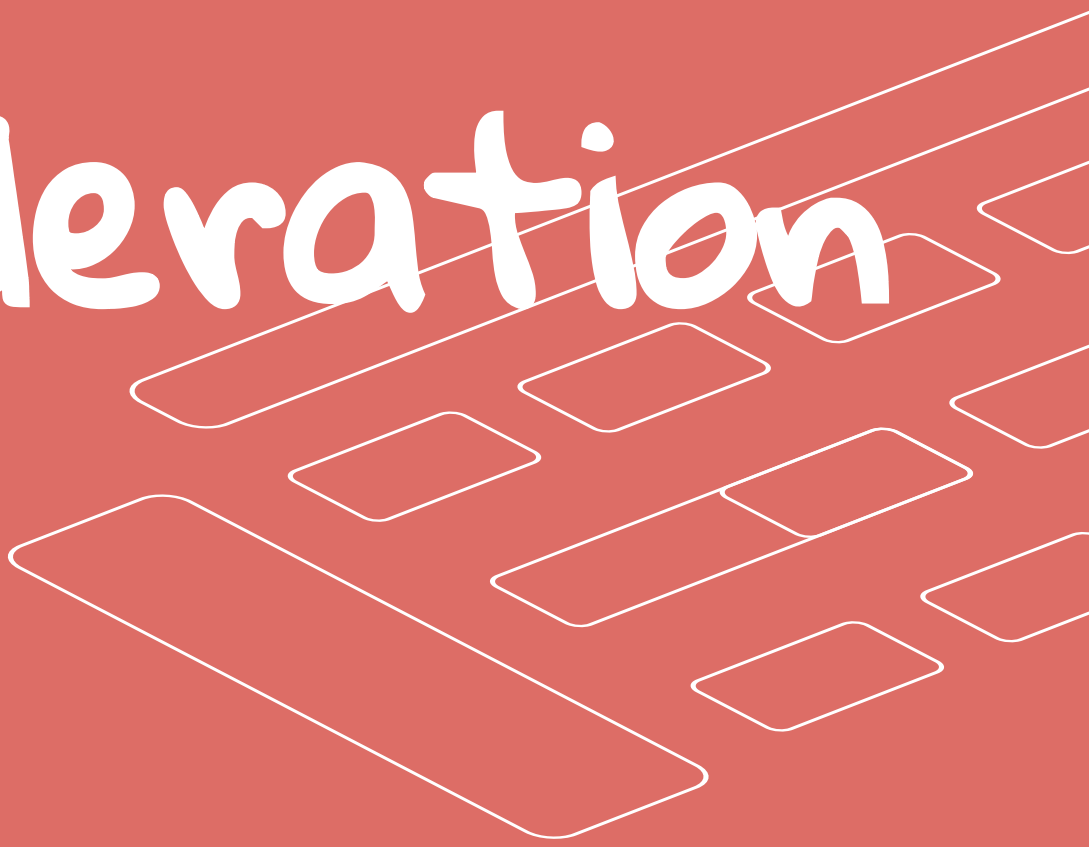
safouen.com

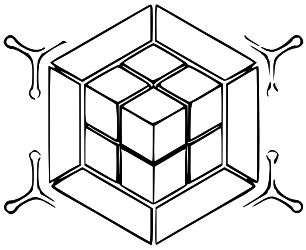
Module Federation



notes available on safouen.com

#2

what the hell is Module Federation?



Module federation is a way to :

Share code dynamically between frontend applications

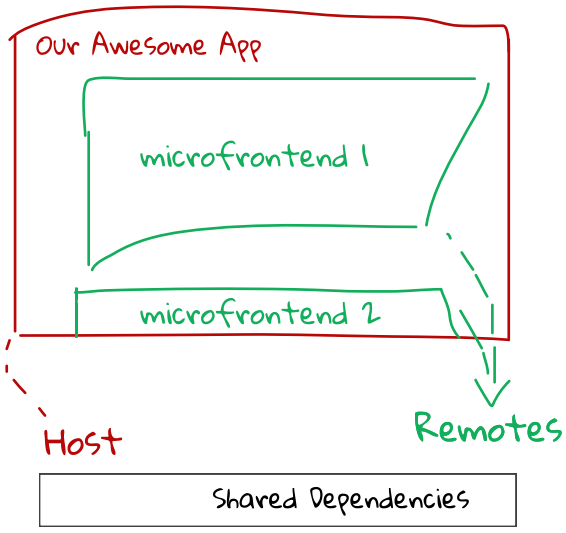
Microfrontends

Similar to microservices on the server-side

But , Why???

- ✓ **Code Sharing** : Share libraries and components across applications without needing a shared repository.
- ✓ **Independent deployments** : Teams can deploy their applications independently

Concepts



teams can work on separate, independently deployable frontend pieces called a Remote which can be integrated into a cohesive application called a Host

remotes and host applications can have shared dependencies between host and remote applications

Getting Started

In your webpack.config.js of the app exposing modules:

Personal Recommondations

- use the same React version
- use Rsbuild as compiler for both sides
- add UID to your remote entry file name to insure synchronisation

```
const { ModuleFederationPlugin } =
require("webpack").container;
plugins: [
 new ModuleFederationPlugin({
  name: "remoteApp",
  filename: "remoteEntry.js",
  exposes: {
   "./CoolComponent": "./src/CoolComponent",
  },
  shared: ["react", "react-dom"]
 })
]
```

remote file to generate

Remote

The one exposing component or app

File or files to export

you can export your entire index file

```
new ModuleFederationPlugin({
 name: "hostApp",
 remotes: {
  remoteApp: "remoteApp@http://localhost:3001/
remoteEntry.js",
 },
 shared: ["react", "react-dom"]
})
```

host

the one that hosts remotes

Served instance of the remote

Both remote and host should be served with different ports

shared packages and libraries

Now just import and Voilaa !

```
const CoolComponent = React.lazy(() => import("remoteApp/CoolComponent"));
```

But should you even do Microfrontends ?