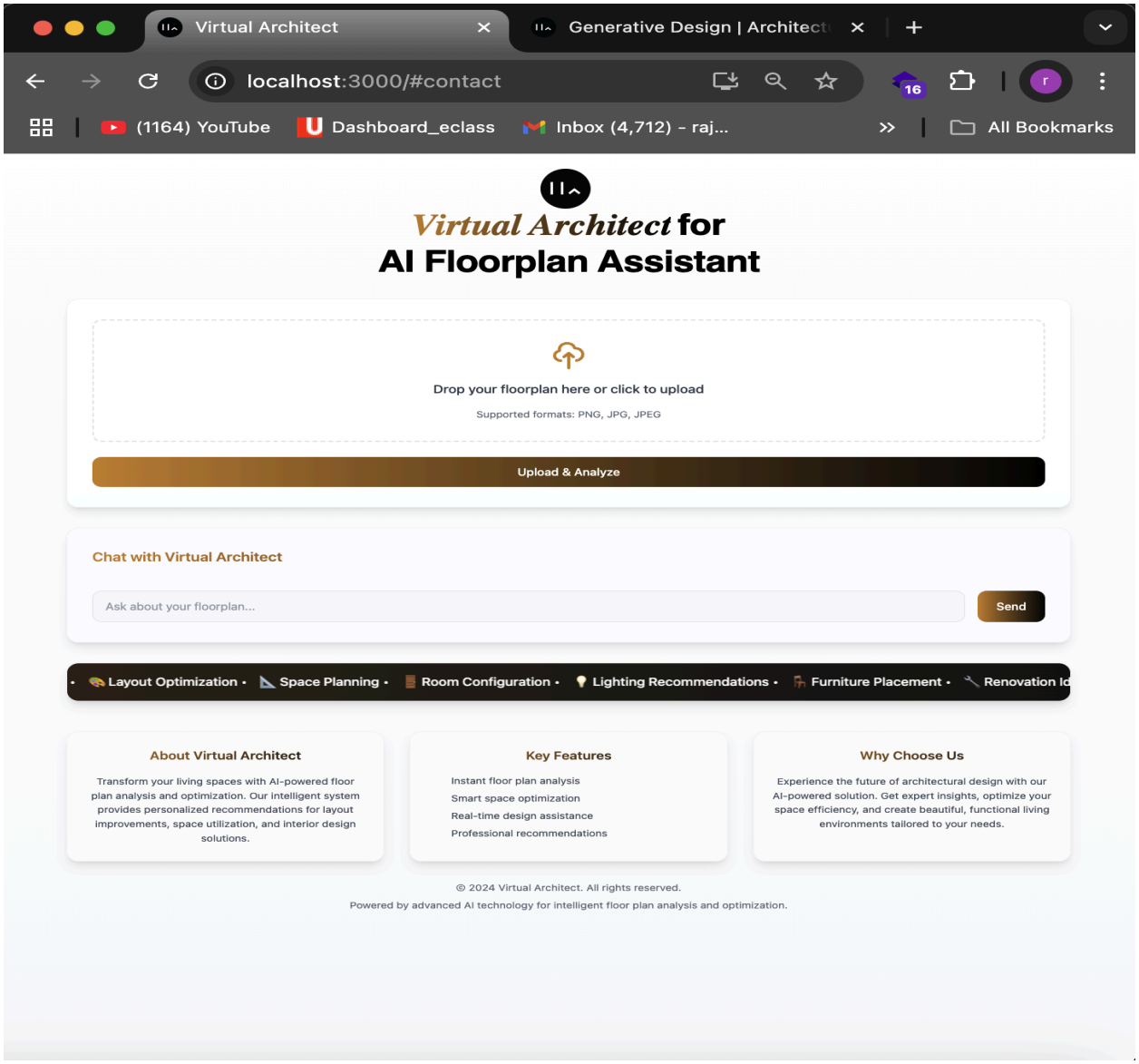
Virtual Architect: AI-Powered Floorplan Analysis & Recommendations

