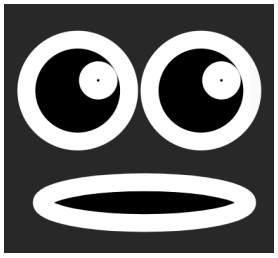
Sugar Labs: Project Proposal

Google Summer of Code 2025



Project Information

Name: Refactor the chatbot in the Speak Activity to use gen-AI [here]
Length: 175 Hours
Mentor: Ibiam Chihurumnaya
Assisting Mentor: Walter Bender

Student Information

Full Name: Aman Naik

Email: amancodes@gmail.com

GitHub Username: amannaik247

Linkedin link: https://www.linkedin.com/in/aman-naik

Matrix Username: @amannaik:matrix.org

Education: Mumbai University, Bachelors in Information Technology

Location and Timezone: Maharashtra, India (UTC+5:30)

What makes me a GOOD FIT?

I am excited to apply for Google Summer of Code 2025 with Sugar Labs, an organization whose mission deeply aligns with my personal interests and professional experience. Over the past several months, I have engaged actively with Sugar Labs' development ecosystem and contribute meaningfully to its mission of empowering children through education-focused software. I believe I bring a unique combination of technical skill, teaching experience, and a passion for child-centric learning tools, making me a strong fit for this opportunity.

Ongoing Contribution to Sugar Labs

Since November 2024, I have been an active contributor to Sugar Labs. During this time, I have explored, tested, and contributed to multiple Sugar Activities—interactive, educational games that help children learn through play. Through my participation in issue resolution, UI improvements, and codebase familiarity, I have gained practical insights into the development philosophy and technical structure behind Sugar Labs' projects.

Direct Experience with Activities

My contributions go beyond code—I have immersed myself in understanding how Sugar Activities are designed to engage young minds. I have built new activities, modified existing ones, and actively participated in discussions to enhance educational impact, accessibility, and user interaction. This hands-on experience has equipped me with the practical knowledge necessary to further innovate within the Sugar ecosystem. Link to my activity [here]

Background in Teaching Young Children

I also bring a valuable perspective as an educator. I have taught children between the ages of 6 to 10, primarily focusing on language development. This direct classroom experience gives me a deep understanding of how children learn, what keeps them engaged, and how to tailor educational content to their developmental needs. It also fuels my enthusiasm for building tools that make learning intuitive and enjoyable.

Passion for Language and Child-Centric Interaction

My teaching experience has cultivated a strong interest in both language development and meaningful communication with children. I believe this interest is crucial in designing educational software that resonates with young users. My ability to empathize with how children think and learn helps me create experiences that are not only functional but also engaging and age-appropriate.

Technical Expertise in AI and Language Models

In addition to my work with Sugar Labs, I have substantial experience with artificial intelligence and large language models (LLMs). I have built several AI-powered projects—ranging from intelligent chatbots to document analyzers—which demonstrate my ability to solve problems using advanced machine learning techniques. (Projects on github)

Research Recognition

Lead a team for an AI research project on commodity price prediction using AI-powered sentiment analysis that was awarded among the Top 10 projects at a state-level competition. This experience not only strengthened my technical skills but also deepened my understanding of how AI can be used to solve real-world problems—especially in domains like education, economics, and communication. Link to the award certificate.

Experienced Graphic Designer

I bring over 3 years of professional experience as a graphic designer. This creative background enables me to design user interfaces and visual experiences that are intuitive and appealing to young users. I believe visuals play a critical role in how children engage with learning tools, and my design skills can significantly enhance the usability and attractiveness of Sugar Activities.

With a blend of technical proficiency, real-world teaching experience, and a strong commitment to educational innovation, I am confident that I can make meaningful contributions to Sugar Labs through Google Summer of Code 2025. I am eager to bring my creativity, skills, and dedication to this community and help create tools that continue to enrich learning experiences for children across the globe.

Why Sugar Labs?

From the moment I discovered Sugar Labs, I felt a deep connection to its mission. The idea of building tools that spark curiosity, creativity, and learning in children—especially in a way that's open and community-driven—feels incredibly meaningful to me. I've always believed that technology should empower, not overwhelm, and Sugar Labs embodies that belief. The focus on activities over applications, on letting children explore and express themselves freely, aligns beautifully with how I think learning should feel: joyful, curious, and hands-on.

Personally, I remember being that curious child who was eager to click every button, open every file, and understand how things worked. If I had something like Sugar growing up—where I could learn not just by using technology but by building with it—it would have changed my world. Now, I want to help build that world for others. Contributing to Sugar Labs is not just an opportunity to learn and grow as a developer, but to be part of a mission that is bigger than code. It's about helping children everywhere discover their voice through technology. I'm keen to work alongside this amazing community, contribute meaningfully, and be a small part of a beautiful, global effort.

