

Educational Chatbot for Second Language Acquisition Among Children

Aizhar Kudenova, Kamila Spanova, Moldir Azhimukhanbet, Rassul Mukhambetkali,
Sanzhar Mazhit

Department of Computer Science
Nazarbayev University, Astana, Kazakhstan
Advisor: Prof. Siamac Fazli

Abstract

We present *Boxy*, an educational chatbot system that helps kids between the ages of three and ten with learning Russian as a second language. By providing a judgment-free chat companion and gamified learning modules, the system addresses the well-known “affective filter” barrier [1]. The system features a real-time Speech-to-Text $\rightarrow$ Large Language Model $\rightarrow$ Text-to-Speech (STT $\rightarrow$ LLM $\rightarrow$ TTS) pipeline using Whisper, Gemma-4-E2B (Q4_K_M), and Silero v5. It is fully deployed on an NVIDIA Jetson Orin Nano, with the speech-to-text and large language model inferences running on the GPU and the TTS synthesis on the CPU, achieving a latency of under 3 seconds. An adaptive level-identification game accurately determines CEFR A1 or A2 levels by utilizing semantic embeddings, and an asynchronous memory module supported by a local SQLite database personalizes each learning session. Three educational games were designed to assist with early grammar and vocabulary. In compliance with GDPR [2] and COPPA [3] standards, all data processing is done on-device. The models were carefully selected by systematic benchmarking on ten LLMs, six STT models and three TTS engines. As the prototype demonstrates, a single edge device with limited resources can provide private and low-latency second language education based on real teaching principles.

Keywords

second language acquisition, educational chatbot, edge AI, speech-to-text, large language model, text-to-speech, gamification, children, Jetson Orin Nano, RAG

I. EXECUTIVE SUMMARY

Children between the ages of 3 and 10 are in the most important linguistic window for learning a second language [4]. And most of the available technologies are limited to only actively tapping the screen, which does not support natural language practice, or sends sensitive audio data to a public cloud-based service. *Boxy* solves this problem as a combination of a real-time voice pipeline, local storage, and adaptive games specifically designed to help young Russian language learners.

OpenAI Whisper in the pipeline transliterates the child’s speech, 4-bit quantized Gemma-4-E2B responds based on language teaching principles, and Silero v5 synthesizes a Russian audio with a natural sound. The SQLite database is used to store each child’s name, their preferences, and the vocabulary learned at the end of each lesson. The level determination game aims to place each learner at A1 or A2 CEFR level and guide them to relevant educational games. Vocabulary, word form adaptation, and spatial thinking are taught in three games: *Animal Recognition Game*, *Colors & Numbers*, and *Word Positions*.

Over the last two semesters, our team has compared ten LLM, six STT, and three TS models and tested them on three devices: Raspberry Pi, Orange Pi, NVIDIA Jetson Orin Nano. We developed a demo-ready prototype that meets all functional and technical requirements.

II. INTRODUCTION

A. Problem Statement

In early childhood, the acquisition of the second language is limited because of the “affective filter” defined by linguist Stephen Krashen [1]: anxiety, self-awareness, and fear of judgment create a psycho-

logical barrier and hinder active language production. Children often learn a language passively but are never able to speak it because the social pressure of speaking in front of peers or teachers is too high for the risk of making mistakes. In Kazakhstan, where proficiency in the Russian language is almost essential, the barrier is particularly acute among young children whose native language is either Kazakh or English.

Current technological solutions are not enough. Passive mobile apps like Duolingo are interactive apps with less active voice production that almost entirely rely on touch. General-purpose smart speakers like Amazon Echo or Google Home don't have child-safe content filtering and structured language learning programs. When users are children under the age of thirteen, the main privacy issue is the local processing of voice data, which is not offered in either of the solutions.

B. Proposed Solution

We propose *Boxy*, a physical educational chatbot device that acts as a judgement-free conversation partner for Russian L2 learners aged 3-10. The entire system runs on a single NVIDIA Jetson Orin Nano: STT and LLM inference execute on the onboard GPU, while TTS synthesis runs on the Jetson CPU - no external server or internet connection is required. The "Listen-Think-Speak" pipeline - Whisper STT, Gemma-4-E2B LLM, Silero v5 TTS - operates fully on-device alongside a touchscreen, microphone, and speaker. An adaptive game suite reinforces vocabulary and early grammar through play rather than drill.

C. Report Organisation

Section III identifies project constraints. Section IV reviews related work. Section V describes the full system design. Section VI narrates the two-semester development journey. Section VII presents benchmarking and qualitative results. Section VIII concludes with future directions.

III. IDENTIFICATION OF CONSTRAINTS

Financial. This project was implemented without external funding. Each model of was open-source and hardware was limited to the tools available in the laboratory. By making specific selections, component costs (Jetson Orin Nano, display, microphone, and 3D printed box) were reduced.

Technical. Edge hardware has hard drive and computational limitations. Models, with demand higher than the available VRAM, needed quantization, which compromises quality in favor of practicality. Because of the limited memory and three models running in parallel, TTS is running on CPU. Jetson Orin Nano was preferred to the Orange Pi due to latter's streaming instability.

Ethical. AI-powered language models are capable of producing "hallucinations" or clearly visible misinformation. Children who use it are particularly at risk. To reduce this, we set the LLM temperature to 0.2 when children engage in play and the system simply directs them toward educational content.

Privacy. Users are children under the age of thirteen. Transcripts, audio data, and personal data cannot be transferred to a third-party cloud API or another server. In accordance with COPPA [3] and GDPR [2], all processing - STT, LLM, TS, and storage - is fully retained by Jetson Orin Nano.

Social. The system is designed to *supplement* rather than replace human instructions. The risk of dependency is reduced by limiting the structure of sessions.

Health and Safety. When designing the duration of classes, the screen time recommendations for children aged three to ten years were taken into account. Maximum audio output is capped within safe levels. The box prevents children from accessing the computer itself.

Sustainability. Without the need for a server, all inferences can be made on a single Jetson device, which reduces overall power consumption and hardware costs due to its small size and insignificant battery usage. The system is flexible enough to be used in low-resource settings.

Scalability. The database layout and pipeline interfaces are language-neutral. Other languages (such as Kazakh) and higher CEFR levels can be added without changes to the architecture.

IV. BACKGROUND AND RELATED WORK

A. Second language acquisition

According to Vygotsky’s zone of proximal development (ZPD), students learn best when they engage with a “slightly more knowledgeable other” [4]. This role is adopted by the AI in *Boxy*, which adjusts the response complexity to the child’s rated level. According to Krashen’s input hypothesis [1], to suppress the “affective filter”, the learning environment must be stress-free, and material must be at the “i+1” level: still understandable, yet challenging. We chose the voice-first approach because Long’s interaction hypothesis [5] suggests that the learning process requires active conversations, not passive listening.

B. Gamification in Education

According to a meta-analysis by Hamari et al., gamification improves internal motivation by offering quick feedback, goal setting, and incentive systems [6]. The Csikszentmihalyi’s “flow” [7] inspired the optimal mutual-adaptive difficulty system, which was achieved when the difficulty matched the level of mastery. The level identification game follows the current assessment of skill and blocks the more difficult content until the learner is ready, ensuring efficient experience.

C. Memory Systems for LLMs

Standard LLMs’ context window is restricted. Early content will be removed as the conversation progresses. Packer et al. proposed MemGPT [8], a hierarchical memory architecture that changes the contexts between archive storage (database) and main memory (active window). We discovered a phenomenon similar to “lost in the middle” [9] in a system set up during our fall semester, where models confuse the details when the context is too dense. This led to reconstructing the conversation using lightweight RAG system and injecting the game states into prompts to give hints accounting for game context.