---

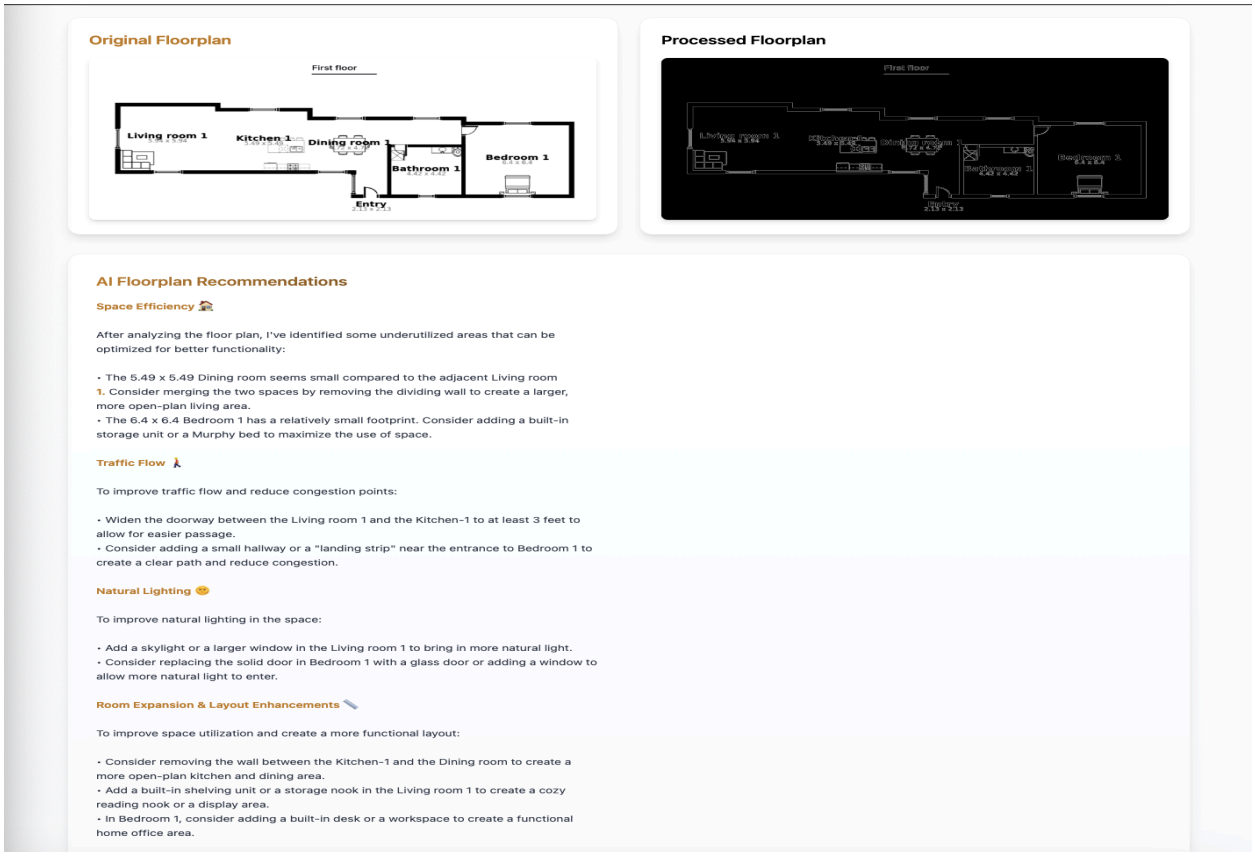
1. Introduction

The **Virtual Architect** is an AI-powered application designed to analyze floorplans, extract architectural features, and provide intelligent recommendations. Users can interact with the AI via a **chat window** or **voice mode**, making it easier to optimize spaces and improve their architectural designs.