My Contributions

Pull requests

PR	Repo	Title	Status
link	jamath-activity	Update activity.info - added summary	merged
link	ball-and-brick-activity	"Port to Python 3" - fix download link	merged
link	jamath-activity	"Port to Python 3" - fix download link	merged
link	abacus-activity	Update README.md	merged
link	speak	Port to Python 3 - fix download link	merged
link	aslo-v4	Update generator.py - Fixed License NO ASSERTION issue	merged

Issues raised

IR	Repo	Title	Status
link	www	Events tab contains old article links	open
link	infoslicer-activity	Activity not available in https://v4.activities.sugarlabs.org/	closed

Experience with Creating Activities

Hands-On Development with Sugar Activities
My hands-on experience with creating and using Sugar Activities has equipped me with the technical and creative skills necessary to contribute meaningfully to Sugar Labs. One of my most notable contributions is the development of Word Quest, a language-based educational game that blends fun and learning to help children expand their vocabulary and language skills.
Link to github repository of the activity.

Overview of Word Quest
Word Quest is an engaging 5-letter word-guessing game designed with simplicity and educational value in mind. The activity encourages learners to think critically, recognize spelling patterns, and build language intuition—all while playing an enjoyable game. Players have six chances to guess the correct word, receiving feedback through a color-coded hint system that indicates whether letters are correct, present in the word but misplaced, or not in the word at all.

Customizable and Reflective Learning Features
What sets Word Quest apart is its category selection feature. Players can choose from six different word categories, tailoring the game to their interests or curriculum focus. This personalization makes the activity versatile for various learning levels and subject areas. To support long-term language development, Word Quest includes a Personal Dictionary—a dedicated space where users can save new words they encounter during the game.

Focus on User-Centered Design
I placed strong emphasis on an intuitive and child-friendly design, ensuring that even first-time users can navigate the interface with ease and confidence. The layout and interactions are crafted to support a seamless experience for young learners while keeping the interface clean and responsive.

Technical Experience and Platform Familiarity

Creating Word Quest allowed me to dive deep into the Sugar Activity framework, learning how to integrate interactive elements, manage data persistence, and optimize user experience for children. Through this process, I gained valuable insights into the development and deployment of Sugar Activities and the pedagogical considerations behind them.

Comfort with the Sugar Ecosystem

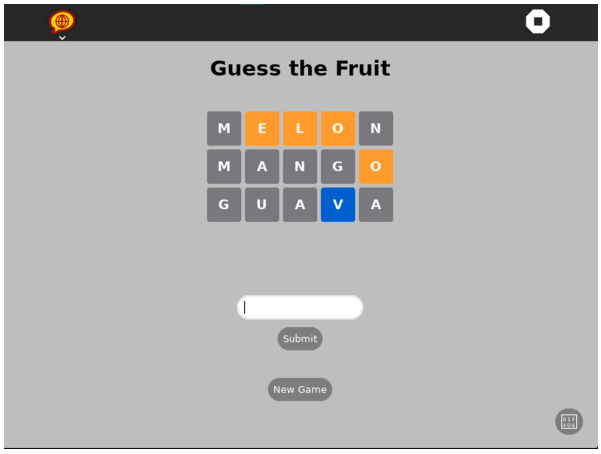
Beyond Word Quest, I have actively explored and tested a variety of existing Sugar Activities to understand what works well and where there is room for innovation. This experience has made me comfortable with the Sugar environment, from both a user and developer perspective.

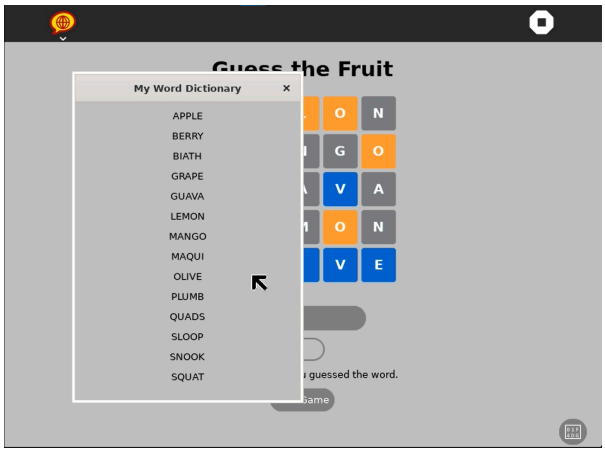
Preparedness for GSoC Project Development

I believe that this background makes me well-prepared to take on a Google Summer of Code project with Sugar Labs. I understand the ecosystem, care deeply about the educational mission, and have demonstrated the ability to build meaningful, engaging activities that align with Sugar Labs' vision.

