Team Diluted

OptiCO$_2$

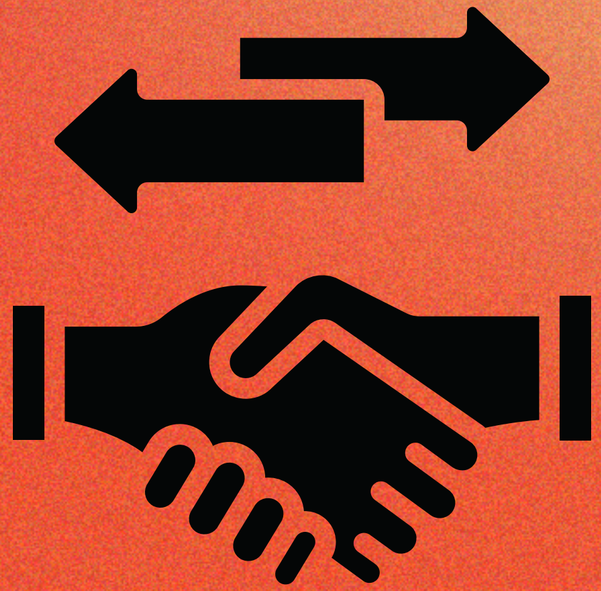
James Weng, Ryan Li, Edwin Lin

Put yourselves in the shoes of a steel company..

Microcosm of considerations

- supply-chain risk
- tariffs
- price volatility
- transport
- **carbon footprint**

How can we critically measure carbon emissions as a byproduct of steel transport and production?



It becomes a matter of understanding tradeoffs.

How we can cater the specific tradeoff between steel cost and carbon footprint to a company’s fit, mission statements, and finances is precisely what we are trying to address with OptiCO2.

7%

of the world’s carbon emissions is attributed to the production of steel

There needs to be an easier entry point...

Main consideration: country of origin and carbon footprint/cost of transport

Carbon-cost Tradeoff

Real world application

Project Outline

Data We Need

Freight Cost ($)/ Ton
Production, Land, Sea, Taxes

Carbon emissions per route
Production, Transport

Footprint / Country

Company Choices / Route

Getting the data

WorldSteel (production data)
ClimateTrace (emissions CSV)
SteelBenchmarker (country pricing)

worldsteel ASSOCIATION

CLIMATE TRACE

SteelBenchmarker™

Implementation

```python
from fastapi import FastAPI, HTTPException
import pandas as pd
from pydantic import BaseModel
import math

app = FastAPI()

# CO2 emission and steel quantity by country
countries = ["CHN", "IND", "JPN", "RUS", "TUR", "BRA", "ITA", "DEU", "GBR"]
raw = pd.read_csv("regional_steel_emissions.csv")[["iso3_country", "emissions_quantity"]]
processed = raw.loc[raw["iso3_country"].isin(countries)].groupby("iso3_country").mean() / 1e6
processed["steel_quantity"] = [1005.1, 149.4, 84.0, 71.0, 36.9, 33.8, 20.0, 37.2, 4.0]

# origin DataFrame
def co2perton(country):
    return processed.loc[country, "emissions_quantity"] / processed.loc[country, "steel_quantity"]
origin = {…
origin_df = pd.DataFrame.from_dict(origin,
                                   orient="index",
                                   columns=["Companies", "Origin_port", "Carbon", "Costs"])

# destination DataFrame
dest = { …
dest_df = pd.DataFrame.from_dict(dest,
                                 orient="index",
                                 columns=["Dest_port", "Land_distance"])
```

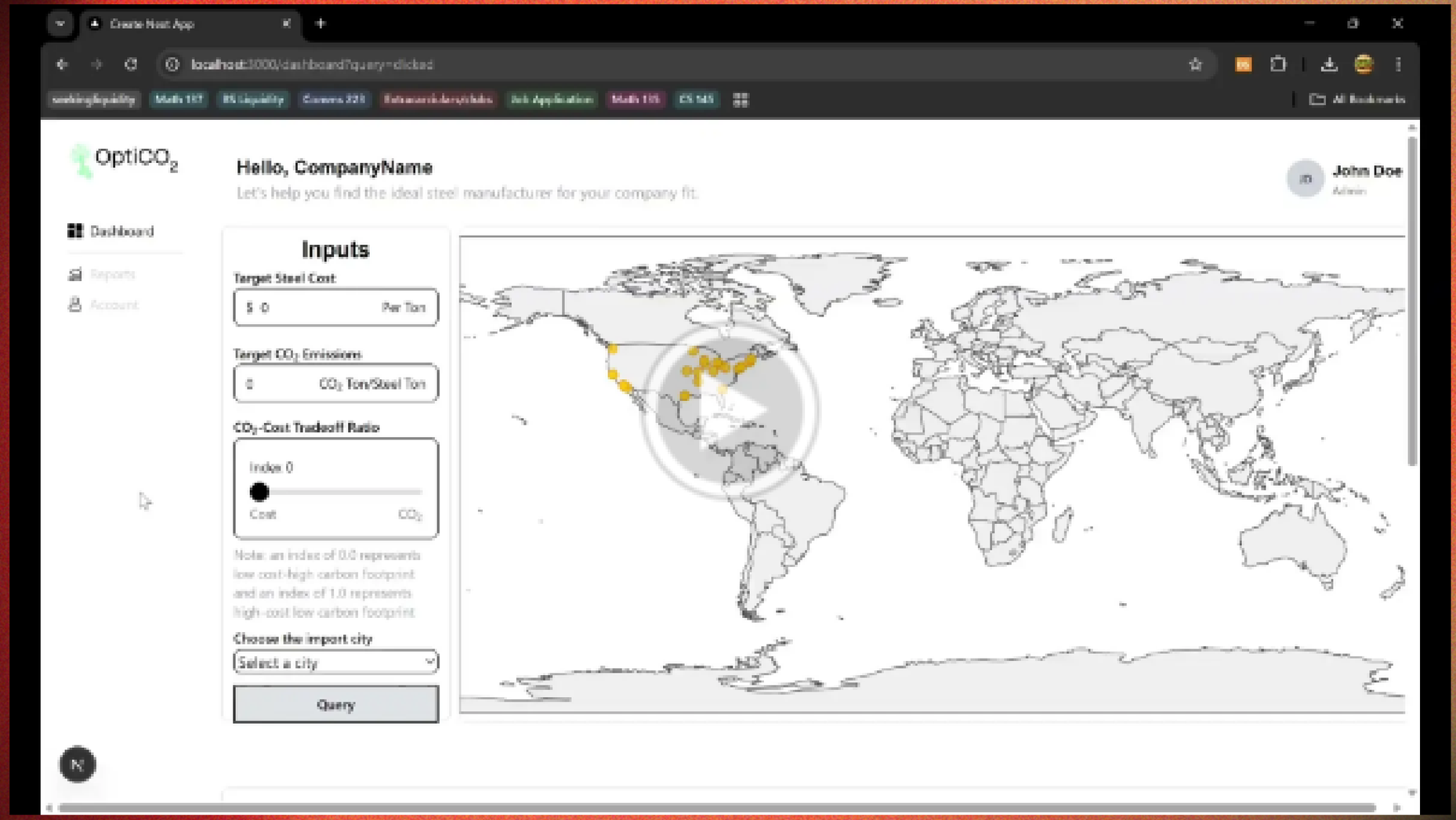
Python (pandas, FastAPI)