D. Speech Recognition

Early offline STT systems such as Vosk relied on Hidden Markov Model–Gaussian Mixture Model (HMM-GMM) architectures. While computationally efficient, our empirical tests showed Vosk achieved a Word Error Rate (WER) above 30% on children’s Russian speech, particularly failing on vowel reduction patterns. Radford et al.’s Whisper [10], a sequence-to-sequence Transformer trained on 680,000 hours of multilingual audio, demonstrated significantly superior zero-shot multilingual recognition, reducing WER below 5% in our tests.

E. LLM Quantisation

Running a LLM with billions of parameters on an edge hardware requires aggressive compression. GGUF Q4_K_M quantization [11], a 4-bit K-quant scheme, reduces VRAM consumption by approximately 63% with an 11.9% accuracy trade-off. This technique enabled us to run Gemma-4-E2B on the Jetson Orin Nano’s onboard GPU without exceeding its memory budget.

F. Text-to-Speech for Children

The naturalness of the TTS is especially important because the child audience is sensitive to robotic or artificial prosody (Terre Cotte et al., 2017) [12]. We evaluated the latency and Russian authenticity of XTTS v2, Piper, and Silero v5. The most balanced in intelligibility, natural prosody, and inference speed was the Silero v5 [13].

G. Comparison with Existing Systems

Table I on the six aspects of children’s L2 education compares *Boxy* with general current approaches.

TABLE I
COMPARATIVE ANALYSIS WITH EXISTING EDUCATIONAL SYSTEMS

Feature	<i>Duolingo</i>	<i>Amazon Echo</i>	<i>Google Home</i>	<i>Prior Work [14]</i>	<i>Boxy (ours)</i>
Voice interaction	No	Yes	Yes	Yes	Yes
Local processing	No	No	No	Partial	Yes
Privacy protection	Low	Low	Low	Med	High
Adaptive difficulty	Partial	No	No	No	Yes
L2 games	Partial	No	No	No	Yes
Child-safe content	Yes	No	No	Yes	Yes
Internet dependency	High	High	High	Hybrid	None
Cost (hardware)	\$0	\$50	\$50	\$85	\$200
Latency	N/A	<1s	<1s	2.77s	<1s*

*Perceived latency to first audio sentence via chunked streaming.

V. PROJECT APPROACH

A. System Architecture Overview

Boxy is a fully independent edge device. The entire system is hosted on NVIDIA Jetson Orin Nano; the STT and LLM models run on the Jetson GPU via `llama.cpp` and `faster-whisper` for inference with CUDA acceleration, while the TTS model (Silero v5) runs on the Jetson CPU in a dedicated background thread. The touchscreen display and USB microphone are connected directly to the Jetson. No network connection, external server, or cloud API are required at runtime for complete task execution. All voice data, transcripts, LLM responds, and user progress remain on-device at all times.

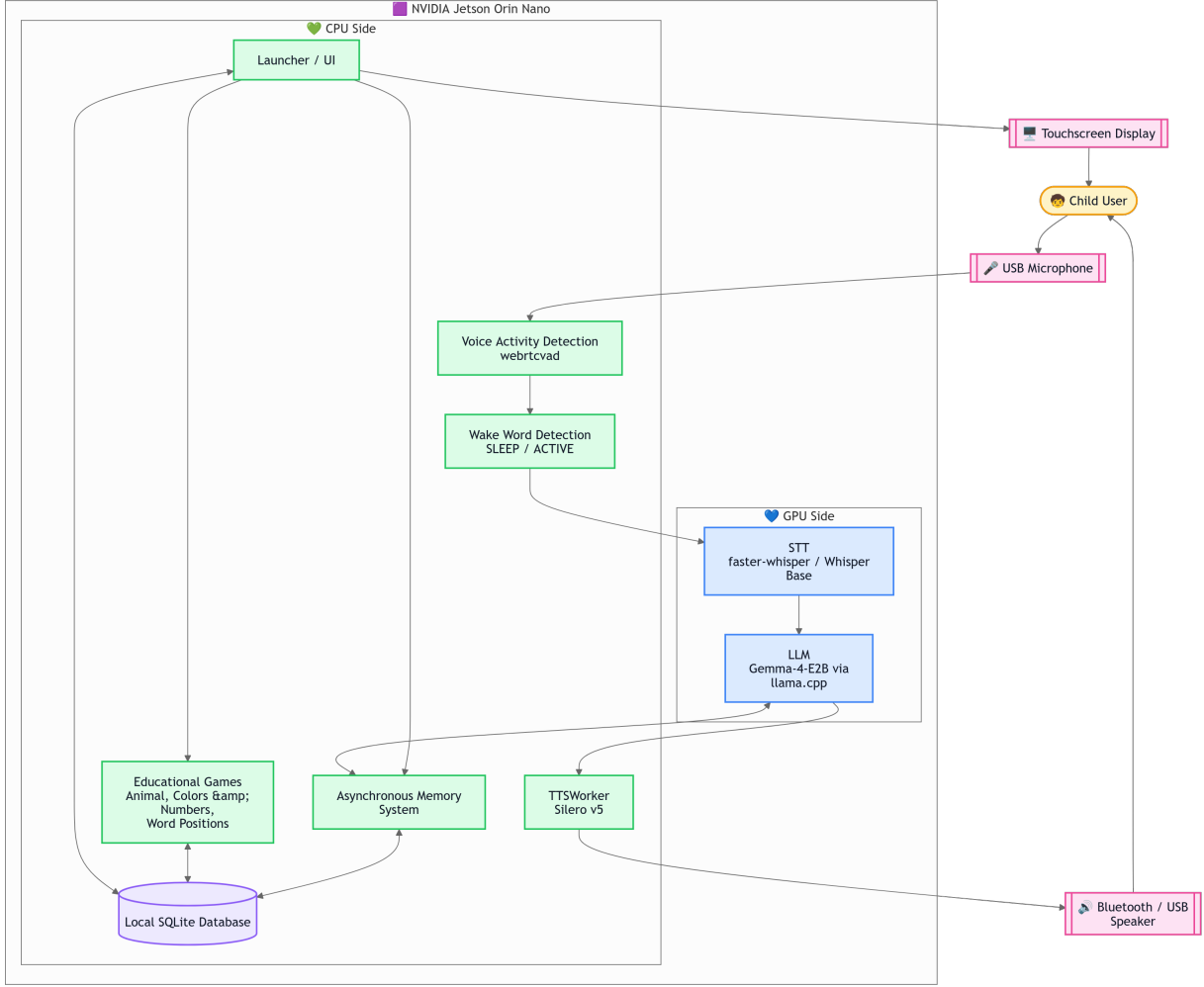


Fig. 1. System architecture of *Boxy*. All components - STT and LLM on the Jetson GPU, TTS on the Jetson CPU, games, memory, and launcher - run on a single NVIDIA Jetson Orin Nano.

B. STT→LLM→TTS Pipeline

The “Listen–Think–Speak” pipeline operates fully on the Jetson Orin Nano as follows:

(1) Voice Activity Detection (VAD). A continuous microphone stream is monitored with the use of `webrtcvad` on the device CPU. Audio frames are buffered only when actual speech is detected and discarded otherwise, eliminating unnecessary GPU inference calls.