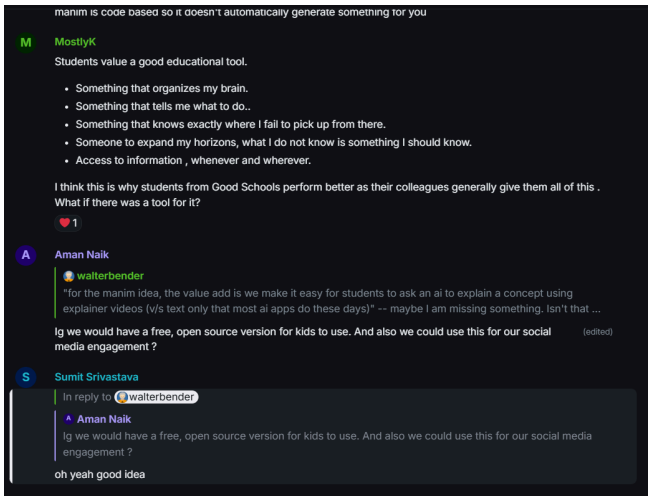
Images and videos Word-Quest

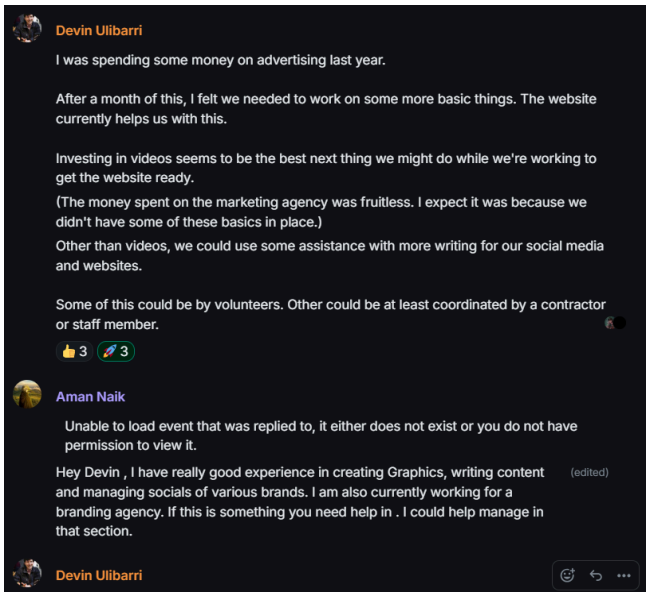
[Link to video Demo]

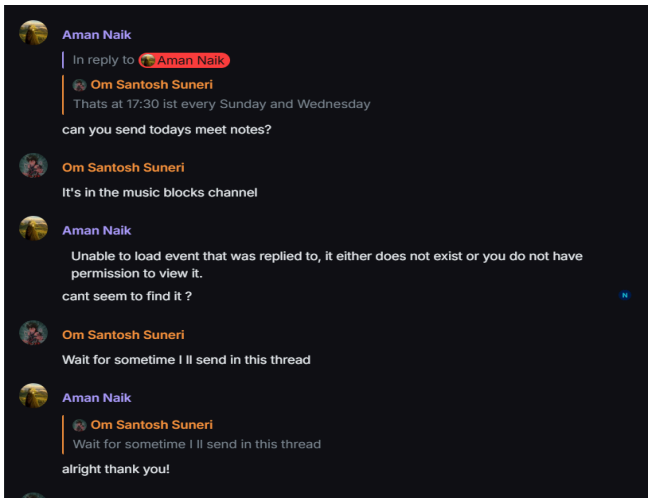




Matrix Chat Engagement







Volunteer Management of Social media content

Voluntary Role in Community Engagement

In addition to my technical contributions, I have been actively involved in managing Sugar Labs' social media presence. This role has allowed me to contribute to the visibility and outreach of the project, helping engage both existing and potential contributors. I have worked closely with Devin, a core maintainer at Sugar Labs, to strategize and execute content for multiple platforms.

Automation of Contributor Highlights

One of the initiatives I led was the automation of showcasing contributor work on social media. To streamline the process, I created a Google Form and linked Spreadsheet that allows contributors to submit their updates. These submissions are then used to automatically generate social media-ready content, making it easier to recognize and promote community contributions consistently.

Visual Design for Event Promotion

Leveraging my background in graphic design, I also created custom graphics for upcoming Sugar Labs events and announcements. These visuals are tailored to maintain a consistent and friendly aesthetic that reflects the educational mission of Sugar Labs, while also being engaging for online audiences.

Planning Content for Year-Round Automation

Recognizing the importance of keeping our social media active throughout the year, I am implementing the idea of creating "perennial" content—timeless posts that can be scheduled at any time of the year.

Vision for Scalable Community Outreach

Through this work, my goal has been to make social media engagement scalable, sustainable, and automated, so that maintainers can focus more on core development while still keeping the community informed and excited. This experience has helped me build a strong understanding of how community communication, technical contribution, and outreach intersect in successful open-source projects.

Project Details

Abstract:

This project aims to enhance the existing "Speak" activity on Sugar OLPC devices by integrating a hybrid conversational engine that supports both pattern-based AIML responses and dynamic replies via a large language model (LLM). Designed for children aged 5–12, the activity focuses on helping learners express themselves through both text and voice. Key features include invented spelling correction, voice-to-text input, guided conversations using persona-based prompts, emotional tone visualization, and safeguards for content filtering. All data is stored privately, ensuring a safe and child-friendly learning environment. The goal is to create an inclusive, encouraging, and interactive platform where children can practice language, engage in meaningful conversations, and grow in confidence and creativity.