next.js (plotly, chartly, framer-motion, etc)

```tsx
import { Pie } from "react-chartjs-2";
import { Chart as ChartJS, ArcElement, Tooltip, Legend } from "chart

// Register the required elements
ChartJS.register(ArcElement, Tooltip, Legend);

const SourcesofCarbon = ({ data }) => {
  const carbon = data?.best_country?.Carbon;

  const carbonData = {
    labels: ["Production Emissions", "Sea Emissions", "Land Emissio
    datasets: [
      {
        data: carbon
          ? [
              carbon[0] || 0, // Production with fallback to 0
              carbon[1] || 0, // Sea with fallback to 0
              carbon[2] || 0, // Land with fallback to 0
            ]
          : [1, 1, 1], // Default data if carbon is undefined
        backgroundColor: ["#c6ac8f", "#4c5c68 ", "#46494c"],
      },
    ],
  };
  return (
    <div className="p-2">
      <Pie data={carbonData} />
      <h1 className="text-center font-bold mt-2">
        Total Carbon Emissions:  {""}
        {data?.best_country?.Total_carbon?.toFixed(2) || "N/A"}
         
          tCO<sub>2</sub> / Ton
        </h1>
```

```tsx
import { useEffect, useState } from "react";
import * as d3 from "d3";
import dynamic from "next/dynamic";
import { getDisplayName } from "next/dist/shared/lib/utils";

const Plot = dynamic(() => import("react-plotly.js"), {
  ssr: false,
  loading: () => <div>Loading map...</div>,
});

interface PrecipMapProps {
  queryResult?: number[]; // optional, initially undefined
}

export default function PrecipMap({ queryResult }: PrecipMapProps) {
  const [data, setData] = useState<any[] | null>(null);

  useEffect(() => {
    // Load the CSV first
    d3.csv("/dummydata.csv").then((rows) => {
      const csvTrace = {
        type: "scattergeo",
        mode: "markers",
        lat: rows.map((r) => Number(r.Lat)),
        lon: rows.map((r) => Number(r.Lon)),
        text: rows.map(
          (r) => `${r.City}<br>Cost: $${r.SteelCost}<br>CO2: ${r.CO2Emission}`
        ),
        marker: {
          color: "rgba(255, 199, 0, 0.8)",
          size: 10,
          line: { color: "rgba(255, 135, 0, 0.8)", width: 1 },
        },
      };

      const traces = [csvTrace];
```

Demo


Create Next App
localhost:3000/dashboard?query=clicked
OptiCO2
Hello, CompanyName
Let's help you find the ideal steel manufacturer for your company fit.
John Doe
Admin
Dashboard
Reports
Account
Inputs
Target Steel Cost
$ 0
Per Ton
Target CO2 Emissions
0
CO2 Ton/Steel Ton
CO2-Cost Tradeoff Ratio
Index 0
Cost
CO2
Note: an index of 0.0 represents low cost-high carbon footprint and an index of 1.0 represents high-cost low carbon footprint
Choose the import city
Select a city
Query

What could this mean for the world?

In just the US alone..

- 6 billion dollars invested
- High Steel Price Volatility
- 26.2 Million Metric Tons of Steel Transported in 2024
- Billions in Tariffs

In the **ideal world**, all companies would place the greatest emphasis on a low carbon footprint, but that is not realistic.

Potential for expansion

Currently US centric

Incorporate inputs such as time, steel category

Consider a more nuanced view of the supply chain, incorporating geopolitical sentiment analysis

The framework is readily adaptable to aid companies track sustainable trade-offs outside of just steel

Team Diluted

James Weng

Edwin Lin

Ryan Li