AN OFFLINE AI ASSISTANT FOR ESIM: EASIER, ACCESSIBLE, OPEN-SOURCE CIRCUIT DESIGN AND DEBUGGING

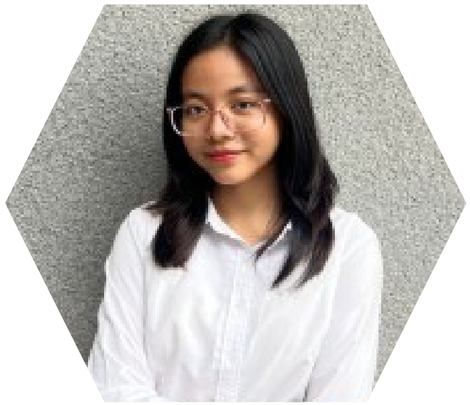
Presented By Sumanto Kar, Aditya Bhattacharya

MENTORS

SUMANTO KAR

PROF. KANNAN MOUDGALYA

CONTRIBUTORS



MYO THINZAR KYAW

ADITYA BHATTACHARYA

RUDRA MANI UPADHAYAY

INTRODUCTION

FOSSEE (Free/Libre and Open Source Software for Education) is a project at IIT Bombay that promotes the use of FLOSS tools in academia and research to reduce dependency on proprietary software. The project is funded by the Ministry of Education, Government of India, through the National Mission on Education through ICT.

OBJECTIVES

The main goals of the FOSSEE project are to:

- Reduce reliance on expensive, proprietary software in educational institutions.
- Encourage the use of equivalent Free/Libre and Open Source Software (FLOSS) tool.
- Develop new FLOSS tools and improve existing ones to meet the needs of the academic and research communities.
- Improve the quality of education by promoting hands-on skill-based training with open-source tools.

INITIATIVES

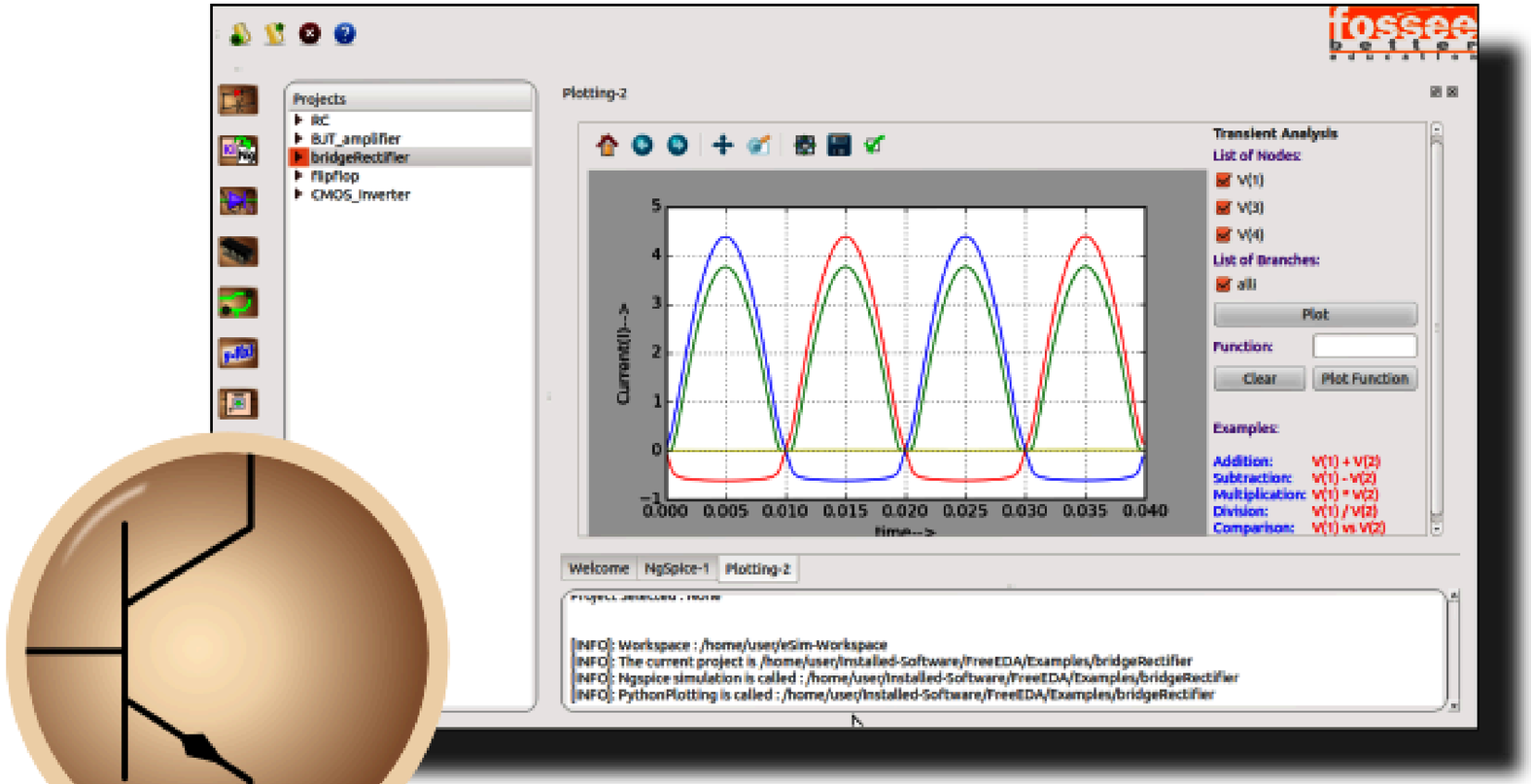
Some of the main Softwares & tools:

- eSim(developed by FOSSEE)
- Scilab and Xcos
- Osdag(developed by FOSSEE)
- OpenFOAM

FOSSEE also promotes the use of other tools for specific purposes:

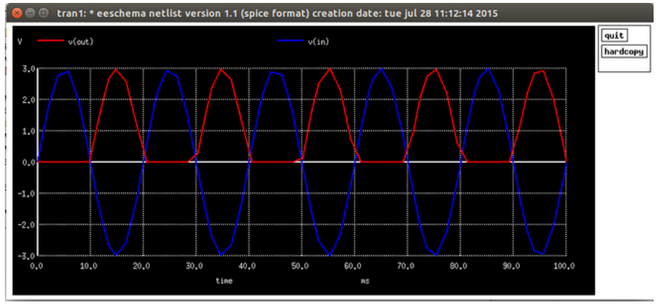
- R
- QGIS
- OpenPLC
- FreeCAD

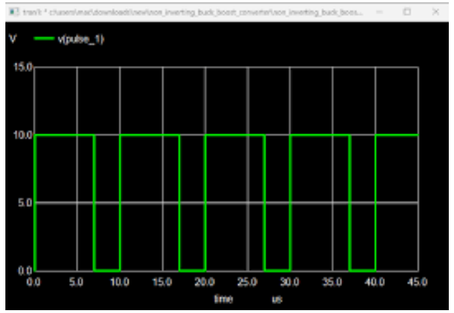
ABOUT ESIM:



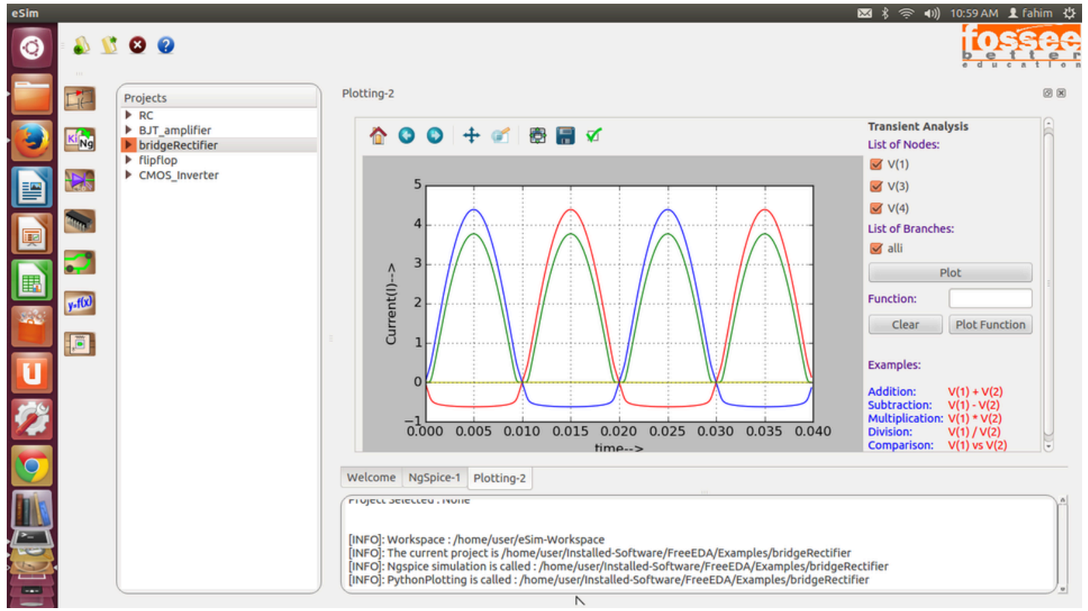
eSim

FEATURES & USES





- **Integrated design environment**: Perform schematic drawing, circuit simulation, and PCB layout design in one platform.
- **Mixed-signal simulation:** Supports analog, digital, and mixed-signal circuits using Ngspice.
- **PCB layout design & Gerber export:** Create multi-layer PCB layouts and generate Gerber files for manufacturing.
- **Custom models & subcircuits:** Build and edit device models with Model Builder and Subcircuit Builder.
- **VHDL support:** Use NGHDL to create custom digital models in VHDL for mixed or standalone circuits.
- **Enhanced plotting & analysis:** Visualize voltages and currents with an interactive Python-based plotting utility.



NEED FOR DEVELOPMENT

- **Absence of Circuit Debugging Support** – No automated mechanism to detect design errors, incorrect connections, or simulation issues.
- **No Query Resolution or Context-Aware Assistance** – Users could not ask "**why**" or "**how**" questions about circuit behavior.
- **Manual Troubleshooting Burden** – Errors had to be identified manually, making the process repetitive and discouraging.
- **Limited Multimodal Interaction** – No integration of visual analysis, textual explanations, or voice feedback.
- **Accessibility and Usability Challenges** – Lack of intuitive guidance for users of different expertise levels.

SOLUTION:
AN OFFLINE AI-ASSISTED DEBUGGING BOT TOOL

BUT WHY OFFLINE: ?

CAUSE

Unreliable Internet Access – Many users (especially students in rural/educational setups) face inconsistent connectivity.

Data Privacy Concerns – Circuit designs and error logs may contain sensitive research data that should not be uploaded to external servers.

Cost of Cloud Dependency – Relying on internet APIs increases costs and creates dependency on third-party providers.

USE

Fixing Errors Automatically – The tool can read error messages from your circuit work and suggest fixes right on your computer.

Classroom and Lab Use – Schools and colleges without strong Wi-Fi can still use the help system fully.

Built-in Help and Guides – The manual and FAQ are already inside the software, no need to open a browser.

BENEFITS

More People Can Use It – Makes eSim a stronger choice compared to expensive software that requires online accounts or licenses.

Equal Access – Students from all backgrounds can use the same high-quality tools without needing fast internet or extra resources.

Stable Performance – No slowdowns caused by weak connections or heavy traffic on servers.

TECHNICAL ASPECTS

GUI AND APPLICATION FRAMEWORK

- PyQt5:
- QTimer / QThread / pyqtSignal:
- OpenCV (cv2):
- Pillow (PIL):
- pyttsx3
- pytesseract:

LARGE LANGUAGE MODEL AND RAG

- Ollama
- LangChain Community Libraries:
- HuggingFaceEmbeddings:
- FAISS:
- TextLoader/UnstructuredPDFLoader:
- RecursiveCharacterTextSplitter:

UTILITY AND SYSTEM LIBRARIES

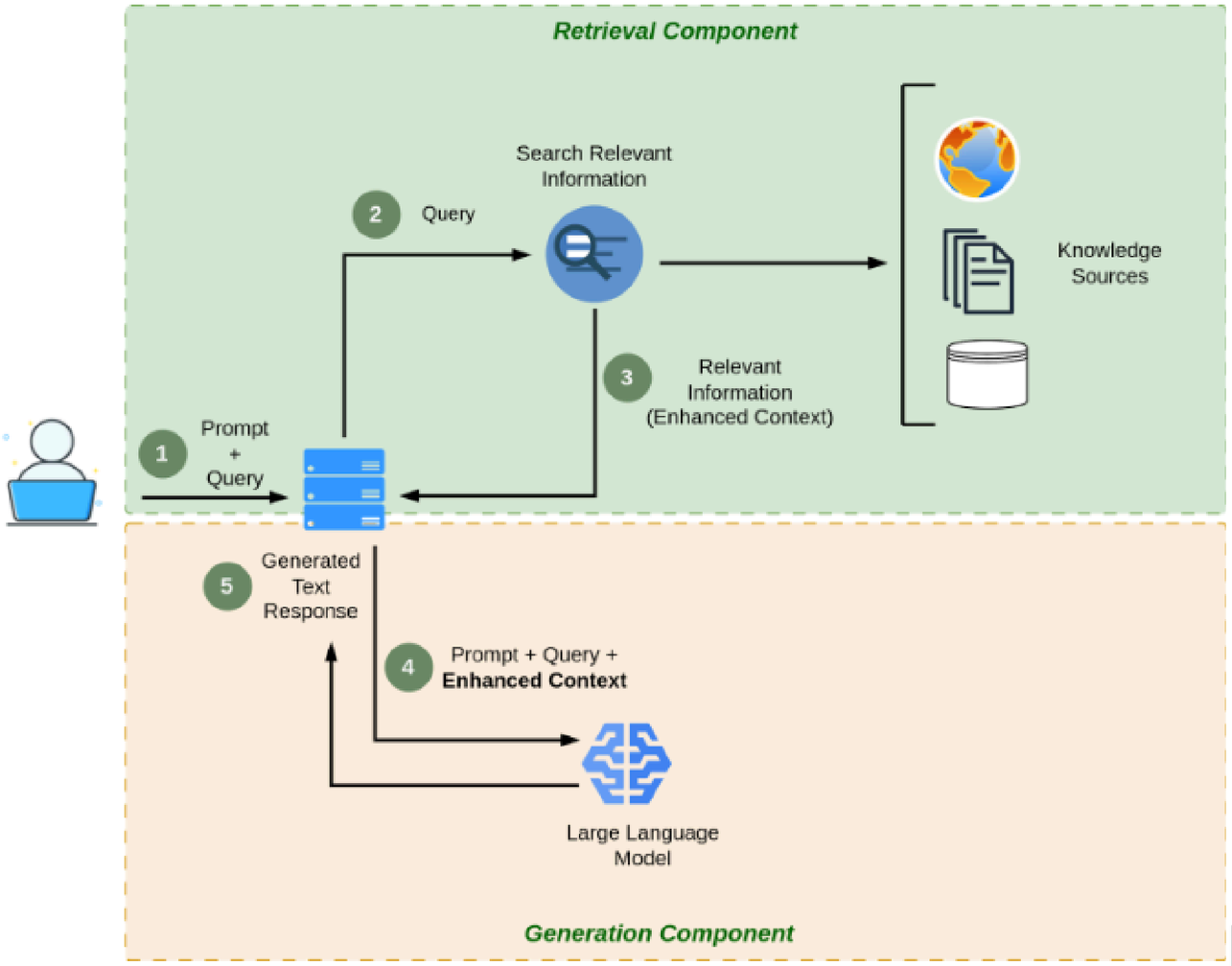
- pyautogui/ImageGrab:
- NumPy
- os / pathlib / time / datetime / re / base64:

CONCEPTS & APPROACH

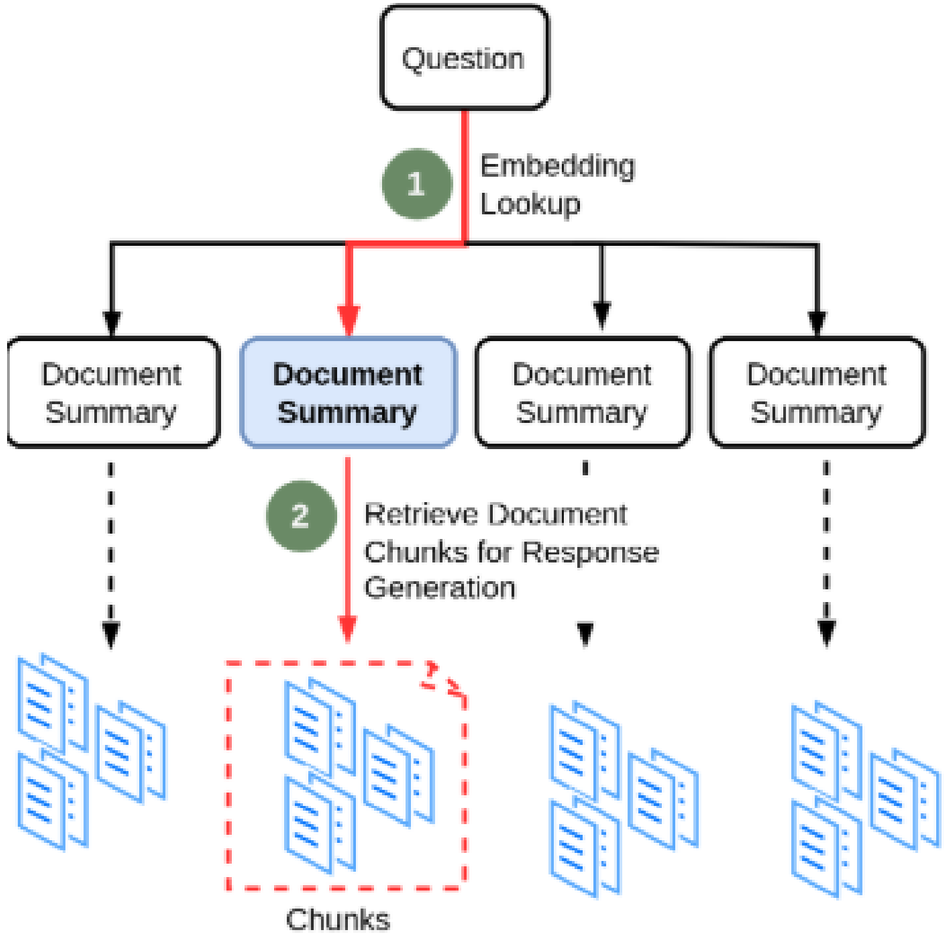
- **Retrieval-Augmented Generation (RAG)**
  - **What it is:** Combines document retrieval with language model generation.
  - **Why used:** To reduce hallucinations and give accurate, context-grounded answers.
- **. Vector Databases**
  - **What it is:** Specialized databases that store embeddings for similarity search.
  - **Why used:** To efficiently retrieve the closest matching document chunks during question answering.
- **Embeddings / Vectorization**
  - **What it is:** Converting tokens into dense numerical vectors that capture meaning.
  - **Why used:** To let the model understand relationships between words in error logs.
- **Error Classification**
  - **What it is:** Categorizing input data into predefined error types.
  - **Why used:** To identify whether an error is due to missing libraries, invalid parameters, etc.
- **Rule-Based System**
  - **What it is:** A logic-based system that applies fixed “if–then” rules.
  - **Why used:** To catch straightforward mistakes (like missing library paths or invalid time steps) quickly.