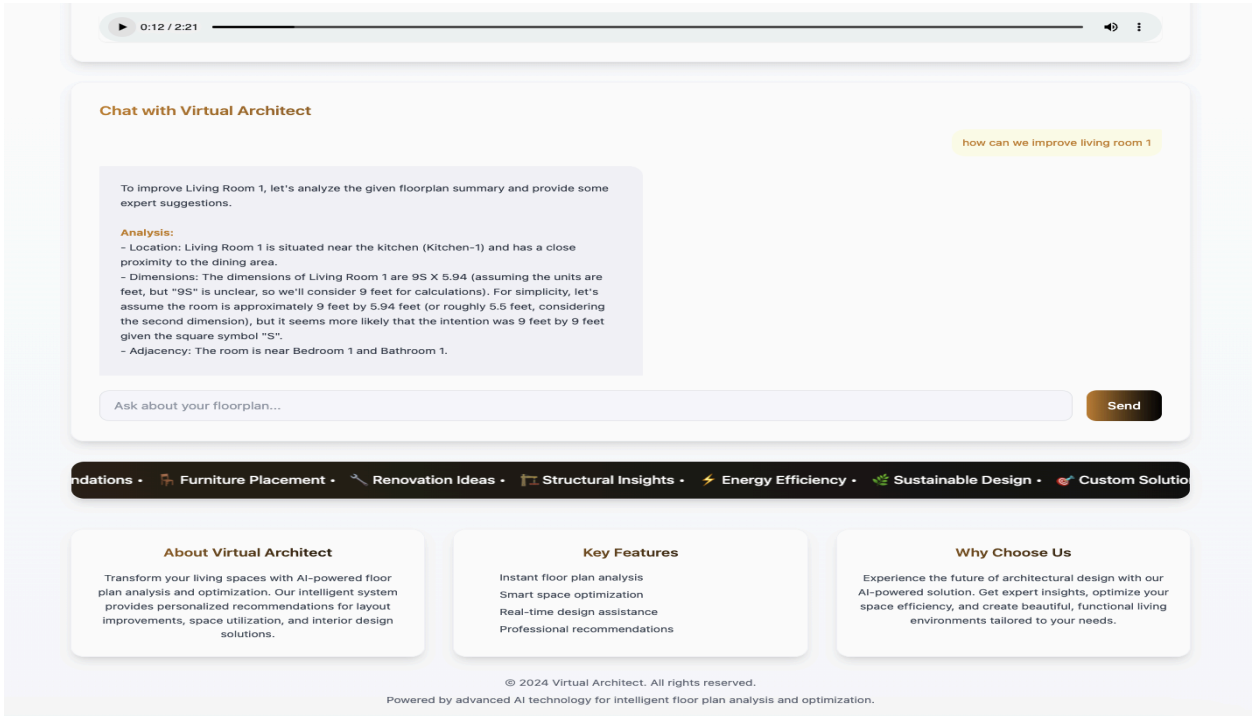
This project was developed as part of an assessment by **MakeIt.ai**, leveraging AI and computer vision to automate floor plan analysis.



(IMG1 - Final Iteration for Virtual Architect using FASTAPI and python for backend , React and Tailwind by Cursor for Custom Frontend which can be anytime Customized)

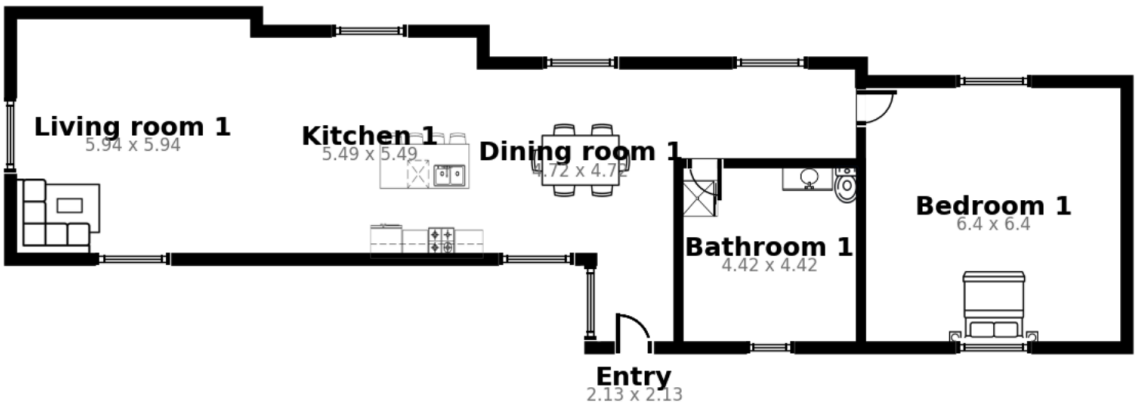


(IMG2 - Original Floor Plan with GrayCode Floor Plan used for extracting features to get AI Recommendations)

