You are a full-stack AI engineer helping me build a scalable system for calculating shadows cast by buildings using solar data. I already have a working mathematical model (in JavaScript) that:

- Calculates the sun's direction vector using azimuth + elevation
- Generates a 3D shadow volume from a building mesh
- Cuts that volume against nearby buildings to detect affected area/volume

I want you to turn this into a production-ready system with the following:

🛠️ TASKS

✅ 1. Convert Shadow Logic to Backend API

- Migrate JS logic to Python (FastAPI) or Node.js (Express).
- Expose an endpoint like:

```
POST /api/shadow/calculate
{
  "buildings": [...],       // 3D geometry or bounding boxes
  "location": [lat, lng],
  "timestamp": "2025-06-21T12:00:00Z"
}
```

Output:

- Shadow volumes
- Affected building IDs
- Surface area and volume affected
- Optional: shadow geometry (for visualization)

✅ 2. Use pvlib or similar for sun position

- Integrate pvlib to get accurate solar azimuth and elevation.

✅ 3. Optimize for Performance

- Use low-poly building approximations (e.g. extruded footprints).
- Allow 1,000+ buildings per request using:
- multiprocessing or Dask (Python)
- Web workers or clustered server threads (Node.js)

✅ 4. Storage

- Store input and output in PostGIS or MongoDB.
- Enable spatial queries like:

Get all buildings affected between 10 AM and 2 PM on June 21

5. Build a Frontend UI

- Tech stack: React + Three.js (or Next.js)
- Features:

- Upload building data (GeoJSON or STL)
- Select time, date, and location
- View buildings, shadows, and affected zones in 3D
- Download results (JSON or map screenshot)

✅ 6. Deployment

- Containerize with Docker
- Deploy on:
- Render.com (simple)
- AWS (Lambda/ECS) or Google Cloud Run
- Add caching for repeated queries

✅ 7. Documentation

- Auto-generate OpenAPI/Swagger docs for the backend.
- Add README.md and setup scripts for easy installation.

💡 EXTRAS (Optional but Useful)

- Support time series (e.g. hourly shadow simulation for a full day)
- Add solar exposure percentage per building
- Support importing OSM data via osmnx

✅ INPUT MATERIALS

You can assume:

- JS logic is complete and includes sun vector → shadow extrusion → boolean cut.
- Building geometries will be simplified boxes or polygons with height.