SCHEMATIC DIAGRAM

RAG: WORKFLOW

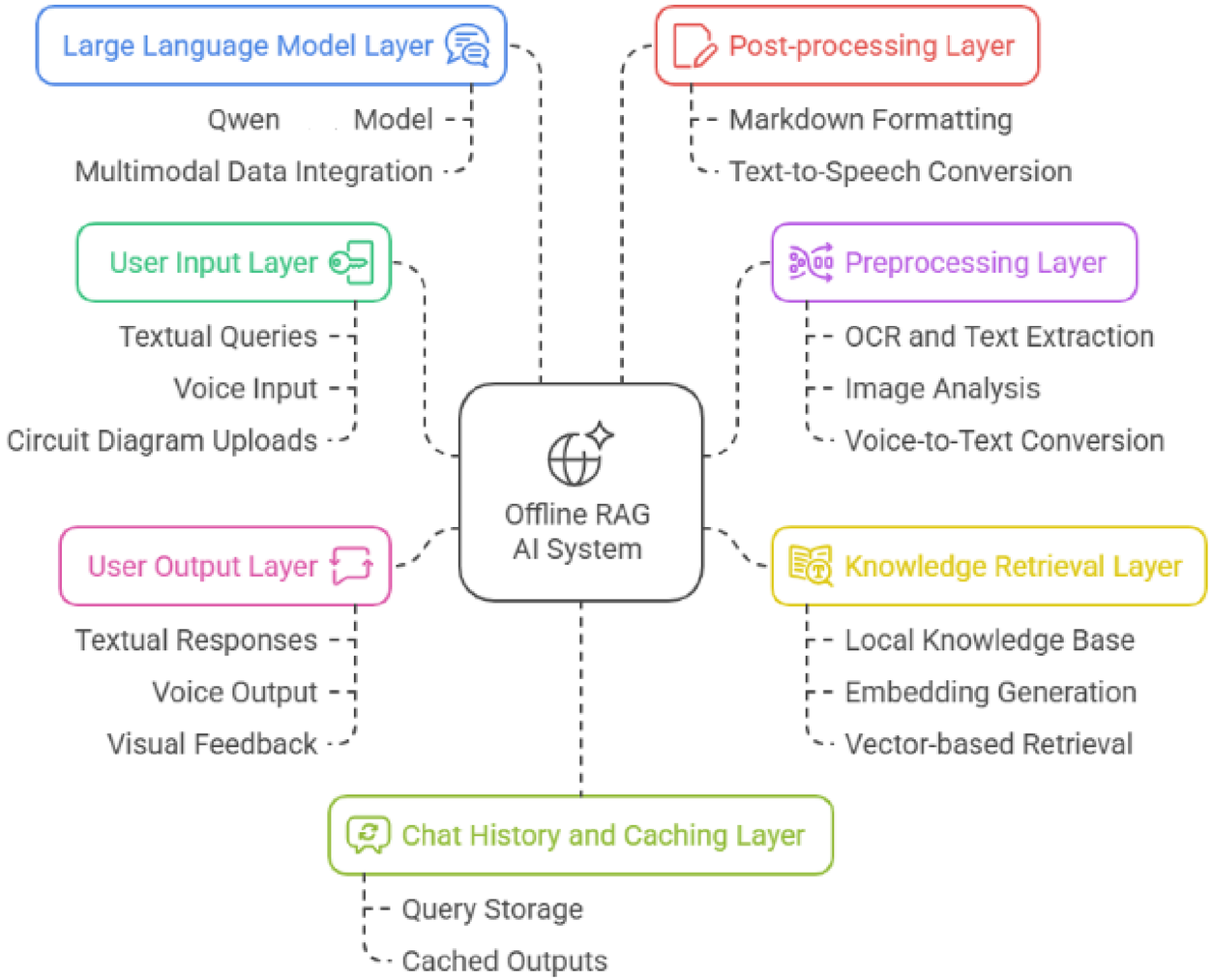


EMBEDDING & CHUNKING

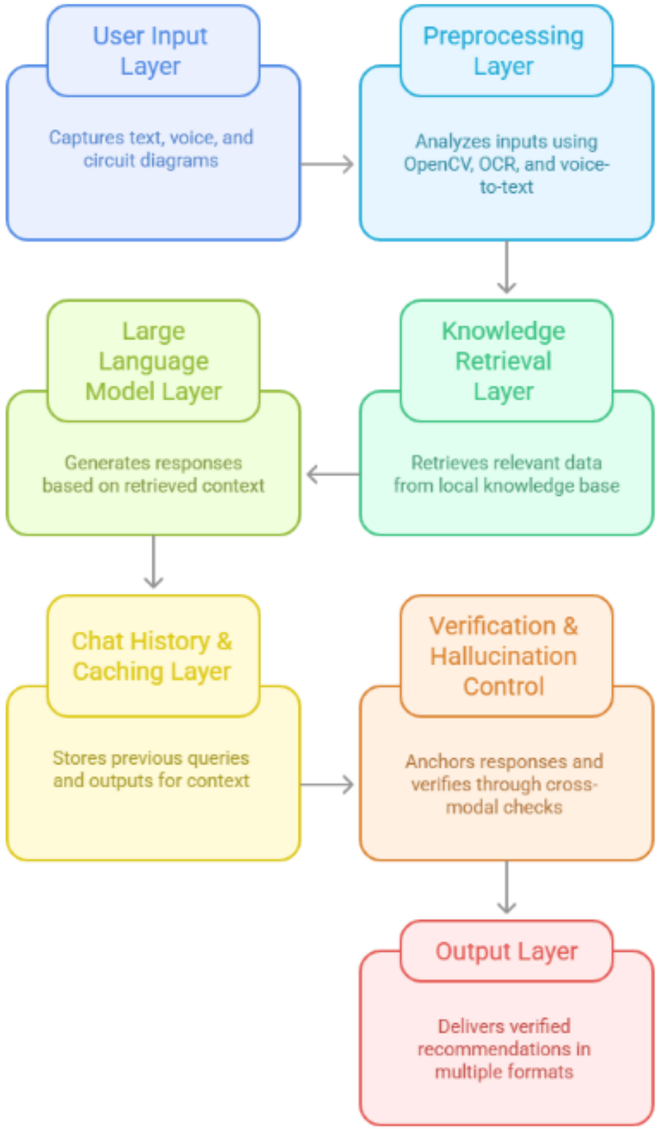


SCHEMATIC DIAGRAM

FEATURES & FUNCTIONS



WORKFLOW



PARADIGM EXECUTION

Components & Layers

- **Large Language Model Layer**
  - **Qwen 2.5B Model –** Lightweight open-source LLM used for local inference.
  - **Multimodal Data Integration** – Handles multiple input types like text, images, and diagrams.

- **User Input Layer**
  - **Textual Queries** – Accepts user input in natural language.
  - **Voice Input** – Captures spoken queries for hands-free interaction.
  - **Circuit Diagram Uploads** – Allows users to upload circuit images for analysis.

- **User Output Layer**
  - **Textual Responses** – Provides AI-generated answers in text format.
  - **Voice Output** – Converts responses into speech for accessibility.
  - **Visual Feedback** – Displays processed diagrams or analyzed results visually with rich formatting for readability.
  - **Text-to-Speech Conversion** – Converts final text answers into natural-sounding audio.

- **Preprocessing Layer**
  - **OCR and Text Extraction** – Converts scanned or image-based text into machine-readable format.
  - **Image Analysis** – Processes diagrams or figures to extract relevant data.
  - **Voice-to-Text Conversion** – Transforms spoken input into text for further processing.

- **Knowledge Retrieval Layer**
  - **Local Knowledge Base** – Stores domain-specific information offline for reference.
  - **Embedding Generation** – Converts text into dense vectors for semantic search.
  - **Vector-based Retrieval** – Finds the most relevant information using similarity search.

- **Chat History and Caching Layer**
  - **Query Storage** – Saves past user queries for context continuity.
  - **Cached Outputs** – Reuses previously generated results to reduce computation.

USER FLOW-PROCESS

1. Circuit Design Input

- User designs a circuit in the tool.
- Any modification triggers the File Watcher, which detects changes automatically.

2. Data Processing

- The Data Extractor parses design files (netlist, components, simulation setup).
- OpenCV performs visual analysis of the schematic to check symbols and structure.

3. Knowledge Retrieval

- The system uses RAG (Retrieval-Augmented Generation) to search a knowledge base for relevant troubleshooting information and theory.

