AI-Powered Rooftop Solar Analysis – Project Documentation

Overview

This project builds an AI-powered rooftop solar feasibility analysis system that takes satellite images as input and produces a detailed report on usable area, potential energy output, ROI, and more. It integrates cutting-edge computer vision and language models to enable end-to-end analysis and recommendation generation.

High-Level Workflow

STEP 1: Satellite Image Acquisition

Goal: Obtain high-quality rooftop imagery.
Sources:
- Google Earth Engine (GEE) for programmatic access.
- Manual Upload for user input.

Output: RGB image (JPEG/PNG) for computer vision models.

STEP 2: Rooftop Segmentation

Goal: Extract rooftop area using deep learning.
Tool: Segment Anything Model (SAM) by Meta.
Process:
- Load RGB rooftop image.
- Apply SAM to generate rooftop segmentation mask.
- Convert mask to polygon if needed.

Output: Binary mask or rooftop polygon.

STEP 3: Obstacle Detection

Goal: Identify unusable rooftop regions.
Tool: YOLOv8 object detection model.
Process:
- Detect obstacles (AC units, water tanks, etc.).
- Mask out obstacle regions from rooftop mask.

Output: Usable rooftop mask (excludes obstacles).

STEP 4: Scene Understanding & Validation

Goal: Confirm image depicts a rooftop and assess environmental factors.
Tools:
- CLIP for image-text similarity.
- GPT-4V (optional) for advanced vision analysis.

Output: Boolean rooftop validation + quality summary (e.g., shade, obstructions).

STEP 5: Solar Potential Estimation

Goal: Calculate installable solar capacity and financial estimates.
Assumptions:
- 1 kW ≈ 6.5–7 m^2
- Output: 1200 kWh/kW/year
- Cost: ₹70,000/kW
- Energy price: ₹8/kWh

Process:
- Convert pixels → square meters
- Estimate installable kW
- Compute:
  - Installation cost
  - Annual savings
  - Payback period

Output: kW, ₹ cost, savings/year, ROI

STEP 6: Summary Generation

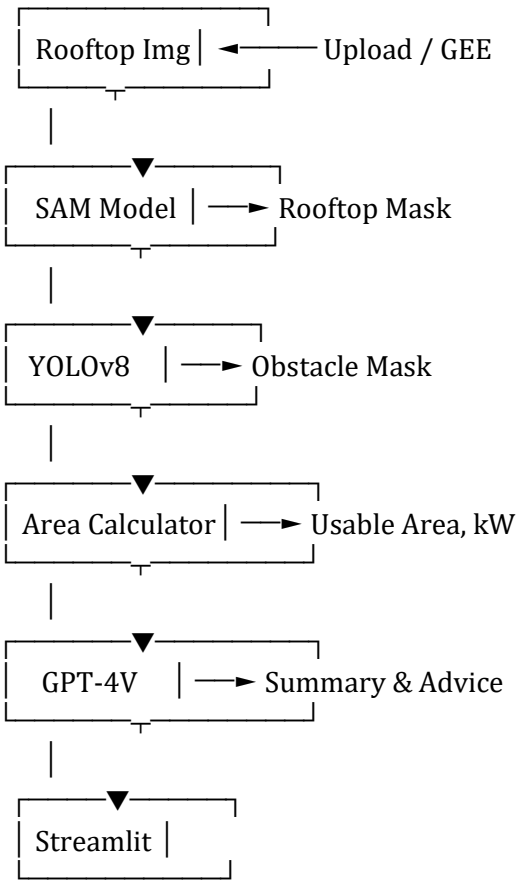
Goal: Generate a human-readable report.
Tool: GPT-4 or OpenRouter LLM
Prompt Example:
“Summarize solar feasibility for this rooftop including potential capacity, ROI, and any concerns.”

Output: Natural-language summary report.

STEP 7: Web Interface

Goal: Provide an intuitive user experience.
Tool: Streamlit
Features:
- Upload/fetch image
- Visualize segmentation + obstacles
- Display statistics and summary
- Export PDF report (optional)

Architecture Diagram (Text-Based)



Project Setup Instructions

1. Clone Repo / Extract Files
2. Set up Environment:

```
python -m venv solar_env
source solar_env/bin/activate  # Windows: solar_env\Scripts\activate
pip install -r requirements.txt
```

3. Download Model Weights:
  - SAM: sam_vit_b_01ec64.pth
  - YOLOv8n: Auto-downloaded via ultralytics

4. Run App:
  streamlit run solar_analysis_app.py

Example Use Case

Input: Satellite image of 500x500 pixels
Output:
- Rooftop area: 45 m^2
- Installable: 6.5 kW
- Cost: ₹4.55 Lakhs
- Annual savings: ₹62,400
- Payback: ~7.3 years
- Summary: "This rooftop is highly suitable for solar installation..."

Future Improvements

- Add 3D elevation analysis (LiDAR or DEM)
- Integrate weather and solar irradiance data
- Suggest panel layout optimization (vertical/horizontal tilt)
- Add multilingual report generation
- Deploy via Hugging Face Spaces or Docker container

Deliverables

- ✅ Complete Codebase
- ✅ Streamlit Web App
- ✅ README with setup instructions
- ✅ Sample Summary Reports
- ✅ Requirements.txt
- ✅ Architecture Flowchart
- ✅ Example Use Cases