(2) Wake Word Detection. The system idles in a `SLEEP` state until it detects the wake-words: “hey boxy”, “privet boxy”, “box box”, and “hi boxy”. After detecting one of the wake-words, it enters an `ACTIVE` state, processing every subsequent utterance until a timeout indicating silence or an explicit “sleep” command is received.

(3) Speech-to-Text. Buffered audio is passed to `faster-whisper` running multilingual Whisper Tiny in an adaptive way that it prioritizes the GPU as it is available and otherwise falls onto the Jetson CPU, returning a transcript within approximately 300 ms per full sentence of 10 words.

(4) LLM Inference. The transcript is injected into a structured prompt and processed by Gemma-4-E2B (Q4_K_M) via `llama.cpp` on the device GPU, following chat-template. The system prompt is kept stable across turns (KV-cache optimisation) while RAG context is injected into the user prompt only, avoiding expensive re-encoding of static instructions on every turn. The LLM streams tokens; as soon as an EOS (end of the sentence) is detected (within period, question mark, or exclamation mark), that sentence is immediately dispatched to the TTS thread - eliminating the waiting for the completion of generating the full response.

(5) **Text-to-Speech.** A dedicated `TTSWorker` thread running on the Jetson CPU processes each sentence using Silero v5 [13]. Text is preprocessed and any English words are transliterated to Cyrillic before synthesis. Audio is played back immediately via `aplay` as each sentence is synthesised, creating seamless streaming of audio chunks for better perceived latency. While the remaining sentences are still being synthesized in the background, the user listens to the beginning of the text to eliminate delays in communication.

The sequence diagram in Fig. 2 illustrates the full on-device interaction flow.

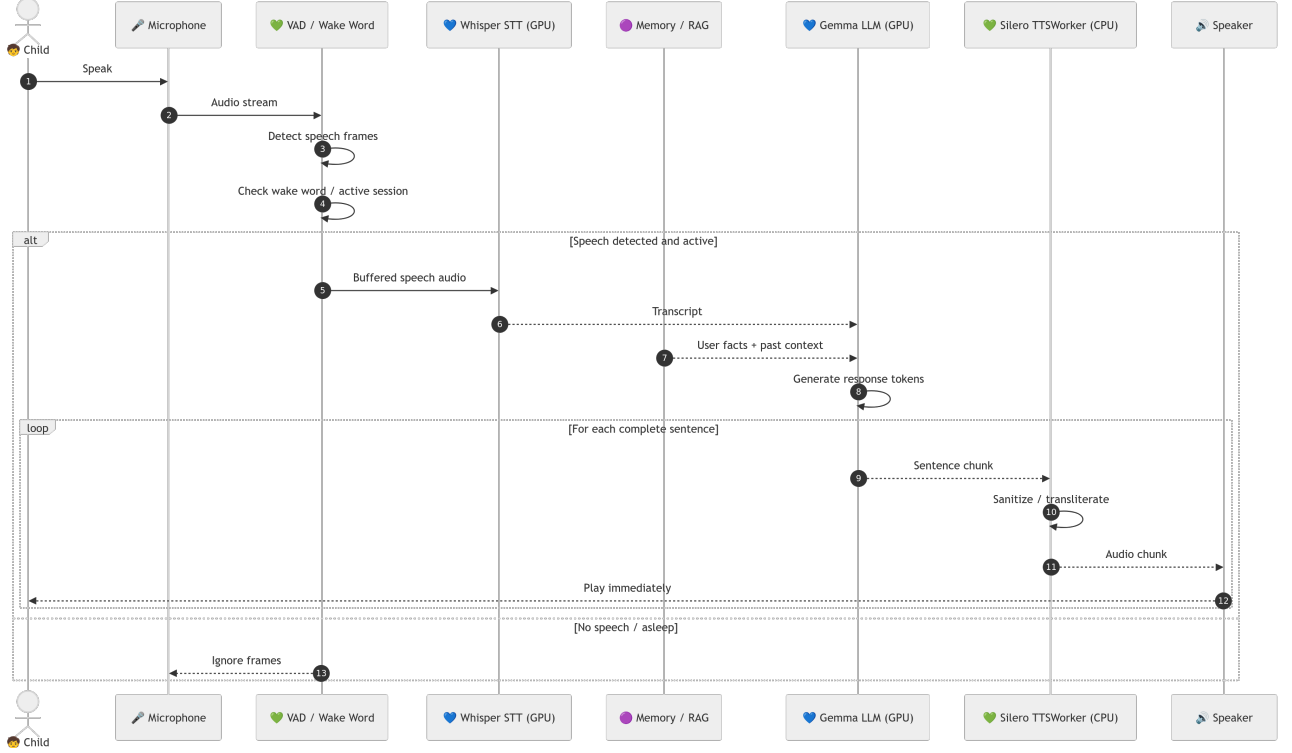


Fig. 2. Sequence diagram of the Listen–Think–Speak pipeline running entirely on the Jetson Orin Nano. Chunked TTS streaming eliminates the wait for full LLM response generation.

C. Asynchronous Memory System

The assistant memory runs in a silent background thread (`_async_memory_ops`) that never interrupts the conversation. It operates at three tiers:

Core Memory. Regular expressions continuously scan conversation context for personal facts (e.g., name, age, favourite activities, family members). The system detects personal facts and writes it to a local SQLite `users` table, then integrates it into the system prompt at the start of each session, giving the assistant persistent knowledge of the child.

Vocabulary Tracking. When the LLM provides a Russian word translation in response to a query (e.g., “how do you say apple”), the system detects the pattern and records the word in the `user_word_progress` table. This continuously builds a child lexicon of taught vocabulary.

Conversational RAG. Every conversation turn is stored in SQLite. Before each LLM call, a fast search over full text retrieves the most relevant past exchanges and injects them into the user prompt, providing contextual continuity across sessions. Reaching memory limits, old conversations are compressed to maintain database performance.

D. Adaptive Level Identification Game

Before accessing the games, every new child completes a diagnostic assessment on the current level of knowledge of the new preferred language. The diagnostic assessment is implemented as a :

- Loads vocabulary topics from the project database, covering A1 and A2 CEFR levels.
- Presents multiple-choice questions in both directions, i.e. Russian $\rightarrow$ English and English $\rightarrow$ Russian.
- Maintains a continuous *skill* estimation which rises with the number of correct answers and falls with incorrect ones.
- Eliminates A2 vocabulary until the child demonstrates consistent A1 proficiency.
- Stops automatically when the level estimation stabilises, avoiding false and unnecessarily long assessments.

LLM-driven question selection. When the local Ollama service (running `qwen2.5`) is available, it dynamically selects the next word based on the child’s current accuracy and tests it, while ensuring that no word is tested twice.

Semantic distractors. Wrong answer choices are generated using a local `bge-m3` embedding model. Cosine similarity between the candidate words is used to select distractors among answer choices. When the skill of the child is estimated as high, those distractors are semantically close (“apple” vs. “banana”); for low skill, words are unrelated (“apple” vs. “car”). This makes the assessment adaptive rather than uniformly easy or hard.

Output. On completion, the game writes a level profile and a word database in `word_knowledge.json` file recording known and unknown words. This file is considered by all subsequent game modules to avoid re-teaching mastered vocabulary.

E. Educational Games

Table II provides a brief overview of the three games. After testing the level, the launcher sends children to the A1 game section (*Animal Recognition Game + Colors & Numbers*) and children at A1 level to the A2 center at A2 level (*Word Positions + Colors & Numbers*).