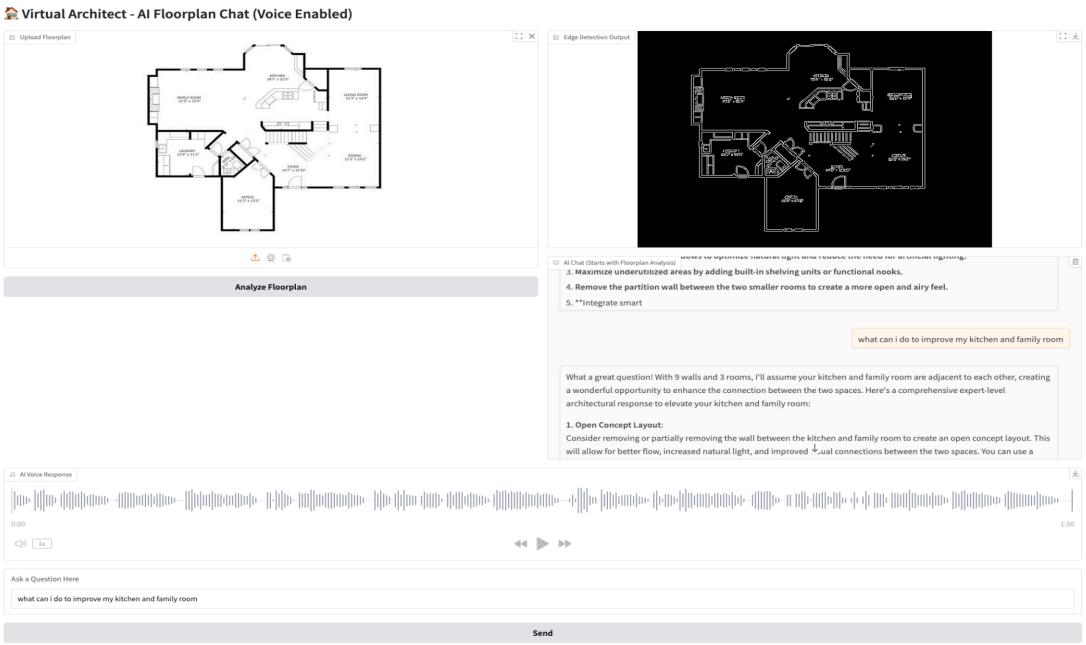


(IMG3 - Optional Voice Mode with Interactive AI Chat and dynamic sliders for Modern UI)

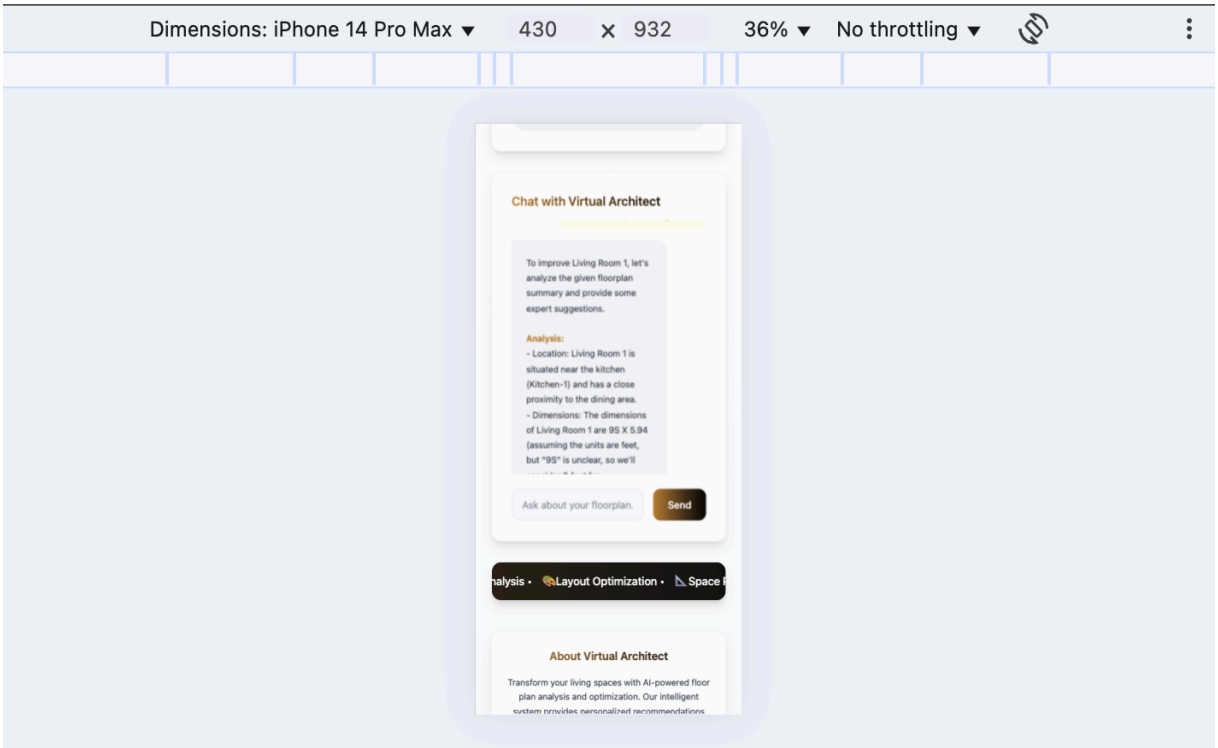
First floor



(IMG4 - Floor Plan Schematic used from makeit.ai)



