apiResponse)}><IMAGEHandler apiResponse={apiResponse}></div>);
19  }else if (contentTypeHeader.includes('application/pdf')) {
20    return (<div onClick={()=>onResponseMessageClick(apiResponse)}><PDFHandler apiResponse={apiResponse}></div>);
21  }
22 };
23 export default ResponseHandler;
  
```

Import e uso dei relativi handler

Click sulla risposta



# Demo