Deliverables Overview:

1. Invented Spelling Support

- Display the original (invented) spelling typed by the learner.
- Show the suggested correction.
- Corrections are shown before sending input to LLM, so the model sees the improved version while the learner sees both.
- Visual cues:
  - Color-coded text.
  - Tooltip-like "Did you mean...?" pop up.
- Optional voice feedback reads both versions.

2. Hybrid Chat Engine (AIML + LLM)

- User input is analysed to check whether it is a pattern based question or complex question.
- If input is complex, it's preprocessed and sent to LLM or sent to the AIML engine.
- LLM responses are generated based on a persona template: friendly adult, encouraging, vocabulary-sensitive.

3. Conversation Mode Toggle

- Two Modes:
  - Learner-led: child types questions or thoughts → bot responds
  - Bot-led: bot initiates conversation with age-appropriate questions
- Useful for:
  - Encouraging shy users to engage
  - Practicing comprehension via Q&A

4. Input/Output Content Moderation

- Input analysis:
  - Flag or gently reject inappropriate language.
- Output analysis:
  - Postprocess LLM response to check for:
    - Inappropriate or overly complex phrases
    - Tone (angry, sad, confusing)
- Add safeguards to prevent unsafe or confusing replies using a filter.

5. Temporary Conversation Storage + Export

- Store session conversations locally (RAM or temp files).
- Allow export as:
  - Plain text
  - JSON for structured analysis
  - Optional Sugar Journal entries
- Conversations deleted or cleaned on exit to ensure privacy.

6. Persona Settings Panel

- Sliders/Controls to customize:
  - Warmth (how kind or formal the persona sounds)
  - Humour level (how playful and humorous persona sounds)
  - Simplicity (vocabulary difficulty)
- Preset Personas:
  - "Playful Teacher"
  - "Sugar Developer"
  - "Curious Friend"
- Settings saved for each session or user.

7. Chatbot Facial Expression (SVG-based)

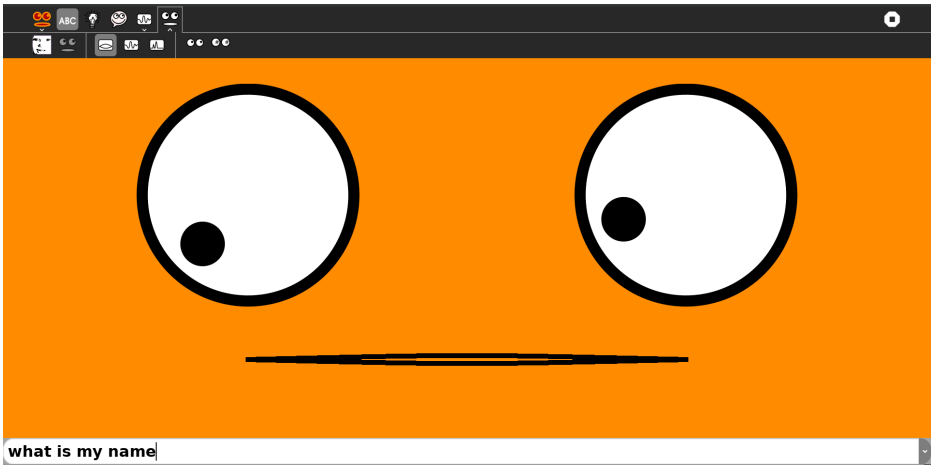
- Analyze the sentiment/tone of the bot's response.
- Show an appropriate face/emotion using animated SVGs:
- Makes the experience more human, more playful, and helps non-readers understand emotional tone.

8. Voice-to-Text Input

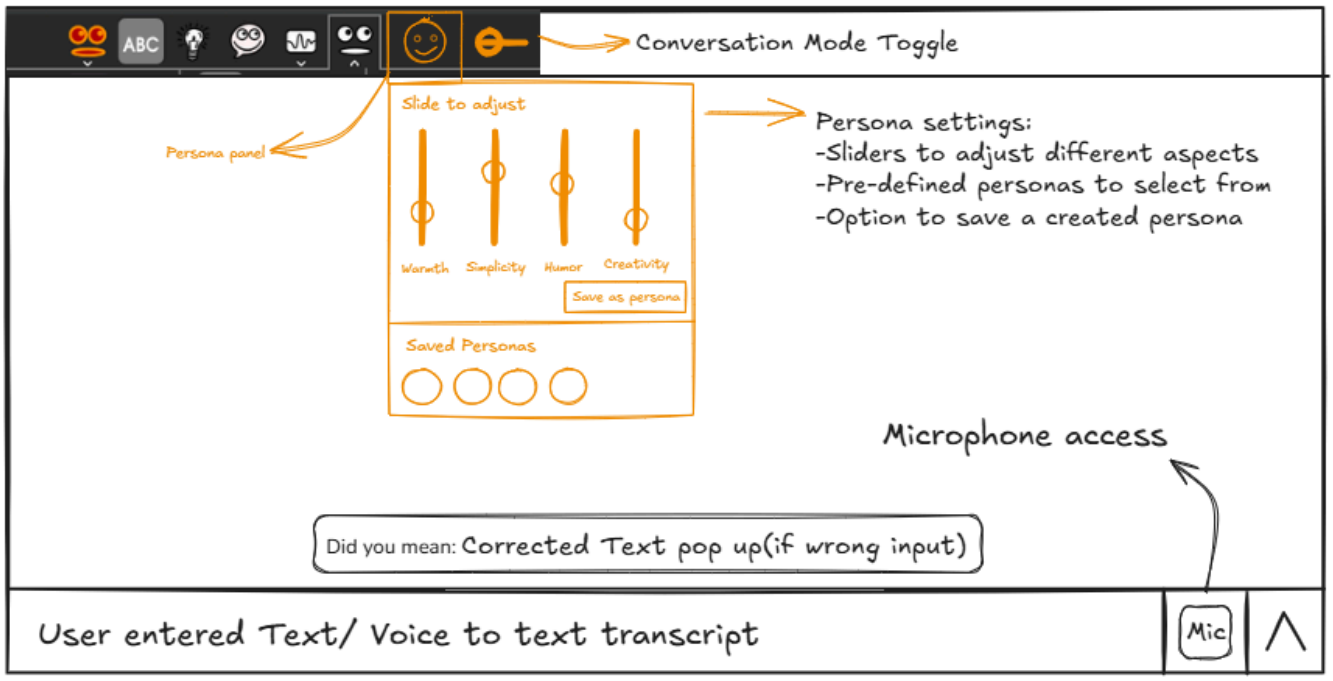
- Allows learners to speak instead of typing.
- Converts speech into text using offline or online speech recognition.
- Supports young learners who are still developing typing skills.