TABLE II
SUMMARY OF EDUCATIONAL GAMES

Game	Level	Skills	Hints	AI
Animal Game	A1	Vocab, audio-picture matching	Yes	Yes
Colors & Numbers	A1/A2	Color/number vocab, morphological agreement	Yes	Yes
Word Positions	A2	Spatial prepositions, sentence comprehension	Yes	Yes

1) *Animal Recognition Game*: Six animals (cat, dog, pig, bird, horse, mouse) are available in two modes. In the Learn mode, the child touches hidden areas to see animals and hear their names in Russian. This helps them remember them by associating objects with other objects. In the play mode, the system prompts one animal at a time via audio prompt. In the **Play mode**, the system asks for one animal at a time via an audio prompt. The child then touches the correct picture. Positive feedback is given for correct choices, while incorrect choices are marked and the correct animal is shown. The score is kept throughout the entire run.

2) *Colors & Numbers*: This game generates a small scene with coloured objects and asks either a colour or a count question. Because it deals with Russian grammatical agreement, it is harder than the Animal Recognition Game. For example, colour adjectives must match the gender of the noun, and count answers depend on the grammatical class of the noun. The game uses the `word_knowledge.json` database to find words that the child hasn't learned yet and words that the child has already learned, so it changes to fit each child's needs. Sessions run for 10 rounds.

3) *Word Positions*: This is the most advanced game, targeting A2 learners. There are six objects on a canvas: a table, a chair, a cup, a box, a ball and a book. The system reads the Russian spatial instructions aloud and the child has to move the objects to make them happy. Supported relations include *on*, *under*, *left of*, *right of*, *inside*, and *between* are examples of supported relationships. In the multiple constraint scenarios, you have to deal with two or three spatial relationships at the same time. Answer checking uses tolerance-based geometry, which means it accepts close placement. This is good for younger children. The game progresses through a series of planned scenarios that become increasingly difficult as you progress.

F. Hint System

Each game has a "Hint" button that activates the LLM hint mechanism. The model receives a structured query that includes the game type, the current question, the number of previous attempts, and the child's incorrect answers in previous attempts. The model is told to speak only in Russian, keep it short, and help the child think, instead of giving a direct answer. TTS also reads the hint given.

An **answer-leak prevention** layer is reworked after each hint is sent. Forbidden words and patterns related to the correct answer are detected using regular expressions. If a violation is found, the system substitutes a safe pre-written fallback hint. In games where you have to guess colors, you cannot use any color words. Each hint usage is recorded in the `game_events` table for session analysis.

G. Launcher and User Interface

The Launcher controls the game hub shell. With a smooth gradient cartoon background, a large, rounded core element, a banner-shaped title, high, depth buttons, and rabbit figures everywhere, it is engl. The entire design is more of an educational program for children than a boring assistant screen.

First-time flow. The home screen will initially display a welcome page with a large image of *educational games in Russian*, a brief statement that the system must first determine the student's level, and an information card indicating the test procedure (words A1 and A2, non-repetition, automatic hub routing). In addition to the main action button, there is an additional *Chat with Boxy* button, *Start Level Test*. The Launcher procedure checks the saved result and immediately after the end of the diagnostic game in its window directs the tear to the corresponding hub.

Returning-user flow. The home page then becomes a dashboard that can be customized to each user profile. Level-based games, a visible progress bar, and a brief reminder that the student's current level badge (A1 or A2) should be ready will be displayed. *Play Level A1/A2 Games*, *Retake*, and *Boxy* three action buttons are presented. The cute rabbit mascot completes the UI and *Exit* button located below the main panel.

Level hubs. The A1 hub presents two game cards side by side - *Animal Game* and *Colors & Numbers* - each with a bunny graphic, a short description, and a *Play!* button. The A2 hub uses the same card layout but shows *Word Positions* and *Colors & Numbers*. A *Home* button at the bottom returns to the dashboard. The hub layout is designed to feel like a curated game shelf rather than a list of options.

Waiting screens. The launcher shows an animated 3-point loader, moves to the holding screen, waits for the *game* to run... or a *level test* in the second window, or a request to release an external window to continue launching when another *game* starts.

Boxy voice chat page. *Boxy Voice Chat* chat has three control buttons (*Start Boxy*, *Stop Boxy*, and *Back*), a card that provides a step-by-step guide to using Boxy, and a status tablet that shows that Boxy is

running. Before starting any game, the launcher automatically stops the boxing process to prevent resource conflicts between audio and microphone.

The UML usage diagram in 3 summarizes the interactions between actors. Toolbar for returning users sec 4. Centers A1 and A2 are shown in figs. Figs. 5 and 5, respectively.

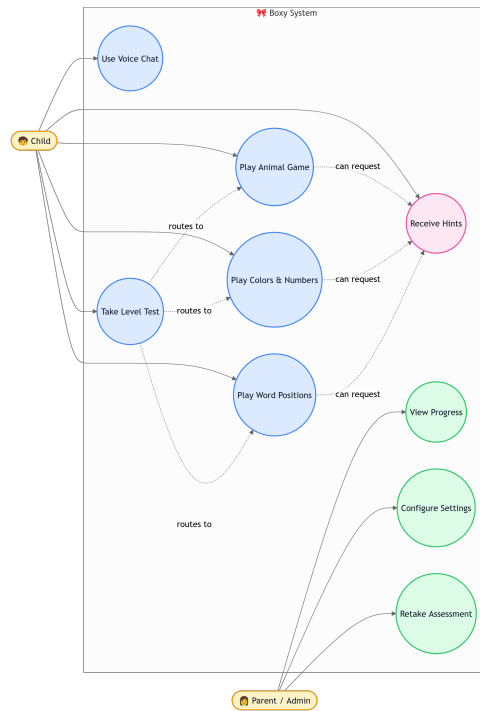


Fig. 3. UML Use Case Diagram. The Child interacts with all game and voice features; the Parent/Admin can configure settings and view progress.

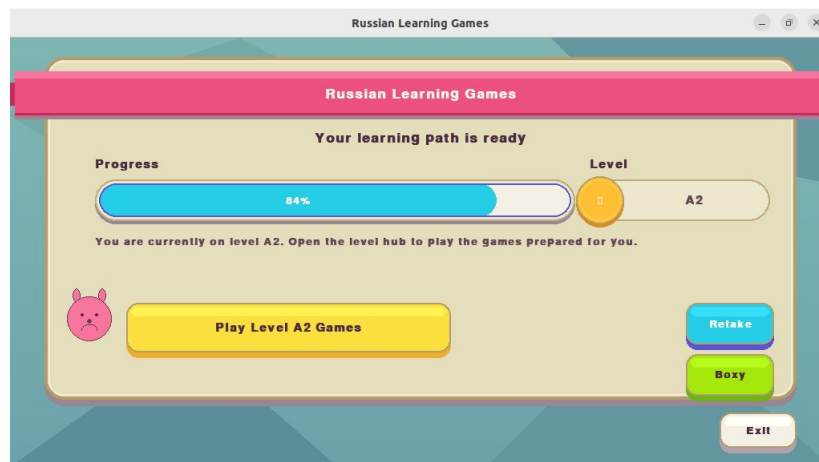


Fig. 4. Launcher main page - returning user view - showing level badge (A1/A2), progress bar, and the Play, Retake, and Boxy action buttons.

