Step 4: Building a Modern Dashboard with React

Tutor: Haythem Ghazouani

1 Introduction

A Machine Learning model is only useful if non-technical users can interact with it. In this tutorial, we build a modern, responsive web dashboard using **React** and **Vite**.

Technical Tip

The frontend code is located in the `frontend/` directory.

2 Project Setup

We use **Vite** for a fast development experience.

```
# Initialize the project
npm create vite@latest frontend --template react
cd frontend
npm install
npm install axios lucide-react
```

3 Component-Based Architecture

React allows us to break the UI into reusable pieces. In our dashboard:

1. **ChurnForm:** Handles input collection for a single customer.
2. **BatchUpload:** Manages CSV file selection and upload.
3. **ResultsDisplay:** Visualizes the risk percentage using CSS transitions.

4 Connecting to the Backend (Axios)

To talk to our FastAPI server, we use **Axios**.

React Concept: Handling CORS

Browsers block requests between different ports (8000 and 5173) by default. Ensure your FastAPI app has the `CORSMiddleware` enabled to allow requests from the frontend.

```
const handlePredict = async (data) => {
  const response = await axios.post('http://localhost:8000/predict', data);
  setPrediction(response.data);
};
```

5 Style Aesthetic

We use **Vanilla CSS** with modern features like `backdrop-filter` for glassmorphism effects and `radial-gradient` for deep background textures. This ensures a premium "Dark Mode" feel without the bloat of heavy UI libraries.

6 Running the Application

Start the development server:

```
npm run dev
```

Open the provided URL (usually `http://localhost:5173`).

7 Exercise

1. **UI Challenge:** Add a "Loading" spinner using `lucide-react` while the API call is in progress.
2. **Validation:** Use JavaScript to ensure the "Age" input is between 18 and 100 before sending the request.
3. **Graphing:** Use a library like `Chart.js` or `Recharts` to plot the churn risk of the last 10 predictions.