(IMG5 - Deployed Virtual Architect Version for Feature Checking and Working https://huggingface.co/spaces/rajat1343/VirtualArchAI)



(IMG6 Responsive ,User Friendly UI that works seamlessly across devices - Virtual Architect)

Key Differences: Gradio vs. FastAPI + React

Feature	**Gradio Final Iteration**	**Cursor (FASTAPI + React)**
UI	Basic , Limited , User-Friendly	Fully Interactive . Dynamic , Customable
Deployment	Easy and Free But limited Control	Scalable, Flexible Hosting but Expensive
AI	Groq API for Responses	More Structured AI Prompts
Image Processing	OpenCV , Pytesseract	Same with better Optimization
Scalability	Limited Compute Power	Scalable Backend and Frontend
Performance	Dependent on Hugging face server limits	More control over API response time and handling

---

2. Features & Functionality

🏠 Upload & Analyze Floorplans

- Users upload a **floorplan image**.
- The AI detects walls, rooms, and structural features.
- Provides **automated recommendations** based on spatial analysis.

💬 AI Chat Window

- Users can **ask questions** about their floorplan.
- AI responds with **architectural suggestions**.
- Continuous chat history enables better interaction.

🔊 Voice Mode

- AI responses are converted into **speech**.(Download and Playback Speed Available)
- Users receive **verbal recommendations** for ease of understanding.

📊 AI-Powered Analysis

- **Identifies walls, room sizes, and inefficiencies**.
- **Suggests optimal lighting, space utilization, and design improvements**.
- Works on **2D architectural floor plans**.

---

3. Technologies & Tools Used

🛠️ Tech Stack:

- **Frontend**: React, TailwindCSS , Gradio , Cursor
- **Backend**: FastAPI (Python) ,
- **Database**: N/A (real-time API processing) (FASTAPI)
- **Environment:** venv with groq api for LLM.
- **Deployment**:
    - Frontend: Netlify
    - Backend: Railway (Future) , Gradio (for final iteration 2) , LocalHost:8000 (Present)

🤖 AI & Machine Learning:

- **LLM Model**: LLaMA-3 (used for generating AI-driven recommendations)
- **OCR (Optical Character Recognition)**: Extracts text from floorplans using **Tesseract OCR**.
- **Computer Vision**:
    - **OpenCV** for edge detection.
    - **Hough Line Transform** to detect walls and structures.
- **Roboflow**: Used for pre-processing and training floorplan detection models.
- **Cursor**: Low Code Tool used for Front End