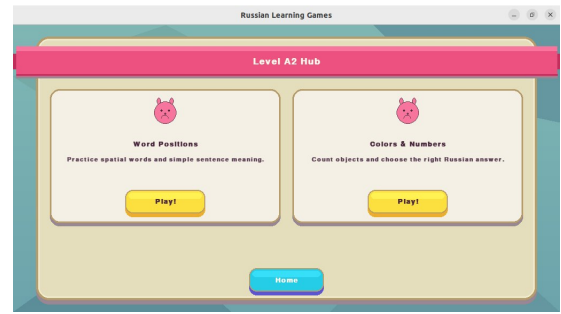
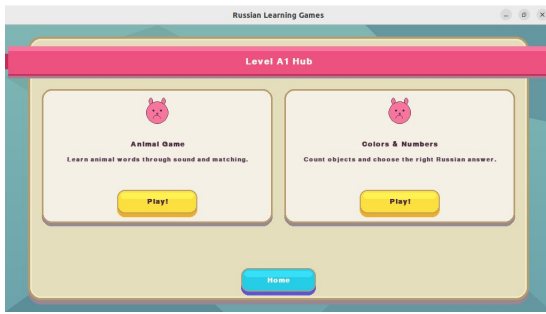


Fig. 5. Left: A1 Hub showing Animal Game and Colors & Numbers cards. Right: A2 Hub showing Word Positions and Colors & Numbers cards. Each card includes a bunny graphic, description, and Play! button.

H. Database Schema

Everything is stored in the local SQLite database. `users` (children's profiles), `topics` and `words` (a dictionary broken down by topic and CEFR level), `word_translations` (Russian-English maps), `word_forms` (grammatical variations of Russian morphemes), `user_word_progress` (learning status per child, number of attempts, accuracy), `games` and `game\events` (full session logs with tips and attempts) are some notable tables. The gadget will not lose any data.

I. Physical Hardware Enclosure

At Nazarbayev University, the application was created using 3D printing using SAPR and PLA. The beige box has cutout for LCD touchscreen monitor, along with HDMI and USB Type-C connections. The rectangular holes on the upper cover allows heat to escape. Power supply, DisplayPort, microphone USB of the Jetson Orin Nano and other USBs for connecting to the display are torn from the back panel. No exposed wires or sharp edges, and the design prioritizes child safety.

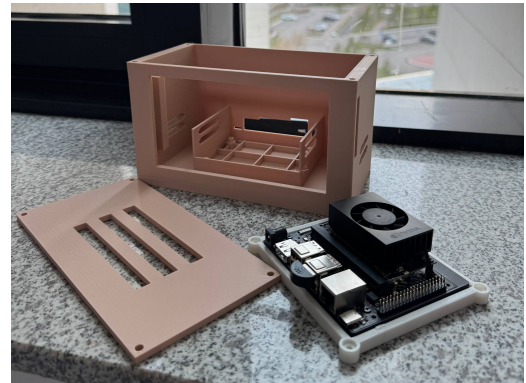
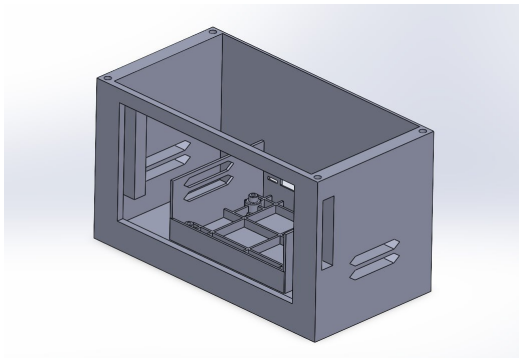


Fig. 6. Left: CAD design of the educational chatbot enclosure showing component layout and ventilation aperture. Right: Completed 3D-printed prototype in PLA before final assembly.



Fig. 7. Fully assembled *Boxy* device showing the front view: LCD touchscreen displaying the Animal game, with the NVIDIA Jetson Orin Nano, USB microphone, and speaker.



Fig. 8. Rear view of the *Boxy* enclosure showing the Jetson Orin Nano module, power supply connector, DisplayPort, microphone USB, and display USB cutouts on the back panel.

VI. PROJECT EXECUTION

A. Team Roles and Responsibilities

Five team members performed the following tasks, and all of them participated in the overall logical design and benchmarking of the system:

- **Aizhar Kudenova:** STT→LLM→TTS pipeline (LLM streaming, TTS streaming, VAD), implementation and design of Animal Recognition Game and Word Positions game, all UI design.

- **Kamila Spanova:** VAD implementation, audio streaming, tests with model selection, `llama.cpp` integration, animal recognition game, acquisition and production of real box components.
- **Rassul Mukhambetkali:** Pipeline support, Animal Recognition Game, Word Position game, 3D printing, and CAD protection design.
- **Sanzhar Mazhit:** Hint system, database integration, game of Colors and Numbers, installation and configuration of Jetson OS.
- **Moldir Azhimukhanbet:** Memory system, level determination game, and system integration (integrating pipeline, memory, games, and level identification into a launcher)

B. Fall 2025: Foundations and Pivots

Infrastructure setup. The team began by installing a typical version of Python and CUDA in the Docker container development environment. Code quality is guaranteed by GitHub affiliate storage policy. Trello was used to track two-week fast sprints over two semesters.

OpenAI API prototype. The first working prototype used OpenAI’s Whisper API for STT and GPT3.5-turbo for the LLM, connected to a Raspberry Pi 5. The overall latency was approximately 2 seconds, validating the pipeline concept. However, by transferring children’s voice data to foreign servers, data sovereignty standards were violated, necessitating a rapid transition to locally produced models.

Local model migration. The Whisper Tiny and Llama 2 were locally trained on the GPU development server. This proved that it is not technically impossible to host locally. Early tests showed that the LLM had no memory retention between different conversations, which motivated the memory system design.

MemGPT implementation and failure. Permanent memory is represented by a recursive generalization system modeled by MemGPT [8]. Call history is generated by LLM in JSON format, updated at each turn. Two tragic failures emerged: (1) the Raspberry Pi 5 CPU slowed down significantly by four to six seconds due to the context paging, (2) During the re-injection, the facts were hallucinated in a densely compressed context, known as “lost in the middle” phenomena [9]. The lightweight SQLite-based RAG system replaced the role of MemGPT.

Vosk STT failure. Vosk (HMM-GMM architecture) used to evaluate Raspberry Pi using fully offline STT. WER exceeded 30% on Russian speech. Whisper Tiny on the GPU reduced WER to below 5%.

Personality Engine exploration. Personality system was prototyped inspired by TARS, injecting descriptors (Humor, Honesty, Helpfulness, Sass, Curiosity, each 0–100%) directly into the LLM system prompt. The longer system prompt delayed LLM inference latency. This feature was eliminated in the Spring to meet a target delay of three seconds.

Gamification pivot. In the initial design, an open, free-form conversation was used as the primary method of conversation. An unofficial test and a review of the literature [6] showed that structured games produce better results in vocabulary. The team redesigned the product to track the game through local DB and injecting the game states into prompts.

Fall deliverable. By late fall 2025, the working audio pipeline (whispering + Llama 2 + early TTS) run on a development GPU server, a Raspberry Pi thin client for audio I/O, with a prototype memory system. The final hardware, games, and launcher were planned to be implemented in spring.



Fig. 9. Fall 2025 prototype: Raspberry Pi 5 thin client with USB microphone and portable speaker, used as the initial audio I/O device before migration to the Jetson Orin Nano.

C. Spring 2026: Integration and Delivery

Phase 1 - Game Logic (Weeks 1–4). The level identification game, Animal Game, and scaffolding for Colors & Numbers and Word Positions were implemented.

