🚀 Dynamic Backend URL Management with useConfig (Arbotix)

# 📌 What is this system?

- Dynamically fetches backend URLs (like Arbotix) from your Config Server.  
- Removes hardcoded URLs from your frontend.  
- Allows backend URL updates without redeploying frontends.  
- Works across all your frontend projects with consistency and scalability.

# 🗂️ Folder Structure

/src  
 /hooks  
 useConfig.js – Check this file in this repo and add in your code  
 /pages  
 login.js  
.env

# 1️⃣ Setting Up Your Config Server

Your Config Server should:  
- Return JSON containing your services.  
- Example expected response:  
  
[  
 {  
 "services": {  
 "arbotix": "http://localhost:5002"  
 }  
 }  
]  
  
This allows your frontend to dynamically extract:  
{ "arbotix": "http://localhost:5002" }

# 2️⃣ Generating and Managing API Key

- Generate a API Key from dashboard   
- Save it in your Config Server environment as CONFIG\_API\_KEY.  
- Share securely with frontend teams.

# 3️⃣ Configuring .env in Frontend

REACT\_APP\_APP\_NAME=arbotix  
REACT\_APP\_ENVIRONMENT=production  
REACT\_APP\_CONFIG\_API=http://localhost:5000/api/configs  
REACT\_APP\_CONFIG\_API\_KEY=<your-api-key>  
  
✅ Replace <your-api-key> with your generated API key.  
✅ Restart your frontend server after editing .env.

# 4️⃣ Using useConfig in Project

import { useConfig } from '../hooks/useConfig';  
  
const { config, error, loading } = useConfig();  
const serviceUrl = config.arbotix;  
  
await axios.post(`${serviceUrl}/login`, { username, password });

# 🎯 Benefits

- Centralized dynamic backend URL management.  
- Supports multiple environments.  
- Removes hardcoded backend URLs.  
- Allows updates without redeployment.  
- Secured with API key validation.

# ✅ Summary

Your dynamic backend URL management system with useConfig and Arbotix:  
- Enables seamless backend updates.  
- Reduces maintenance overhead.  
- Keeps your projects scalable and clean.