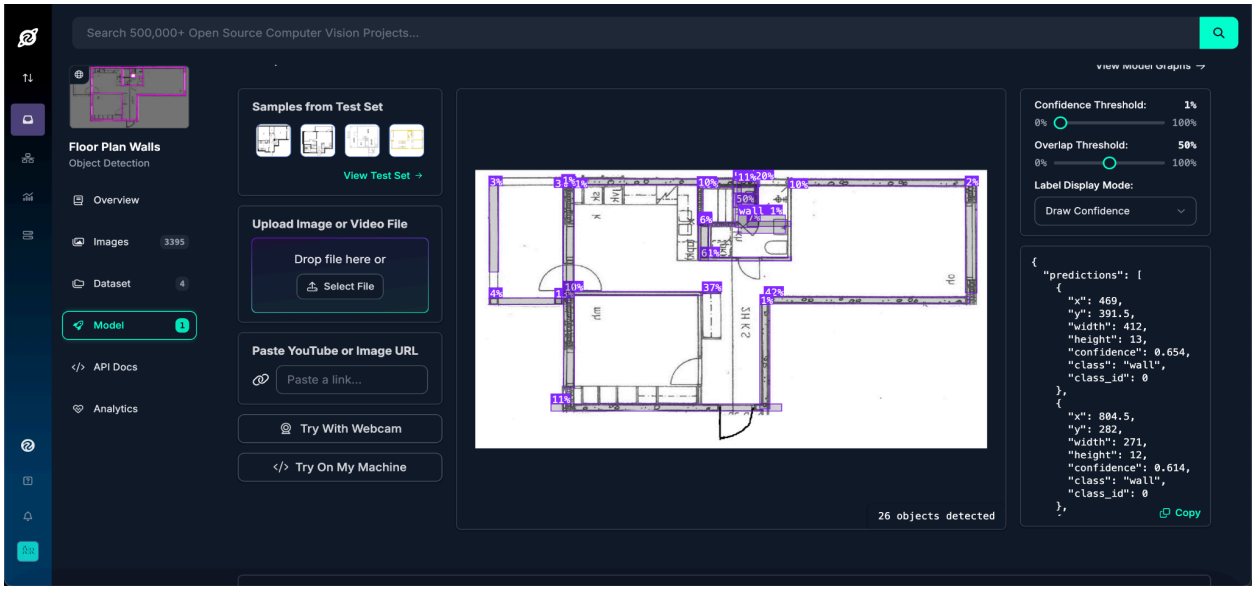
🎤 Speech & Audio Processing

- **gTTS (Google Text-to-Speech)** for converting AI responses into audio.

---

4. How the AI Recommendation System Works

1. **Image Processing**:
    - The user uploads a floorplan image.
    - The system **converts the image to grayscale** and applies **edge detection**.
    - **Hough Line Transform** is used to detect walls and structural elements.
2. **Feature Extraction**:
    - The AI extracts **rooms, walls, doors, and furniture placements**.
    - **OCR scans any embedded text** (e.g., room labels, dimensions).
3. **Data Sent to LLM**:
    - The extracted architectural data is **formatted into structured text**.
    - A **specialized AI prompt** is crafted, providing context for space optimization.
    - The **LLaMA-3 model** generates **tailored architectural recommendations**.
4. **AI Response Generation**:
    - The AI outputs insights such as:
        - **Traffic flow optimization**
        - **Natural lighting improvements**
        - **Room expansion suggestions**
        - **Furniture placement guidance**
    - The response is displayed in **text form** and **converted to speech**.



(IMG7 Roboflow Floor wall Detection AI Model based on YoloV8 for reference which could be trained on makeit.ai floor plan data sets for better detection of objects, walls and segmentations for future integrations)

---

6. Conclusion

The Virtual Architect successfully integrates **AI-powered analysis, real-time chat, and voice recommendations** to assist users in optimizing their floorplans. By leveraging **computer vision, LLM models, and OCR**, the system provides an intuitive way to understand and improve architectural designs.