User Interface design overview(Chatbot Tab):

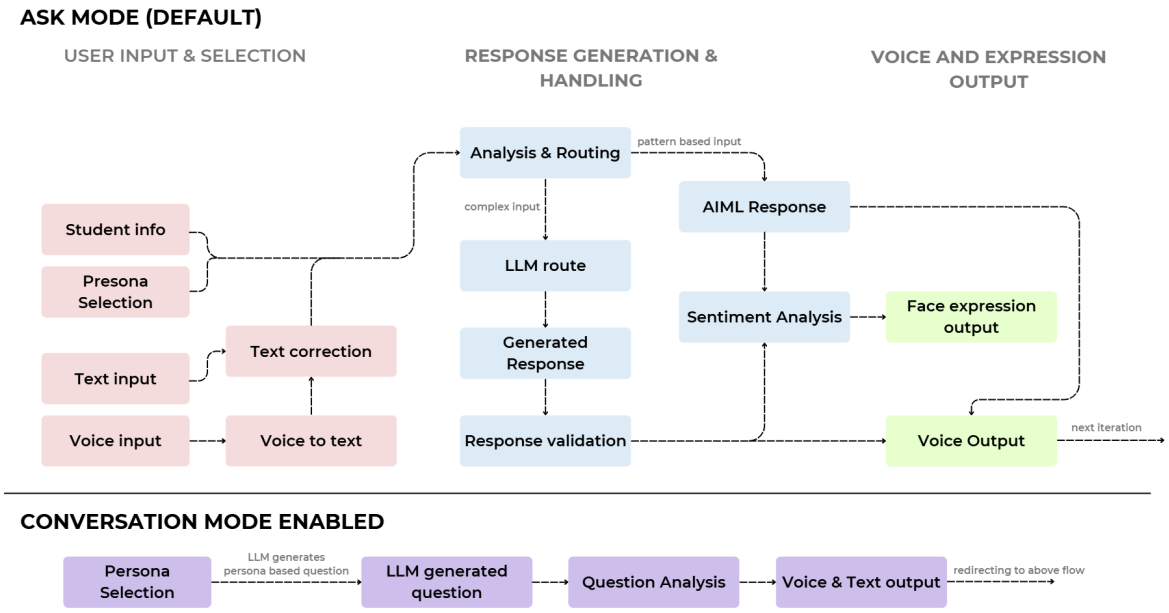
Existing UI:



Proposed Modification:



Chatbot Section Structure



Section 1: User Input & Selection

This section enables the student to interact with the activity using different input methods, choose personas, and initiate conversations. It consists of the following modules:

1.1 Student Info

- **Purpose**: Personalize the experience by identifying the user (name, age).
- **Data Handling**:
    - Extracted from info stored within sugar.
    - Used to adjust difficulty level of chatbot based on student age
    - **Tech Used**: Local storage via flat files on Sugar OS; no external server is required.

1.2 Persona Selection

- **Purpose**: Allows children to select from predefined or custom personas
- **Interface**:
    - Sliders for **Warmth**, **Humor**, **Simplicity**, and **Curiosity** to tune the chatbot's personality.
    - Persona presets can be saved and reused.

- **Tech Used**:
- Simple UI built using Sugar Toolkit (GTK3 or HTML5-based UI).
    - Slider values are stored in the session and influence LLM prompt generation when active.

1.3 Text Input

- **Purpose**: Learners can type words or sentences freely.
- **Spell Correction**:
    - Uses a **lightweight open-source spell checker** such as **SymSpell** or **Aspell**.
    - Shows:
        - Learner's original (invented) spelling.
        - Corrected version.
    - Visual cues like color coded text or tooltips ("Did you mean...?").
    - User can select the corrected text or go ahead with the original text
    - Filtering inappropriate words or language
    - Optional voice feedback reads both versions aloud.