Phase 2 - Hardware migration (Weeks 3–6). The Raspberry Pi 5 was abandoned because even small quantized models were running too slowly to provide real-time interaction due to poor RAM and CPU processing. Orange Pi was used to test TTS using an RKNN/NPU accelerator. However, the audio stream was unstable and the open-source Russian TTS engine could not be installed to use RKNN conversion. The NVIDIA Jetson Orin Nano was selected for its mature CUDA ecosystem, 8 GB unified memory, and stable on-device inference. Critically, the architecture was simplified from client–server to fully self-contained: STT and LLM run on the Jetson GPU, TTS on the Jetson CPU - no external server is required. Sanzhar installed the Jetson OS (JetPack), configured CUDA drivers, and deployed all Python services.

Phase 3 - TTS resolution (Weeks 5–9). In Russian speech synthesis, XTTS v2 has a large delay (~ 1850 ms) and is poorly understood. The Russian language could not be produced by Piper TTS (only the English model). Silero v5 ru [13] produced natural, intelligible Russian, with at approximately ~ 210 ms latency, and thus was selected. Kamila has completed flow stability and TTS model tests. Instead of waiting to receive each response in a whole, Aizhar launched a TTS streaming, which sends each synthesized sentence to an audio output as it is being produced. It resulted in decreasing the latency from 5 seconds to 1 second.

Phase 4 - LLM and pipeline refinement (Weeks 4-10). The Jetson GPU was used for a detailed comparison of ten models. During model experiments and `llama.cpp` integration, Kamila tested each candidate on each of three criteria (Latency, VRAM usage, and Output Quality). Models with parameters less than 1b (Gemma 270m, Tinyllama, Qwen 0.6b) were unable to support comprehensive learning dialogue. Latency budget increased by parameters 7b (Mistral, Qwen 2.5-7b). Q4_K_M Gemma-4-E2B was the best combination in terms of response quality, time-to-first data (290 ms), and VRAM consumption (2.1 GB). To ensure that the sentence boundary definition accurately triggered the TTS pipeline without interrupting the middle generation responses, Aizhar simultaneously updated the VAD integration and the flow logic of LLM. At this stage, Rassul also assisted with pipeline support and Jetson integration testing. Moldir began designing the architecture of the storage system to deliver customized context to the pipeline.

Phase 5 - System integration (Weeks 8–12). Moldir supervised the integration of all components created in stages 1-4 and combined the pipeline, memory system, games, and launcher into a single, integrated program. Sanzhar finished all the database schemes and ensured that all the game modules

were correctly connected to the tracking columns of `game_eventss`, `user_word_progress`, and `hints`. The level determination game and the general `word_knowledge.json` vocabulary of the three educational games were tested and verified end-to-end in the JSON interface.

Phase 6 - Hardware enclosure (Weeks 10–14). After finishing the design of the CAD casing, Rassul created the final beige PLA casing with cutouts to accommodate the jeton, display, and all ports. Additionally, Kamila was involved in 3D prints and sourced all the components.

Phase 7 - Benchmarking and finalisation (Weeks 13–16). Each STT, LLM, and TTS model has been formally compared by the entire team. Three evaluators completed these nine qualitative rapid assessments. Aizhar has finished designing the user interface for each launch page and in-game screen. Moldir ensured that the end-to-end systems of all installed components were stable. An integrated demo prototype and documentation have been prepared and prepared for presentation.

VII. EVALUATION

The full model benchmarking is provided in the appendix. Equipment selection, speech recognition accuracy in noisy and quiet environments, end-to-end delay, LLM response quality, and automated quality measurements are the five parameters of the test.

A. Hardware Platform Comparison

Three apparatus candidates were evaluated for placement (see Table VIII in the appendix). Due to CPU computing being too slow to infer in quantized LLM, Raspberry Pi 5 has a catastrophic end-to-end lag ($>8s$). TTS acceleration allowed the Orange Pi’s RKNN NPU to be tested, although there was no open-source Russian TTS engine that could modify the RKNN at the time of recording, and the audio stream was unreliable. The NVIDIA Jetson Orin Nano was chosen because it has a special CUDA, 8 GB of unified memory between the processor and the graphics card, and an embedded software environment that allowed us to run STT and LLM on the GPU and TTS on the CPU.

B. Speech Recognition Accuracy

Of the six alternatives (see Table X in the appendix), the multilingual Whisper Tiny was selected, as its silent state was 4.8% WER. We evaluated the Whisper using 120 Russian words sampled on three acoustic backgrounds using different noise levels as adult speakers to assess their resilience in the real world. Table III presents the results.

TABLE III
WHISPER TINY (MULTILINGUAL) WER BY ACOUSTIC ENVIRONMENT

Environment	WER (%)	Avg. Latency (ms)
Quiet room (<35 dB)	4.8	320
Moderate noise (35–55 dB)	9.3	335
High noise (>55 dB)	17.6	341

Recognition quality is below the acceptable threshold in a quiet and moderate environment covering the primary placement scenario (home or school). WER rises to 17.6% in high-noise environments, indicating that future iterations will benefit from the noise suppression preprocessing step.

C. End-to-End Latency Breakdown

Table IV shows the costs associated with the installation of the system (Whisper Tiny + Gemma-4-E2B Q4_K_M + Silero v5 ru), including delays across the entire pipeline.

TABLE IV
END-TO-END PIPELINE LATENCY BREAKDOWN (50 UTTERANCES, JETSON ORIN NANO)

Stage	Mean (ms)	% of Total
VAD + audio buffering	80	8.3%
STT - Whisper Tiny (GPU)	320	33.3%
LLM time-to-first-token (GPU)	290	30.2%
TTS first sentence (CPU)	210	21.9%
System overhead	60	6.3%
Total (perceived)	960	100%
Full response (all sentences)	~2600	—

The main focus of optimization is the GPU levels of STT and LLM, which account for 63.5% of the accepted latency. The chunked TTS stream means that child is approximately hears the first audio sentence at 960 milliseconds, after which he stops speaking, which is much earlier than the 3-second conversational threshold. Only 6.3% of the total latency is due to the interprocess communication between the TTS thread in the processor and the inference processes in the GPU, which is measured by the top row of the system.

D. LLM Response Quality: Automated Metrics

In addition to the qualitative 9-prompt assessment (Table XII in the appendix), we also evaluated the response quality with two automated measures: the cosine similarity of the prompt embedding and response embedding (measure of relevance) and the confusion of the response with the corresponding records language model (measure of fluency/conformity). To evaluate their performance in these scenarios, three finalists LLMs were tested in three interaction scenarios 90 total per model (30 interactions each). Table V shows the results.

TABLE V
AUTOMATED QUALITY METRICS BY CONVERSATION CONTEXT (GEMMA-4-E2B Q4_K_M)

Conversation Context	Cosine Sim.↑	Perplexity↓
Basic greetings & introductions	0.81	12.4
Vocabulary translation requests	0.78	14.1
Grammar explanation questions	0.74	16.8
Overall mean	0.78	14.4

TABLE VI
AUTOMATED QUALITY METRICS: MODEL COMPARISON (OVERALL MEAN)

Model	Cosine Sim.↑	Perplexity↓
Gemma-3-4B Q4_K_M	0.69	19.7
Qwen 2.5-7B-Instruct	0.74	16.2
Gemma-4-E2B Q4_K_M	0.78	14.4

Gemma-4-E2B had the highest cosine similarity (0.78 total) and lowest perplexity (14.4), indicating that the response was both linguistically competent and accessible to the child. It is not surprising that cosine similarity diminishes when using grammatical explanatory differences, as complex educational

responses should naturally have a word difference for a short investigation (0.74). These results confirm Gemma-4-E2B as the appropriate final model selection and confirm the qualitative evaluation results in Table XII.