---

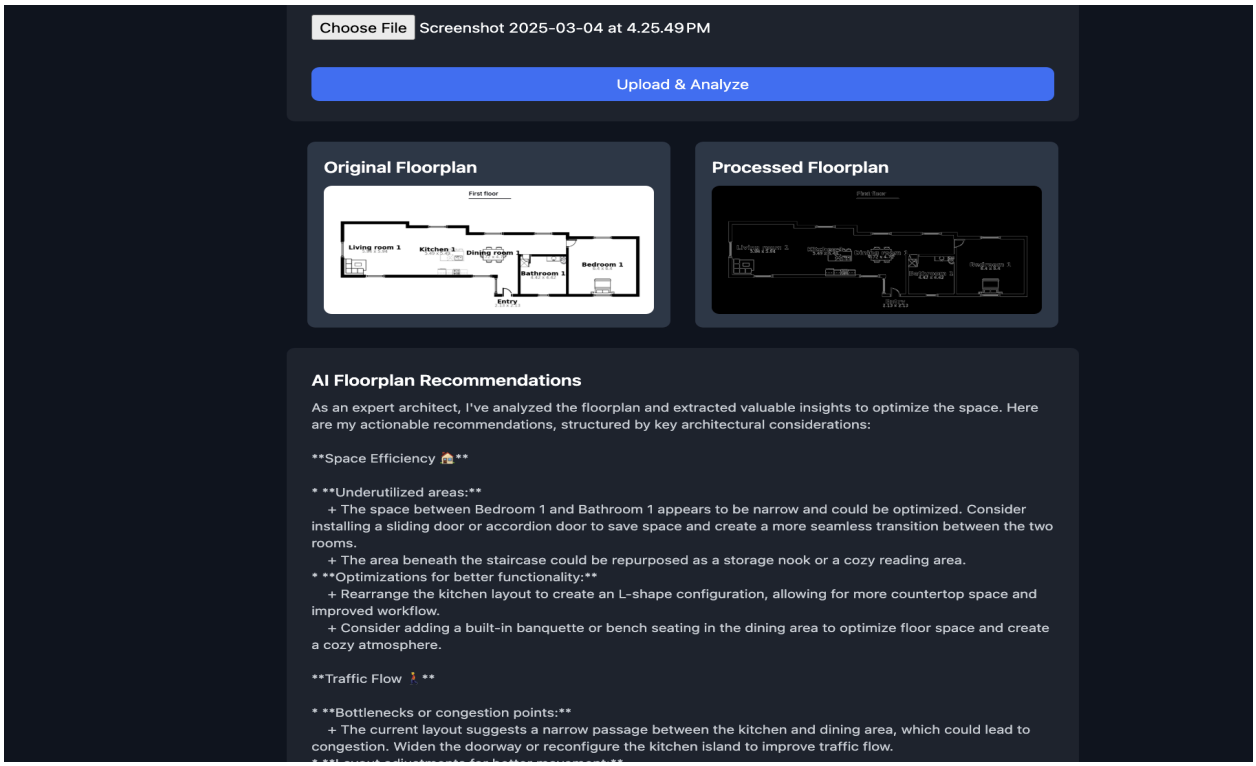
7. Improvements & Optimizations

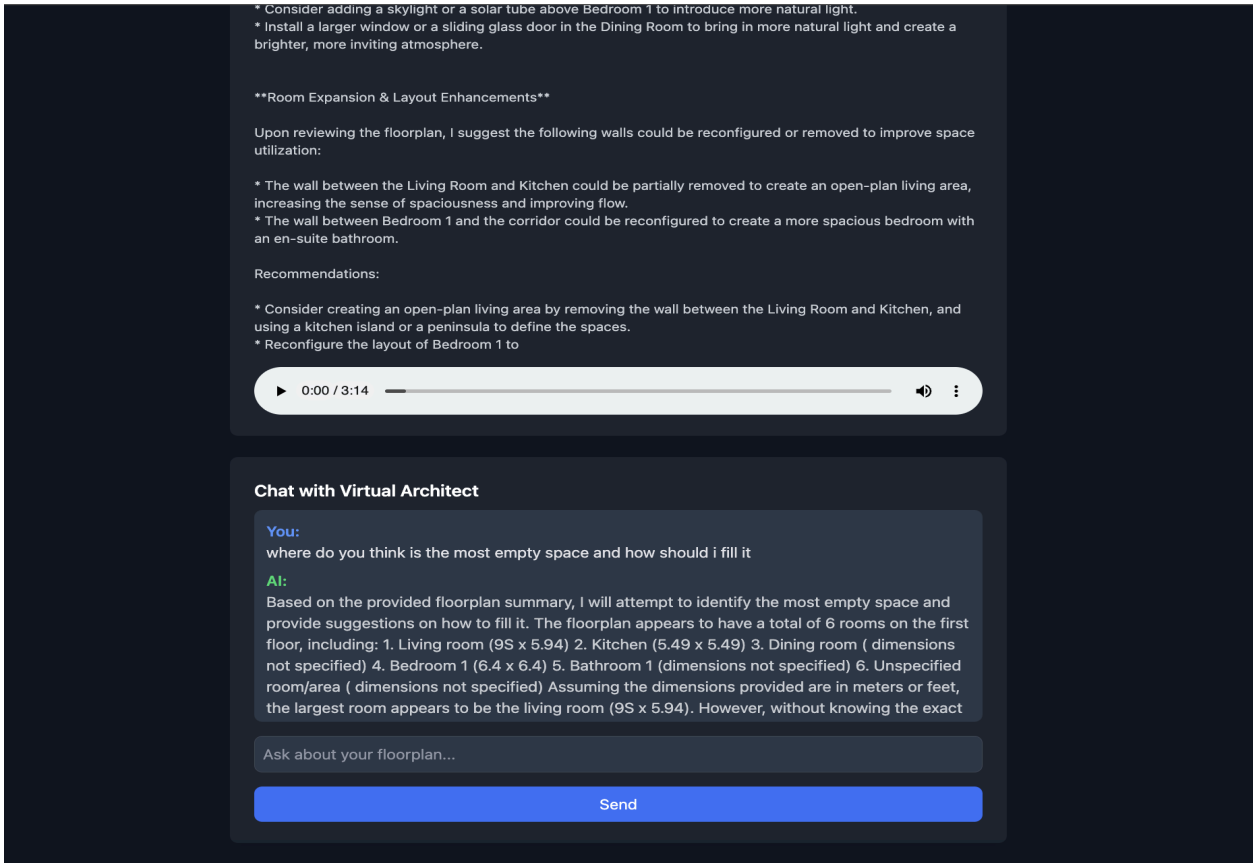
1. **Better AI Models** – Use **GPT-4V, DeepSeek, or Claude** for **built-in image-to-text recognition**, improving text extraction accuracy.
2. **Custom Object Detection** – Train **YOLOv8** on **MakeIt AI floorplan datasets** for **precise room, furniture, and layout detection**.
3. **Interactive UI** – Enhance with **drag-and-drop layout editing, real-time AI tooltips, and 3D modeling integration**.
4. **Faster & Scalable Deployment** – Optimize **API response times** and deploy on **Hugging Face Spaces with GPU or cloud hosting**.