- **Integration**:
    - Corrected version goes to the hybrid engine (AIML or LLM).
    - Learner sees both versions for learning and comparison.

1.4 Voice Input

- **Purpose**: Offers accessibility to non-readers or early learners.
- **Implementation**:
  Uses **Vosk** (offline, open-source STT engine).
    - Converts spoken input to text, followed by the same spell correction and routing process.
    - Handles basic speech in supported languages (initially one language, expandable).
- **UX**:
    - A visual mic button toggles recording.
    - **Captured text is editable by the learner before being sent for response generation.**

Section 2: Response Generation and Handling

2.1 Input Analysis & Routing

- **Purpose:** Decide whether to use AIML or LLM to generate the response.
- **Routing Logic:**
  - Pattern-based inputs (e.g., "what is your name?") go to the AIML engine.
  - Open-ended or unrecognized inputs are routed to the LLM.
- **Technologies:**
  - Python-based rule engine using regex or a basic classifier (scikit-learn).
- **Outcome**:
  - Ensures efficient use of resources by leveraging AIML when applicable.
  - Enables conversational, creative responses using LLM when needed.

2.2 AIML Response

- **Purpose**: Provide factual, structured answers based on known patterns.
- **Process**:
  - Input is matched against .aiml files.
  - Generates static responses for predictable, educational queries.
- **Technologies**:
  - Existing AIML engine
- **Integration**:
  - Fast response with minimal resource use.
  - Fall back mechanism for LLM response; works offline.

2.3 LLM Response

- **Purpose**: Handle open-ended queries, storytelling, and rich conversations.
- **Process**:
  - Input is sent to an LLM backend via API.
  - Response is generated based on selected persona and sliders (warmth, humor, etc.).
- **Technologies**:
  - Model: Open-source LLM like LLaMA 2, Mistral, Gemma which is fine tuned to give better student friendly response.
  - Deployment: AWS (SageMaker, EC2, or Lambda).
  - Serving: FastAPI
- **Integration**:
  - Response passed to sentiment and validation modules before output.
  - Supports future multi-language expansion.

2.4 Sentiment Analysis

- **Purpose**: Determine the emotional tone of the chatbot's response.
- **Process**:
    - Lightweight sentiment models label response as happy, curious, neutral, etc.
    - Label is used to select a chatbot's facial expression.
- **Technologies**:
    - VADER, TextBlob, or ONNX/TFLite-based BERT variants.
- **Integration**:
    - Not applied to every LLM response.
    - Enhances perseverance of generated response's tone.

2.5 Response Validation

- **Purpose**: Ensure safety and appropriateness of LLM responses.
- **Checks Performed**:
    - Inappropriate content (e.g., profanity).
    - Excessive length or irrelevant tangents.
- **Technologies**:
    - Regex-based filters.
    - Libraries like profanity-check or clean-text.
- **Integration**:
    - Applied optionally for specific prompts or sessions.
    - LLM responses failing validation can be replaced with a fallback reply or another response is generated from the LLM

Section 3: Output and Expression Handling

3.1 Text & Voice Output

- **Purpose**: Present the response in both written and spoken form for better engagement and language learning.
- **Process**:
  - Response (from AIML or LLM) is displayed as text.
  - Simultaneously read aloud using an existing voice synthesis engine.
- **Technologies**:
  - Uses existing TTS (Text-to-Speech) functionality from the original Speak activity
- **Integration**:
  - Seamless, as the activity already supports voice output.
  - Can be muted or repeated by the learner.

3.2 Expression Output

- **Purpose**: Visually convey the emotional tone of the response using facial expressions.
- **Process**:
  - Sentiment label (e.g., curious, happy, neutral) triggers a specific SVG face.
  - The SVG comes into action and persists for a few seconds after the response.
- **Technologies**:
  - Predefined set of SVG facial expressions.
  - Lightweight frontend update logic in Python layer.
- **Integration**:
  - Does not affect response content.
  - Helps build emotional context and engagement with the learner.

3.3 Chat History

- **Purpose**: Allow learners to revisit past conversations for reinforcement and reflection.
- **Process**:
  - Temporary session chat is stored locally during activity runtime.
  - Option to export chat as text file at the end of the session.
- **Technologies**:
  - Stored using JSON or simple text files.
- **Integration**:
  - Fully offline-compatible.
  - Preserves privacy as no data is sent externally.

3.4 Conversation Mode (Toggle)

- **Purpose**: Switch from passive reply mode to an engaging, chatbot-led conversation.
- **Process**:
    - Learner enables "Conversation Mode" via a toggle button.
    - Prompts selection of a persona (e.g., friendly teacher, playful robot).
    - LLM then starts the dialogue with an opening question.
- **Technologies**:
    - UI toggle in the activity interface.
    - Persona selection menu with predefined options.
- **Integration**:
    - Persona settings modify LLM behavior using prompt engineering or parameters.
    - Default mode is passive "Ask Mode."