E. LLM Qualitative Evaluation

We divided the three finalists into three educational groups and evaluated them according to nine verbal guidelines: Basic Concept (BU1-BU3), Helpful for Learning (HL1-HL3), and Correct Russian (CR1-CR3). Three outside evaluators rated the answer on the Likert scale from 1 to 5. Table XII in the appendix contains all the information. Gemma-4-E2B outperformed Qwen 2.5-7B (4.20) and Gemma-3-4B (3.63) with an average score of 4.62 on all questions. Gemma-4-E2B did very well on grammar explanation tasks (HL category), where smaller models were simplified or incorrect explanations were produced.

F. Quantisation Impact on Response Quality

To support the Q4_K_M selection, we evaluated the same Gemma-4-E2B base model at three quantization levels. Table VII reports on the preservation of accuracy, speed of inference and memory reduction (compared to the FP16 baseline on 100 educational prompts).

TABLE VII
QUANTISATION IMPACT ON GEMMA-4-E2B QUALITY AND PERFORMANCE

Quantisation	Accuracy	Mem. Reduction	Speedup
FP16 (baseline)	100%	0%	1.0×
Q8_0	94.3%	38%	1.3×
Q4_K_M	88.1%	63%	1.7×
Q3_K_M	74.6%	74%	1.9×

With a $1.7\times$ speed increase, 63% memory savings, and 11.9% reduction in accuracy, Q4_K_M provides the best balance. Q3_K_M degradation (25.4%) was found to be too heavy to use with an educational application where accuracy is critical.

G. System and Game Screenshots

Fig. 10 shows the adaptive level identification game. Figs. 12–13 illustrate the three educational games.

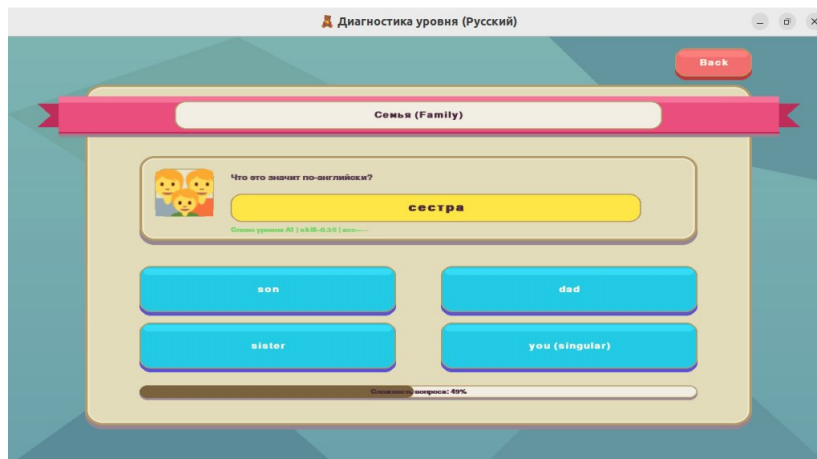


Fig. 10. The level-based identification game features four answer options, an increasing difficulty signal, a flexible multiple-choice question format, a topic banner, and a specified target word..

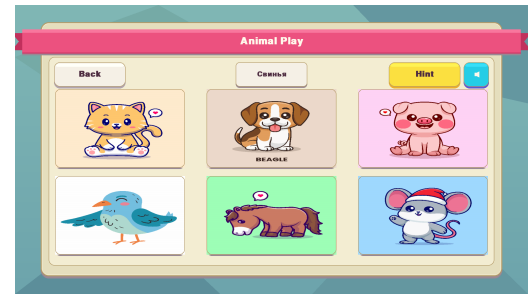
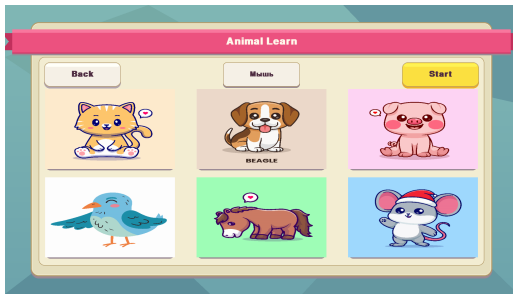


Fig. 11. Animal Game (A1). On the left: learning mode: a young man can touch hidden areas, see animals, and hear their names in Russian. Right: In game mode, a young child responds to an auditory cue by clicking on the corresponding animal; the score is shown at the bottom right.

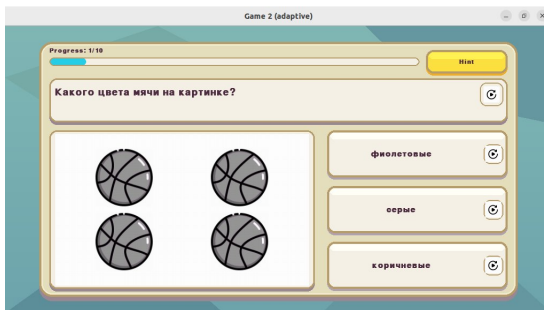


Fig. 12. The color-number game (A1 / A2) used re-painted objects to create a visual scene. Multi-choice questions require correct morphological agreement regarding the qualities of counting and color in Russian. Left: color question. Correct: question about numbers.



Fig. 13. Word position spatial thinking game in word positions (A2). In response to a vocal Russian spatial request such as "Put the cup on the table, the ball between the chair and the box," the young man arranges the items.).

VIII. CONCLUSION AND FUTURE WORK

Our project *Boxy* is an educational chatbot on a single edge device for child L2 learners that has low latency, fully preserves privacy, and is self-contained. The final system runs entirely on an NVIDIA Jetson Orin Nano: STT and LLM on the GPU on board, while TTS runs on the CPU. The project consists of a real-time voice pipeline with a perceived latency of 2.6 s, a RAG-backed memory system, an adaptive CEFR A1/A2 level-identification game, and three purpose-built Russian vocabulary games. Several evaluations justify the selection between ten LLMs, six STT models, and three TTS engines. Automated metrics and a qualitative evaluation were used to choose the LLM model that balances between high quality response

as a child tutor and computational efficiency given limited resources. All data is stored locally on the device and is not sent to external servers, which supports privacy principles of regulations such as COPPA and GDPR.

Limitations. A formal study with children was not completed within the project timeline. Therefore, educational effectiveness is yet to be confirmed with empirical data. Because of hardware limitations, it was impossible to run three models simultaneously on GPU. As a result, speech synthesis was done on CPU, separate from the GPU inference pipeline. WER degrades to 17.6% in high-noise environments, which could limit the performance in noisy classrooms.