These upgrades will improve **accuracy, usability, and performance**, making the tool more powerful for architects and designers.

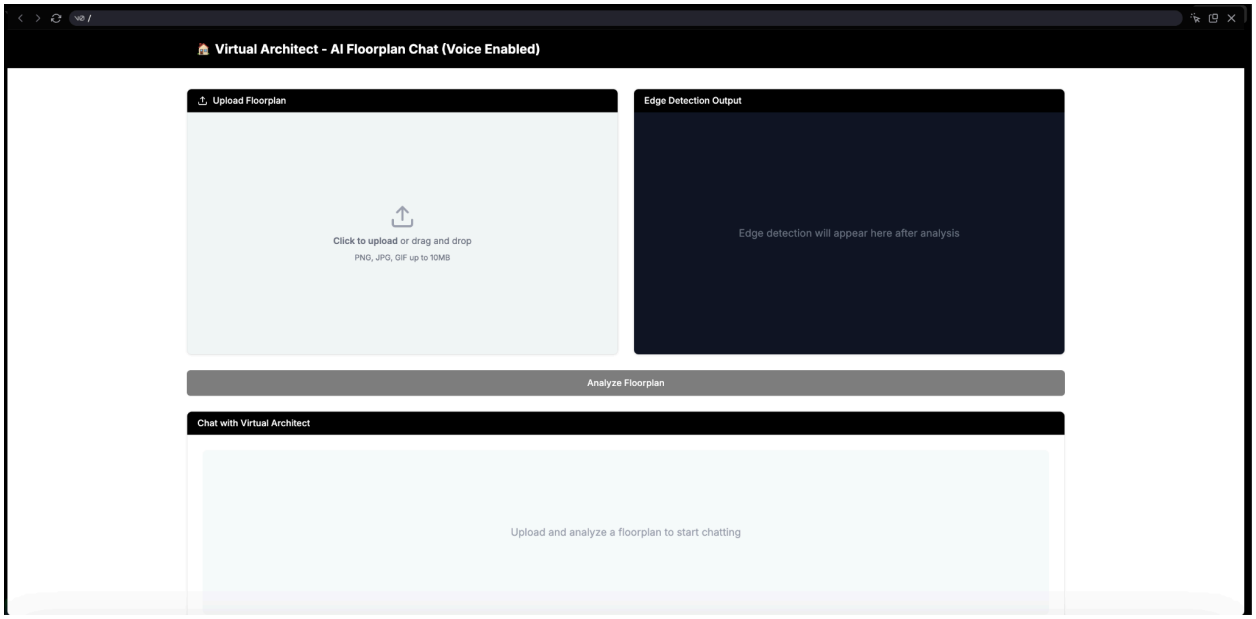
---

8. Some Previous Iterations





(IMG8 - Previous Iteration During Development Phase without any Interactive , Dynamic or modern UI)



(IMG9 - Another Previous Iteration in which UI made Using Loveable but was not Interactive or Dynamic)

9. What I Learned ?

Throughout this project, I gained **valuable insights beyond just technical skills**. While implementing different iterations—**Gradio with Hugging Face** and **FastAPI with React & Tailwind**—I realized that **problem-solving, adaptability, and continuous learning** matter more than just coding expertise.

As I refined the tool, I saw firsthand that **engineering isn't just about knowing everything upfront but about iterating, improving, and embracing challenges**. This aligns with a key perspective:

> **"There's no shortage of talented developers, but what truly sets apart a great engineer is curiosity, adaptability, and the ability to work without ego. The best engineers aren't the ones who claim to know everything but those who continuously learn, admit when they're wrong, and grow from every challenge."**

This project reinforced that mindset—building something useful isn't about perfection from the start, but about **learning, refining, and pushing boundaries with each iteration**. Excited to see what comes next!