3.5 Persona Customization

- **Purpose**: Let learners personalize the chatbot's tone and behavior.
- **Process**:
    - Sliders provided for warmth, humor, simplicity, and curiosity.
    - Adjusts how the LLM phrases its responses.
- **Technologies**:
    - UI sliders tied to LLM prompt tuning parameters.
    - Saved locally for reuse.
- **Integration**:
    - Persona state persists per session or per learner.

Text to Voice Section Structure

Invented Spelling in "Turn Text to Voice" Tab

- **Spell Check Integration**
  When a learner types a sentence, a lightweight spell checker (e.g., SymSpell) runs silently in the background.
- **Dual Output Display**
  Both the original text (invented spelling) and the corrected version are shown side by side before converting to voice(only if there are incorrect spellings detected).
- **Voice Playback Options**
  - Button to hear original input (helps with self-assessment).
  - Button to hear corrected version (supports learning proper pronunciation and structure).
- **Visual Cues**
  Corrected words are subtly highlighted to help children notice spelling differences.
- **Learning Impact**
  Reinforces vocabulary and pronunciation while respecting the child's early spelling attempts.

Technologies

- *Spell Correction*
  – SymSpell or Aspell (offline, lightweight)
- *Voice Input*
  – Vosk API (offline speech recognition)
- *Chat Engine*
  – Existing AIML engine (pattern-based)
  – LLM via API (e.g., AWS Lambda, SageMaker, or Hugging Face)
  – Models: LLaMA 2, Mistral, OpenChat(selection in research phase)
- *Emotion Mapping*
  – Lightweight classifier
  – SVG's
- *Storage*
  – Local JSON/text file (chat logs)
- *UI*
  – GTK (PyGObject – Sugar-compatible)

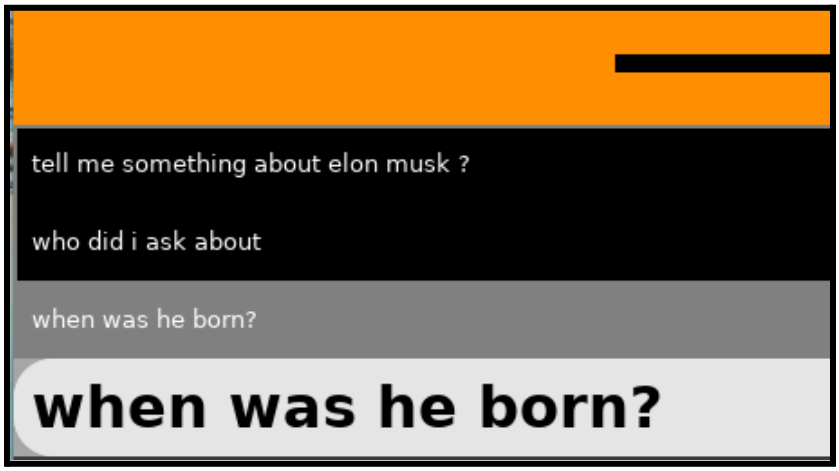
Demo Implementation of Chatbot Response

(This demo is to only verify LLM response within the activity)

- Implemented LLM chatbot response into the already existing response from the aiml engine.
- The code analyses and routes the input based on certain rules(Temporarily routing to LLM by default to verify its working)
- The LLM response is spoken by the chatbot as well as logged on the terminal.
- The response is stored temporarily and used as context for further input.

- **Video link of the demo:** [here]
- **Draft PR link:** [here]

Input from the user:



Output from the Chatbot:

```
sudo-aman@sugar:~/Activities/speak$ sugar-activity3

Loading brain from bot/alice.brn...done (40564 categories in 2.66 seconds)
Elon Musk is a billionaire entrepreneur and CEO of SpaceX and Tesla.
Elon Musk.
June 28, 1971.
```

As you can see the LLM response is seamlessly integrated into the activity. Response is concise and the LLM also has context of the previous messages.

Code:

```
88  def input_routing(text):
89      # Returning false for now to verify LLM response
90      return False
91
92
93  def respond(text):
94      if input_routing(text) is True:
95          # aiml response
96          if _kernel is not None:
97              text = _kernel.respond(text)
98          if _kernel is None or not text:
99              text = _("Sorry, I can't understand what you are asking about.")
100
101     else:
102         # LLM Response
103         llm_response = get_llm_response(text, context)
104         if llm_response:
105             print(llm_response)
106             context.append(f"You: {text}")
107             context.append(f"LLM: {llm_response}")
108             return llm_response
109         else:
110             return _("Sorry, I can't understand what you are asking about.")
111
```

Impact this project creates?

This project goes far beyond just code and functionality — it creates a safe, expressive, and engaging space for children to communicate, explore language, and feel heard.

For many learners, especially in under-resourced or multilingual settings, expressing thoughts confidently can be difficult. Spelling mistakes, hesitation in speech, or limited exposure to conversation often hold them back. With the hybrid chat engine, voice input, and carefully tuned personas, this project breaks those barriers. It gently corrects, encourages, and listens — without judgment.

