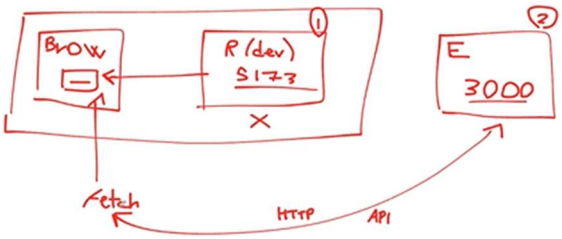
# 04_07 CLIENT-SERVER INTEGRATION REACT

In realtà noi abbiamo due server:

- Server react → lancia l'applicazione react (localhost 5173)
    - Che a sua volta mostra, all'interno di un browser, l'applicazione react
- Server express (localhost 3000)

*(nel nostro caso i due server sono sullo stesso pc ma possono anche essere su diversi dispositivi)*

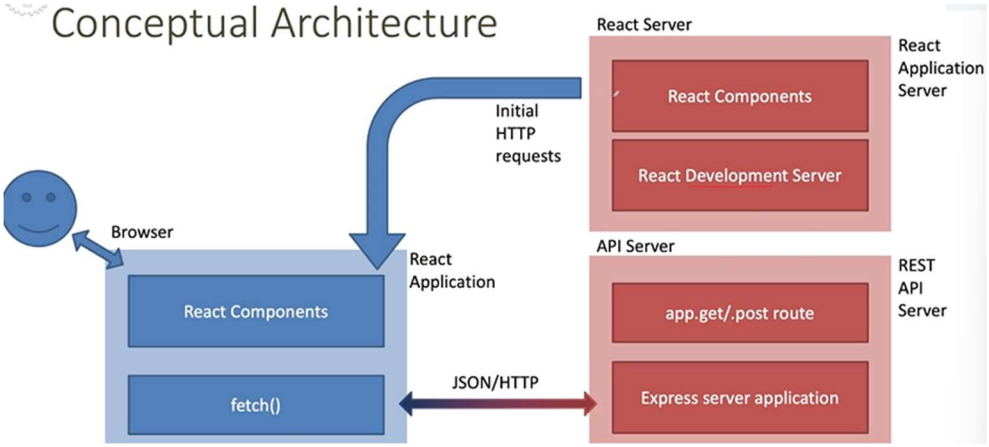


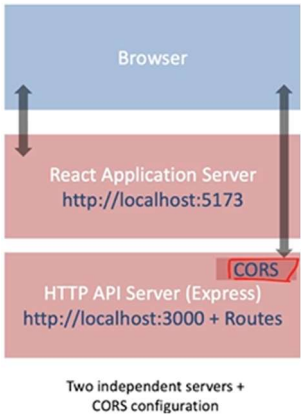

La fetch viene eseguita nel browser dell'app react. Noi vogliamo che il browser, attraverso la fetch, faccia chiamate http con il server Express sulla porta 3000 e non sul React che gira su 5173.

Nota: la fetch, di default, effettua chiamate http allo stesso dominio in cui il browser sta girando (5173).

Per abilitare la fetch ad effettuare richieste ad altri server:

- Mantenendo i due server **abilitando**, sul server express (3000), **CORS** (possibilità di ricevere chiamate cross-origin)
- *Fare un build dell'applicazione react e farla hostare direttamente dal server express, in questo modo la fetch andrà di default sul localhost 3000 perché sarà lui ad ospitare*.

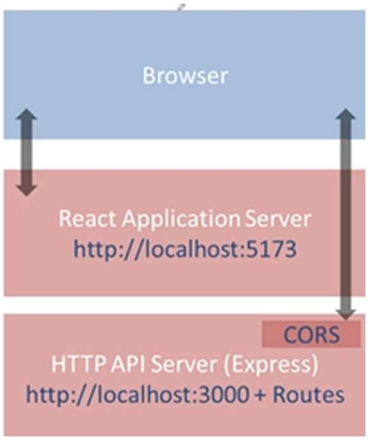



**Considerazioni**:

- Deployment
    - Limitazioni di sicurezza a causa del cross-origin
- Vantaggi:
    - Avere due server, separa i carichi di lavoro
    - Si può usare un qualsiasi tipo di API server

## Come impostare il set-up sul doppio server:




Noi abbiamo un server react di sviluppo e un server express API che hanno un host diverso/diverse porte.
Il browser riceve l'applicazione react dal primo server e invia le richieste di aggiornamento dei dati dal secondo server.

- Le chiamate arrivaranno da 5173 ma saranno dirette a 3001.

**In express:**

```
// index.js (node express server)
var cors = require('cors') ;  // npm install cors

//Enable All CORS Requests (for this server)
app.use(cors());
//Use ONLY for development, otherwise restrict domain
```

Nota: in produzione non bisogna permettere richieste CORS da ogni origine ma va specificata l'origin (non come abbiamo fatto qui… andrebbe ristretto a localhost 5173)

**API.js in the React Application**

```
const APIURL=new URL('http://localhost:3000');

async function getCourses() {
  return fetch(new URL('/courses', APIURL))
    .then((response)=>{
      if(response.ok) {
        return response.json() ;
      } else {
        throw response.statusText;
      }
    })
    .catch((error)=>{
      throw error;
    });
}
```

Called in useEffect()

**index.js for the API Server**

```
const express = require('express');
const port = 3000;
const cors = require('cors');
const app = express();
app.use(cors());

app.get('/courses', (req, res) => {
  dao.listCourses()
    .then((courses) => res.json(courses))
    .catch((dbErrorObj)=>
      res.status(503)
         .json(dbErrorObj));
});

app.listen(port, () => console.log(`Example app
listening at http://localhost:${port}`));
```

Calls DAO.js

--ultimi 17 minuti, codice Exercise 12