4. Query Composition

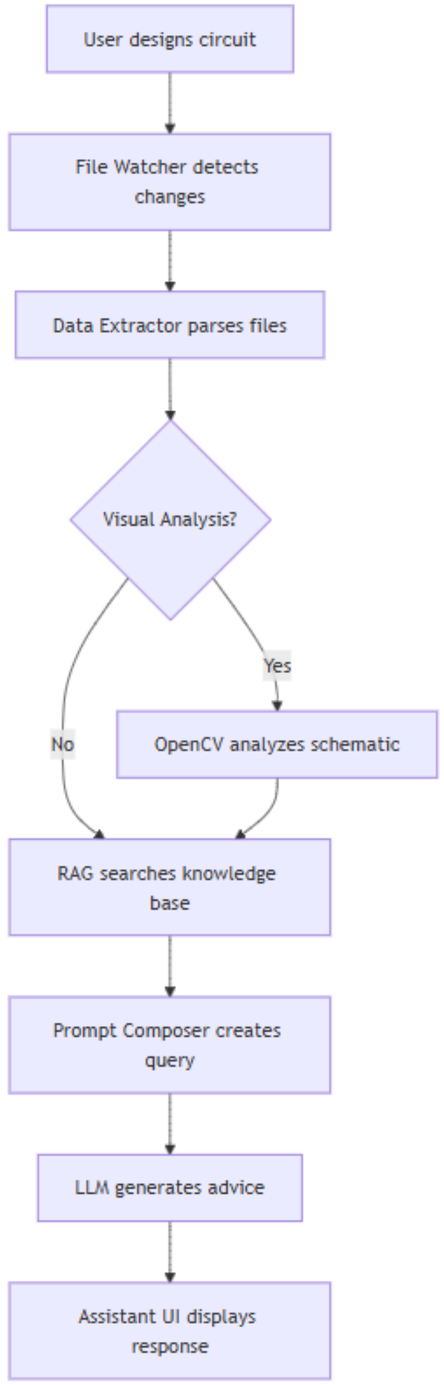
- Prompt Composer combines extracted data + knowledge base results into a structured query.

5. AI Reasoning

- LLM (AI model) + LSTM generates context-aware advice, error explanations, and debugging suggestions.

6. User Assistance Output

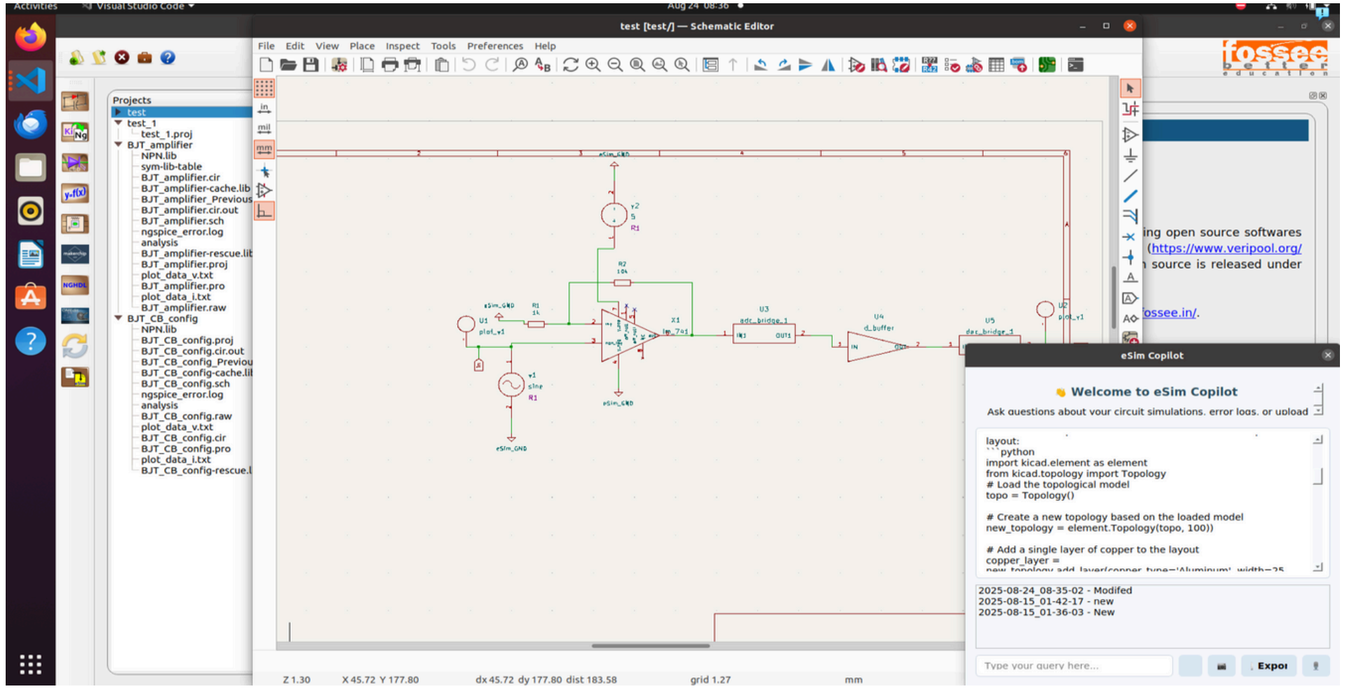
- The Assistant UI displays the response with recommendations, fixes, or learning explanations.



HANDLING CIRCUIT-RELATED QUESTIONS AND ERRORS

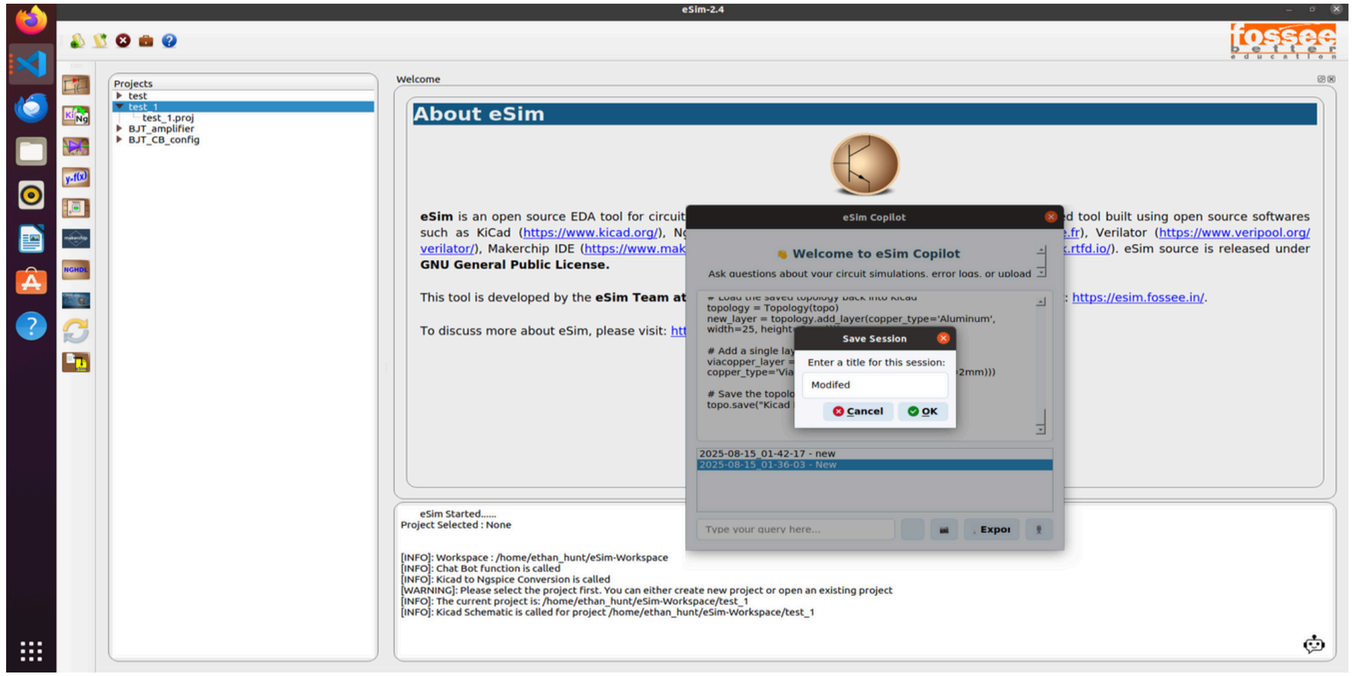
The Chatbot is capable of:

- **Answering Circuit Theory Queries:** From basic resistor–capacitor questions to more advanced transistor biasing issues.
- **Interpreting Simulation Logs:** When Ngspice generates an error, the Chatbot can analyze the output text, detect keywords (e.g., **“Convergence Problem,” “No such device or model”**), and respond with targeted advice.
- **Guiding Parameter Adjustments:** The Chatbot can suggest valid ranges or default values for common SPICE parameters if they are missing or incorrectly specified.
- **Suggesting Debug Steps:** For instance, recommending that the user verify library paths, check netlist syntax, or run a DC operating point analysis prior to a transient simulation
- **Recommending Component Selection:** Helps users choose appropriate components (e.g., resistor wattage, capacitor type, transistor models) based on circuit requirements.
- **Explaining Simulation Results:** Interprets waveforms (voltage, current, frequency plots) and highlights mismatches between expected and actual outcomes.
- **Detecting Common Design Mistakes:** Identifies floating nodes, missing ground connections, or short circuits in the schematic.
- **Providing Educational Insights:** Offers theoretical background on concepts (**e.g., RC time constant, biasing conditions, feedback mechanisms**) linked to the user’s circuit.

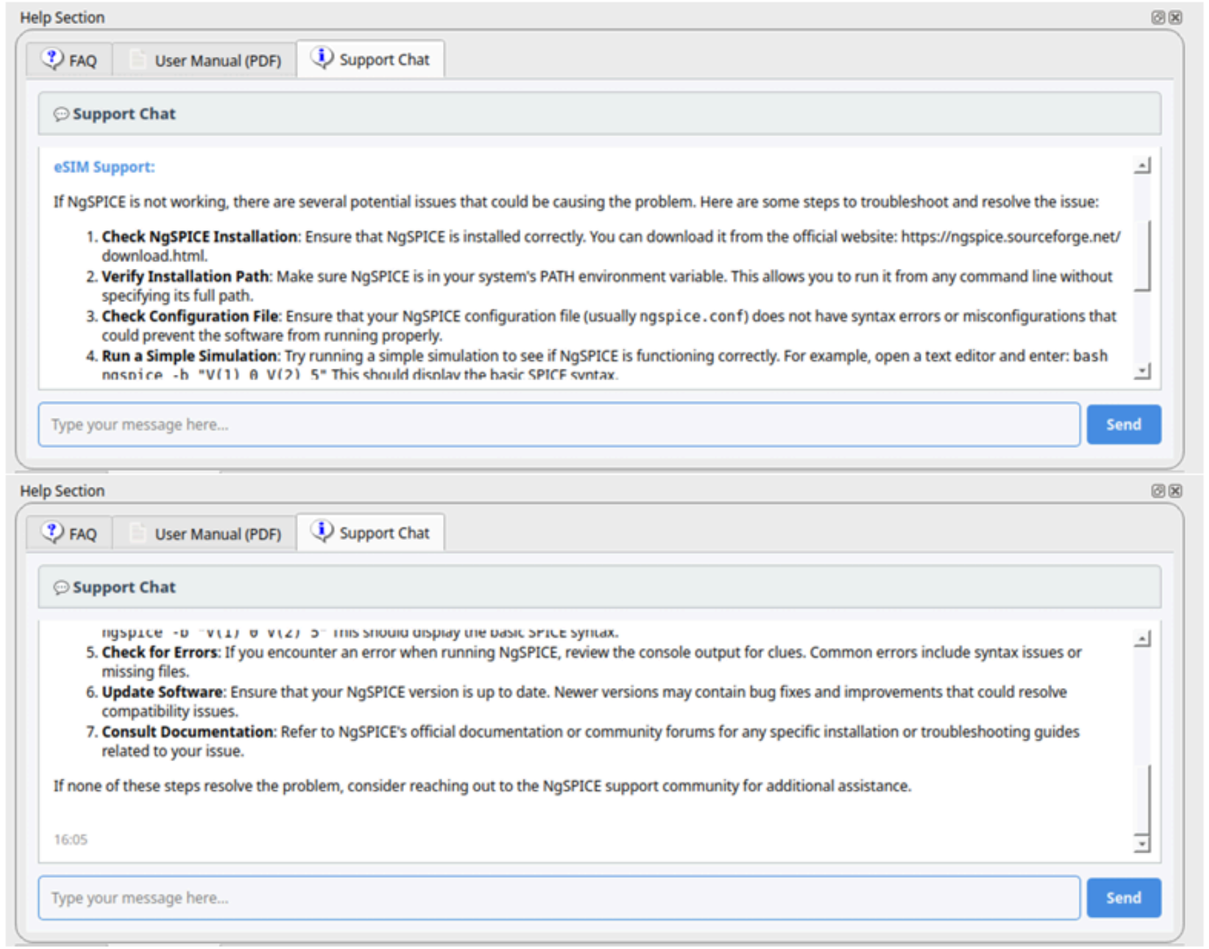


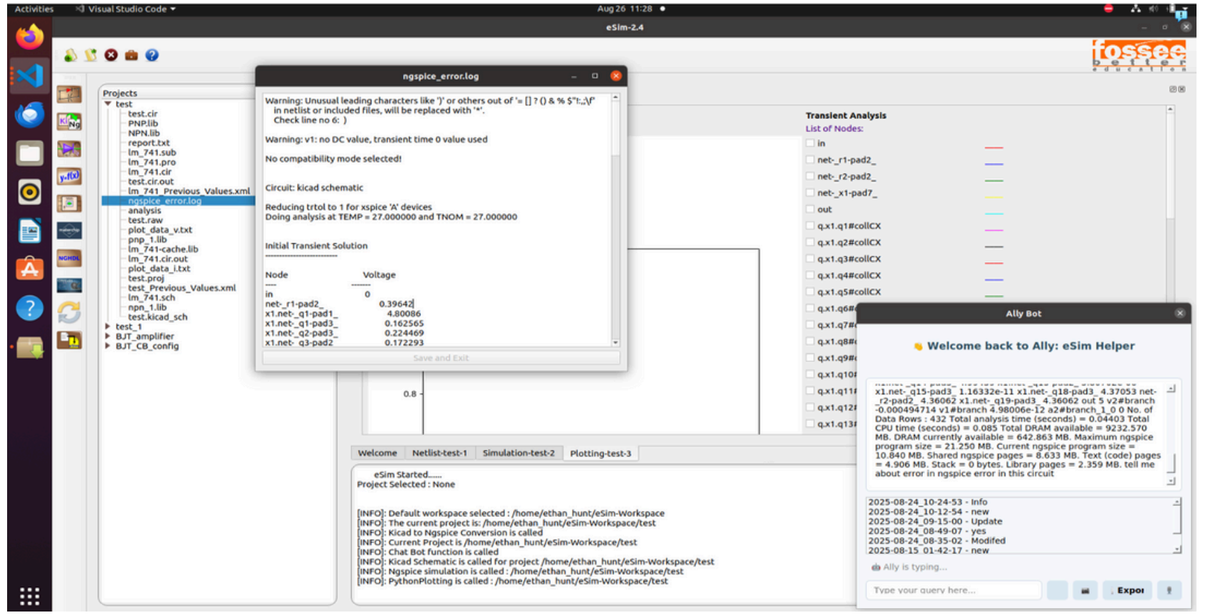
REAL TIME ANALYSIS & WORKING SCENARIOS

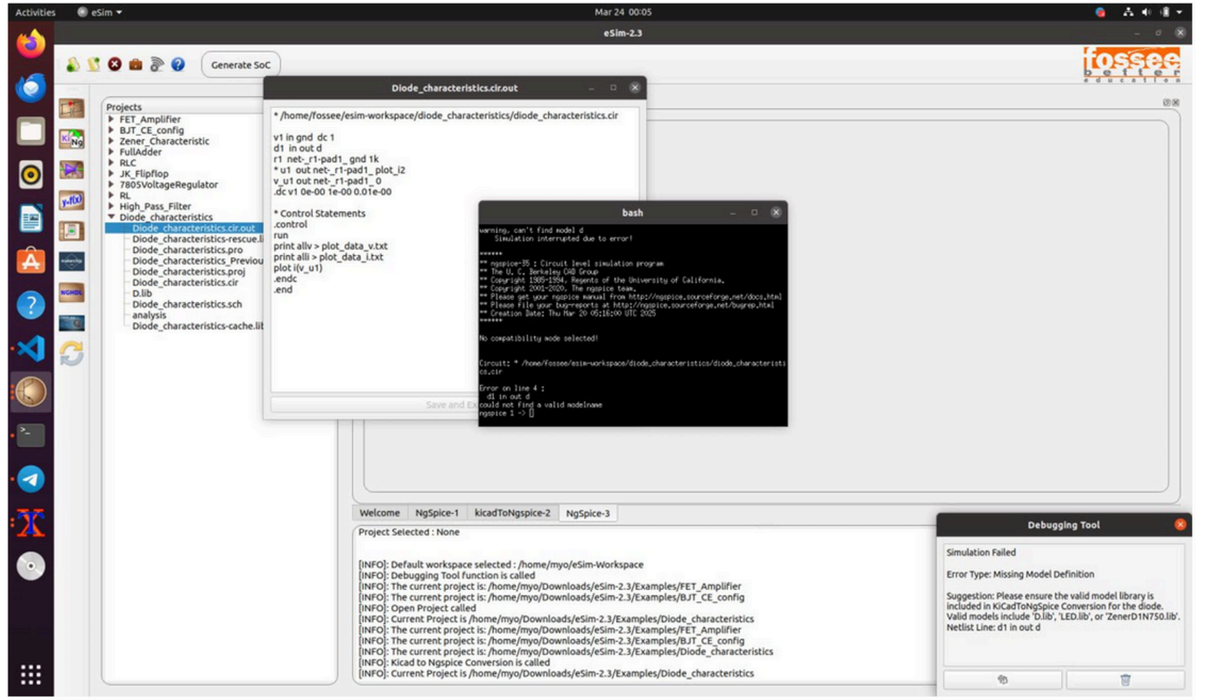
DEBUGGING SESSION & ANALYSIS OF ERRORS

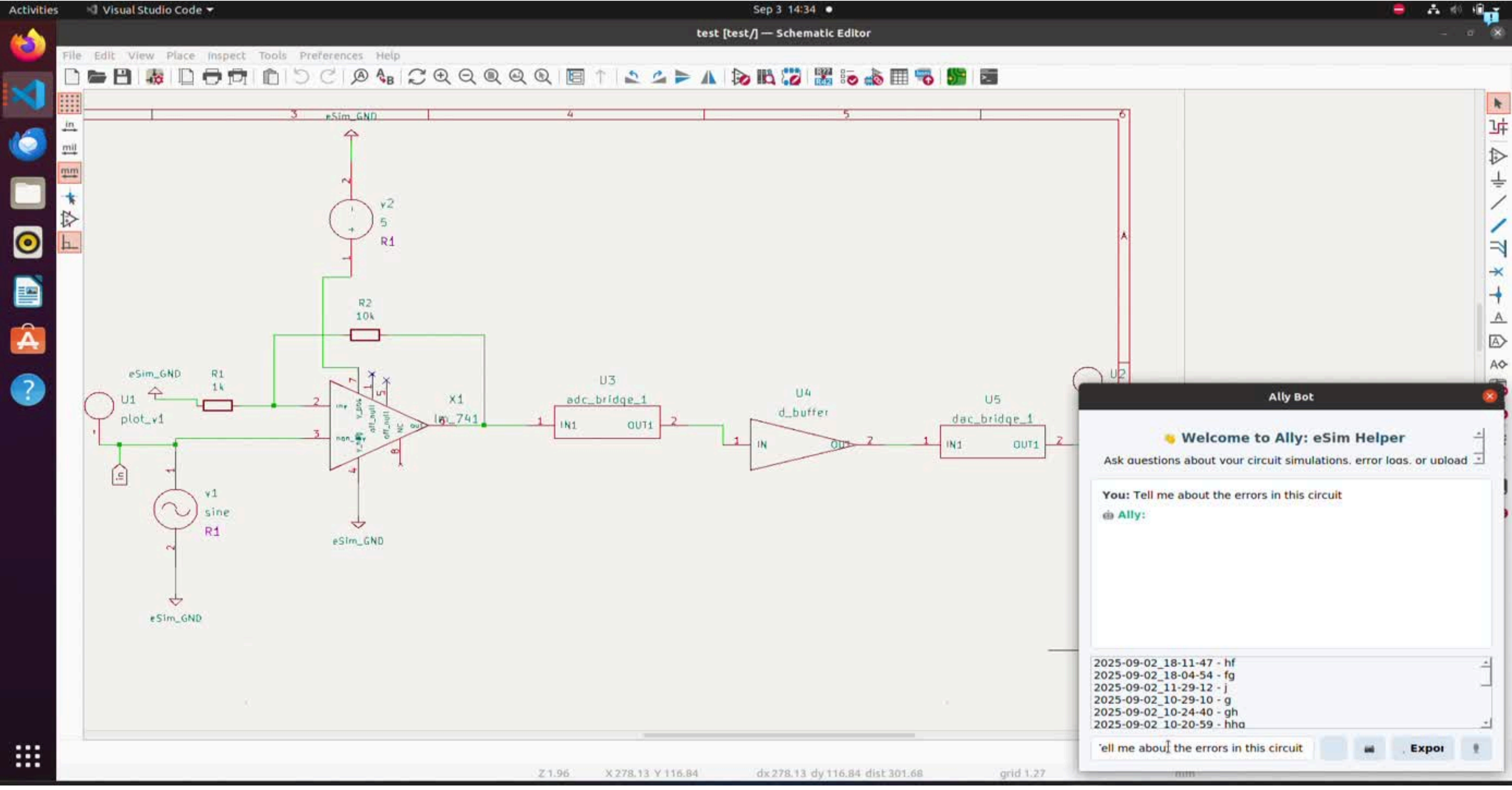


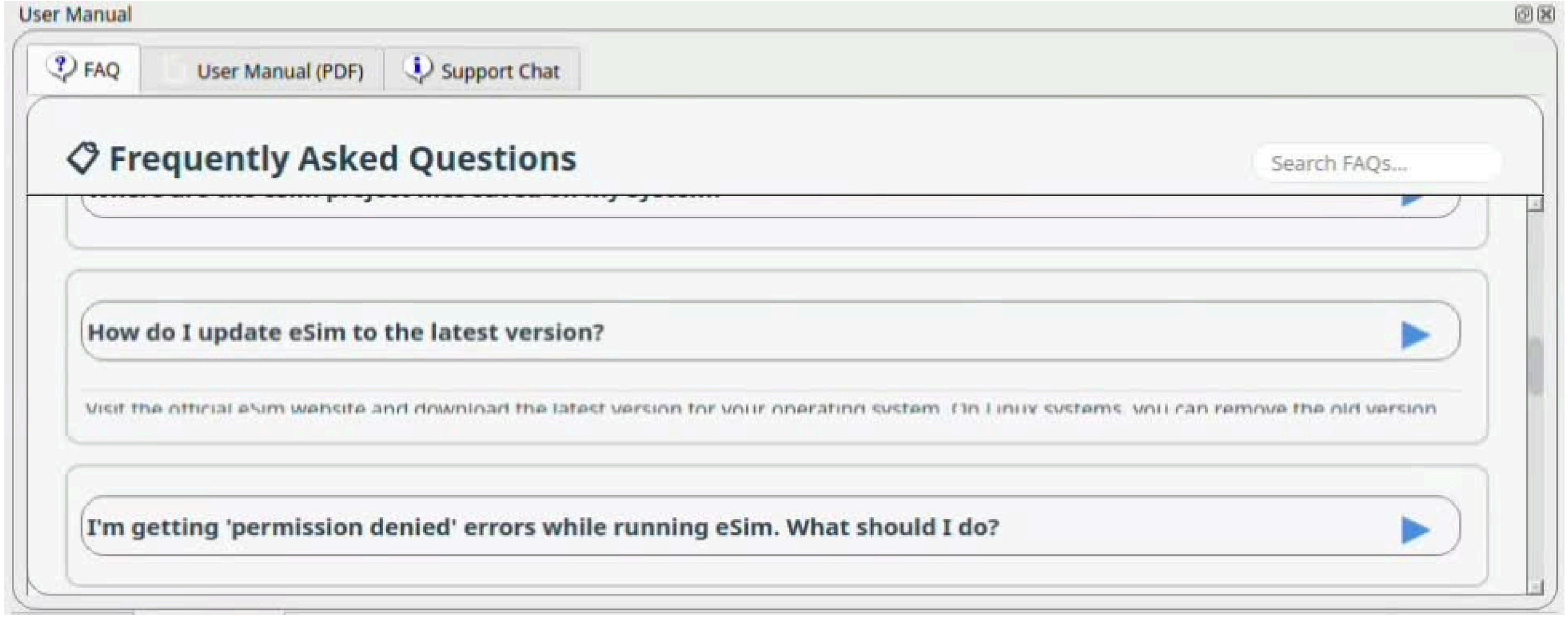
FAQ & CHATBOT SECTION: WORKING DEMO



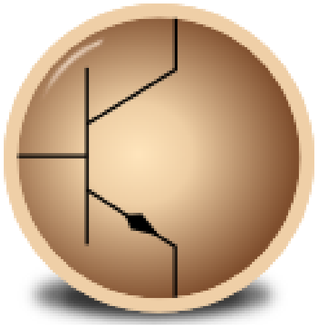





Activities
Visual Studio Code
Sep 3 14:34
test [test/] — Schematic Editor
File Edit View Place Inspect Tools Preferences Help
eSim_GND
v2
5
R1
R2
10k
eSim_GND
R1
1k
U1
plot_v1
X1
lm_741
U3
adc_bridge_1
IN1
OUT1
U4
d_buffer
IN
OUT
U5
dac_bridge_1
IN1
OUT1
U2
v1
sine
R1
eSim_GND
eSim_GND
Ally Bot
Welcome to Ally: eSim Helper
Ask questions about your circuit simulations, error logs, or upload
You: Tell me about the errors in this circuit
Ally:
2025-09-02_18-11-47 - hf
2025-09-02_18-04-54 - fg
2025-09-02_11-29-12 - j
2025-09-02_10-29-10 - g
2025-09-02_10-24-40 - gh
2025-09-02_10-20-59 - hhg
'ell me about the errors in this circuit
Expor
Z 1.96
X 278.13 Y 116.84
dx 278.13 dy 116.84 dist 301.68
grid 1.27


User Manual
FAQ
User Manual (PDF)
Support Chat
Frequently Asked Questions
Search FAQs...
How do I update eSim to the latest version?