Imagine a child who struggles with spelling but has a beautiful story to tell. This activity doesn't stop them — it helps them express that story, learn from it, and even hear it read back to them with warmth. It respects their effort, nurtures their curiosity, and builds a bridge between their ideas and the world.

The tone sliders — warmth, humor, simplicity, curiosity — aren't just design elements. They **reflect real values that every child deserves to experience when learning**: kindness, joy, understanding, and the freedom to ask "why" as many times as they want.

Personally, this matters to me because I've seen how small acts of support — a corrected word shown kindly, a question asked with empathy — can change how children see learning. They stop being afraid. They start playing, exploring, and expressing themselves. And that's what learning should feel like.

By blending simple technologies with thoughtful design, this project brings not just smarter learning — but more human learning. And in a world that often rushes children to perform, we are choosing instead to patiently listen and uplift. That's the real impact!

TIMELINE

Time Frame	Tasks
Pre-GSoC Period 8 April – 7 May	• Deep dive into the existing Speak Activity codebase and Sugar framework. • Explore how the current AIML engine is integrated. • Test the voice system and analyze how text input → voice flow is implemented. • Research about low resource LLM models for offline deployment to sugar ai. • Experiment with lightweight spell correctors (e.g., SymSpell, Hunspell).
Community Bonding Period 8 May – 1 June	• Interact with mentors and other contributors at Sugar Labs. • Clarify architectural doubts (esp. LLM-AIML hybrid flow). • Start blog posts documenting architecture insights. • Begin LLM persona design + slider mappings (warmth, humor, etc.). • Prototype UI layout for persona and modes.
Coding Phase 1 2 June – 22 June	• Set up the hybrid chat engine logic (AIML vs LLM decision flow). • Implement spell correction module in input pipeline. • Add visual feedback for corrected text. • Integrate offline voice-to-text input with spell check. • Ensure chat gets properly routed post-processing.

Coding Phase 2
23 June – 13 July
	• Implement LLM-based conversation mode:
	  ○ Persona selector + default personas
	  ○ Sliders for tone tuning
	  ○ LLM initiates age-appropriate questions
	• Build toggle for Ask Mode & Conversation Mode.
	• Add student info support (name, age) and persist locally.

Midterm Evaluation
14 July – 18 July
	• Demonstrate:
	  ○ Full input → hybrid engine → output cycle
	  ○ Conversation mode + persona sliders
	  ○ Spell correction with UI
	  ○ Voice input working.
	• Gather mentor feedback and iterate as needed.

Coding Phase 3 19 July – 9 Aug	• Implement output filters + analysis layer: • Length validation • Inappropriate content filtering • Emotion analysis (lightweight ML model) • Display chatbot emotion via SVG face expressions. • Hook this into every LLM output conditionally.
Final Coding Phase 10 Aug – 24 Aug	• Implement chat history export + temporary session storage. • Add text-to-speech enhancements using corrected spelling flow. • Finalize UI updates for better kid-friendly navigation. • Bug fixing + performance tuning.
Final Week 25 Aug – 1 Sept	• Write the final user and teacher guide. • Submit final GSoC deliverables. • Final walkthrough with mentors.
Post GSoC 2 Sept – 7 Nov	• Respond to community feedback and fix issues. • Write detailed developer documentation. • Discuss with mentors future enhancements (multi-language support, model deployment options, etc.).

Progress Report:

I'll keep the mentors updated daily via Matrix chat and showcase my progress during biweekly meetings. Additionally, I'll publish a blog post every alternate week on Medium to document my journey, which I'll also share across my social media channels.

Post GSoC Plan:

Even after the official GSoC period ends, I truly want to stay connected with the **Sugar Labs community**. This project isn't just a summer commitment for me—it's something I care deeply about, especially because it impacts how children **learn and express themselves**. I've learned so much from the mentors, the **collaborative spirit**, and the **mission of building accessible tools for education**. I plan to remain involved in **community discussions**, continue improving the **Speak activity**, and help **review or mentor** future contributors if possible. Staying in touch with the people who made this experience meaningful is important to me—not just as a developer, but as a **learner too**.

Beyond this project, I want to explore more areas within Sugar Labs and contribute to other **educational tools** that empower young learners. I see myself diving deeper into **child-computer interaction**, experimenting with **AI to enhance learning**, and building fun, meaningful features that keep kids curious. I also hope to write **technical blogs**, create **onboarding guides**, and be a **reliable voice in the community** that helps other contributors feel welcomed—just like I was. Additionally, I plan to **actively manage Sugar Labs' social media content and postings**, ensuring that community updates, events, and stories continue to reach a broader audience and inspire more contributors. This journey with Sugar Labs is just beginning for me—**I want to grow with it**.

Conclusion:

This project is more than just a technical challenge for me—it's a chance to build something that truly matters for children. My background, passion for educational technology, and commitment to open-source make me a strong fit for this work. I deeply resonate with the values of Sugar Labs and believe in creating tools that are joyful, accessible, and meaningful for young learners. With a clear plan, dedication to the community, and a heart for impact, I'm excited to bring this vision to life and grow alongside this project.