Future Work

1. **Formal user study.** Gather a group of at least 10 children aged 5-8 in a controlled classroom environment and measure vocabulary, engagement, and enjoyment over a period of at least 4 weeks using pre-post vocabulary tests and engagement self reports. This will provide the empirical data currently lacking in the evaluation.
2. **Noise suppression.** Adding a noise suppression model like RNNoise or DeepFilterNet would reduce WER in high-noise environments from the current 17.6% closer to the quiet-room baseline of 4.8%. DeepFilterNet adds a negligible latency of 40 ms, independent of the CPU model [15].
3. **Kazakh or other languages.** The pipeline, database schema, and game logic work regardless of a specific language. Adding Kazakh language requires: a Kazakh vocabulary (available via the National Corpus of the Kazakh Language), STT model with Kazakh (Whisper supports Kazakh zero-shot), and a Kazakh TTS voice (Silero maintains Kazakh model). All three components are open-source.
4. **Progress dashboard.** A local web application (FastAPI + React, no external connectivity required) would allow parents to view session history, learned vocabulary, game scores, and hint usage. The `game_events` and `user_word_progress` SQLite tables already capture all necessary data, only the GUI remains to be built.
5. **Advanced game types and higher CEFR levels.** The current games cover CEFR A1-A2 vocabulary and basic grammar. Future game types could include: (a) sentence construction tasks at B1 level using drag-and-drop word tokens (b) story-based roleplay scenarios where the child converses with the LLM inside a structured narrative, and (c) listening comprehension games where the child answers questions about a short Russian audio passage.

APPENDIX

TABLE VIII
EDGE HARDWARE PLATFORM COMPARISON

Criterion	RPi 5	Orange Pi	Jetson Orin Nano
RAM	8 GB	8 GB	8 GB
GPU	None	RKNN NPU	1024-core CUDA
Russian TTS on NPU	N/A	No	N/A (CPU used)
LLM on GPU	No	No	Yes
Streaming stability	Poor	Poor	Good
CUDA ecosystem	No	No	Yes
End-to-end latency	>8s	N/A	<1s*
Selected	No	No	Yes

*Perceived latency to first audio sentence.

TABLE IX
TTS MODEL BENCHMARK (RUSSIAN LANGUAGE)

Model	Lat. (s)	Intel. (1–5)	Russian	RAM (MB)
Piper ru_RU	4.0	1.1	Yes	~50
XTTS v2	7.0	2.3	Yes	~2100
Silero v5_ru	1.3	4.6	Yes	~130

Intel.: mean score from 3 blind evaluators.

TABLE X
STT MODEL BENCHMARK — QUIET CONDITIONS, RUSSIAN SPEECH

Model	WER (%)	Latency (s)	VRAM/RAM (GB)	Languages
Vosk en-us-0.22-lgraph	38.4	0.22	~0.5	1
Vosk en-us-0.15	35.1	0.24	~0.4	1
OmniASR CTC 300M	22.6	1.09	~2.0	1
Whisper Medium (English)	12.3	1.98	~4.7	1
Whisper Tiny (multilingual)	18.7	0.40	~1.0	98
Whisper Base (multilingual)	4.8	0.89	~1.8	99

TABLE XI
LLM BENCHMARK ON JETSON ORIN NANO (10 MODELS)

Model	TTFT (s)	VRAM (GB)	Quality
Gemma 3 270M	0.43	0.4	Poor
Qwen 3:0.6B	0.51	0.6	Poor
TinyLlama 1.1B	0.61	0.9	Poor
BitNet b1.58-2B-4T	0.65	1.2	Acceptable
Qwen 2.5:1.5B	0.71	1.4	Acceptable
Gemma-4-E2B Q4_K_M	0.90	2.1	Excellent
Gemma-3-4B Q4_K_M	1.18	3.1	Good
Gemma-3-E4B Q4_K_M	1.30	3.4	Good
Qwen 2.5-7B-Instruct	2.42	5.6	Good
Mistral 7B	2.54	5.8	Good

Quality rated via 9-prompt qualitative evaluation (Table XII).

Nine spoken-style prompts grouped into three categories: Correct Russian (CR), Helpful for Learning (HL), Basic Understanding (BU). Three evaluators scored independently on a 1–5 Likert scale; values are means.

TABLE XII
LLM QUALITATIVE EVALUATION – FULL RESULTS (1 = VERY POOR, 5 = EXCELLENT)

Model	Correct Russian			Helpful for Learning			Basic Understanding			Mean
	CR1	CR2	CR3	HL1	HL2	HL3	BU1	BU2	BU3	
Gemma-3-4B Q4_K_M	3.7	3.4	3.8	3.5	3.3	3.6	3.9	3.7	3.8	3.63
Qwen 2.5-7B-Instruct	4.2	4.0	4.1	4.3	4.1	4.2	4.4	4.2	4.3	4.20
Gemma-4-E2B Q4_K_M	4.6	4.5	4.7	4.6	4.4	4.7	4.8	4.6	4.7	4.62

CR1: Ask for correct translation of English word into Russian. **CR2:** Correct a Russian grammar error. **CR3:** Ask a simple question in Russian about food.

HL1: Ask about the purpose of any part of speech in Russian. **HL2:** Ask a question that would explain the dependency of an adjective on the noun. **HL3:** Ask a question that would explain the relation between a noun and a depending verbs' ending.

BU1: Ask a simple question about colors of rainbow in Russian. **BU2:** Ask a simple question about school subject. **BU3:** Ask a more complicated question about culture of any country.

REFERENCES

- [1] S. Krashen, "Principles and practice in second language acquisition," *Pergamon Press*, 01 1982.
- [2] European Parliament, "General data protection regulation (gdpr)," Official Journal of the European Union, 2016, regulation (EU) 2016/679.
- [3] Federal Trade Commission, "Children's online privacy protection act (coppa)," United States Code, 1998, 15 U.S.C. §§ 6501–6506.
- [4] L. S. VYGOTSKY, *Mind in Society: Development of Higher Psychological Processes*. Harvard University Press, 1978. [Online]. Available: <http://www.jstor.org/stable/j.ctvjf9vz4>
- [5] M. H. Long, "The role of the linguistic environment in second language acquisition," in *Handbook of Second Language Acquisition*, W. C. Ritchie and T. K. Bhatia, Eds. San Diego: Academic Press, 1996, vol. 2, pp. 413–468.
- [6] J. Hamari, J. Koivisto, and H. Sarsa, "Does gamification work? — a literature review of empirical studies on gamification," 01 2014.
- [7] M. Csikszentmihalyi, *Flow: The Psychology of Optimal Experience*, 01 1990.
- [8] C. Packer, V. Fang, S. Patil, K. Lin, S. Wooders, and J. Gonzalez, "Memgpt: towards llms as operating systems." 2023.
- [9] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang, "Lost in the middle: How language models use long contexts," *Transactions of the association for computational linguistics*, vol. 12, pp. 157–173, 2024.

- [10] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, “Robust speech recognition via large-scale weak supervision,” in *International conference on machine learning*. PMLR, 2023, pp. 28 492–28 518.
- [11] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, “Qlora: Efficient finetuning of quantized llms,” *Advances in neural information processing systems*, vol. 36, pp. 10 088–10 115, 2023.
- [12] D. House, L. Bell, K. Gustafson, and L. Johansson, “Child-directed speech synthesis: evaluation of prosodic variation for an educational computer program,” 09 1999.
- [13] S. Team, “Silero models: pre-trained text-to-speech models made embarrassingly simple,” <https://github.com/snakers4/silero-models>, 2025.
- [14] M. Aubakirov, D. Gaziz, N. Amanzhol, L.-A. Kozhamkulova, S. Fazli, and M. Ristin, “Autonomous chatbot for second language acquisition: An agile, edge-based system for early childhood education,” Nazarbayev University, School of Engineering and Digital Sciences, Astana, Kazakhstan, Senior Project Report, 2025.
- [15] H. Schröter, T. Rosenkranz, A. N. Escalante-B, and A. Maier, “Deepfilternet: Perceptually motivated real-time speech enhancement,” *arXiv preprint arXiv:2305.08227*, 2023.