I'm getting 'permission denied' errors while running eSim. What should I do?

FUTURE DEVELOPMENT GOALS



Phase 1: Core AI/ML Foundation

- Build diverse training datasets.
- Develop circuit-specific embeddings.
- Set up continuous evaluation.

Phase 2: Domain-Specific Intelligence

- Fine-tune for RTL/HDL and PCB tasks.
- Add multimodal ML (text, diagrams, outputs).
- Apply feedback-driven learning (RLHF).

Phase 3: Scalable and Adaptive AI Integration

- Optimize with quantized models for low-end devices.
- Deploy adaptive Agentic RAG for speed vs accuracy.
- Enable continual learning with eSim updates.

Phase 1:

Phase 2:

Phase 3:

Phase 1: Data & Model Enhancement

- Expand training with diverse simulation errors.
- Collect user feedback for error handling.
- Enable continuous model retraining.

Phase 2: Debugging & Verification

- Add RTL/HDL debugging support.
- Introduce PCB verification features.
- Use domain-specific embeddings.

Phase 3: Deployment & Integration

- Release as part of official eSim package.
- Optimize for low-end devices with quantized models.
- Provide update mechanism for improvements.

CONTACT US

(+91) 22 25764111

Email : info.fossee.in, contact-esim@fossee.in

https://fossee.in, https://esim.fossee.in

Aero-Annex Building, IIT Bombay


THANK YOU