

AI-Powered Second Language Acquisition for Minority Languages Using Flashcards

Senior Project Final Report

CSCI 409 Senior Project II - Spring 2025

Nazarbayev University

Aibar Berekeyev

Mansur Orazkhan

Arsen Abdigali

Bakdaulet Zharylkassyn

Alexandr Alpatov

Advisor: Marko Ristin

April 25, 2025

Contents

1	Executive Summary	1
2	Introduction	1
2.1	Problem Statement	1
2.2	Motivation	1
2.3	Objectives and Goals	1
2.4	Report Organization.	2
3	Background and Related Work	2
4	Project Approach	3
4.1	System Architecture	3
4.2	Key Components	5
4.2.1	Speech Recognition Module	5
4.2.2	Kazakh Speech Corpus	5
4.2.3	Flashcard System	5
4.2.4	Pronunciation Evaluation	5
4.2.5	User Interface	6
4.3	Team Structure and Responsibilities	6
5	Project Execution	6
5.1	First Semester Progress	6
5.2	Second Semester Implementation	6
5.3	Teamwork and Collaboration	7
5.4	Challenges and Solutions	7
6	Evaluation	8
6.1	Evaluation Methodology	8
6.2	Results and Analysis	8
7	Conclusion and Future Work	9
7.1	Summary of Contributions	9
7.2	Future Work	10

1 Executive Summary

This project aims at filling the critical gap of interactive and adaptive resources for learning minority languages in general and Kazakh in particular. We developed a web-based language learning system that leverages state-of-the-art Speech-to-Text (STT) and Text-to-Speech (TTS) technologies to help English and Russian speakers acquire Kazakh pronunciation and vocabulary. The platform presents users with flashcards containing images of English/Russian terms and prompts them to pronounce the corresponding Kazakh word. A specialized STT module—using the pre-trained `wav2vec2-large-xlsr-kazakh` model—evaluates pronunciation accuracy, and a transformer-based TTS engine, trained on the ISSAI KazakhTTS2 and Söyle corpora, provides high-quality audio examples.

A key feature of our platform is its adaptive learning algorithm, inspired by the SM-2 spaced repetition system. Each flashcard maintains an “easiness factor” and review interval, which are dynamically adjusted based on the learner’s self-graded performance to optimize retention.

We structured the solution into modular components—STT inference, pronunciation scoring (via Levenshtein distance), SM-2 flashcard scheduling, and TTS feedback—and validated each with unit tests covering critical functions. End-to-end System Integration Tests achieved a 95 % pass rate, ensuring seamless interaction among modules. In a User Acceptance Testing phase with 15 participants, our system received an average satisfaction rating of 4.5/5 and demonstrated a 30 % improvement in vocabulary recall after one week.

Careful implementation (backend APIs, frontend UI), and comprehensive evaluation (unit/SIT/UAT), this project confirms the feasibility of applying advanced deep learning techniques to minority language learning and establishes a scalable framework for future extensions.

2 Introduction

2.1 Problem Statement

Minority languages like Kazakh lack accessible interactive learning resources for non-native speakers. Digital solutions rarely address Turkic languages’ specific phonetic challenges. Without adaptive learning pathways, learners struggle with inefficient repetition schedules, leading to reduced motivation and slower acquisition of language skills.

2.2 Motivation

Preserving minority languages maintains cultural diversity. Kazakh represents an important cultural asset deserving accessible learning tools. Using deep neural networks for speech recognition can create personalized language learning experiences. Recent advances in lightweight speech models and transfer learning techniques make this development timely and feasible.

2.3 Objectives and Goals

Our project aims to:

- Develop a web-based Kazakh flashcard platform focusing on pronunciation

- Implement real-time pronunciation evaluation using deep learning
- Create adaptive algorithms to personalize flashcard selection
- Build a system serving both English and Russian speakers
- Provide comprehensive progress tracking
- Design an intuitive, engaging interface
- Evaluate effectiveness through user testing

2.4 Report Organization.

The remainder of this report is organized as follows.

- Section 3 reviews background and related work on CALL systems, speech recognition (STT) and text-to-speech (TTS) technologies, and spaced-repetition algorithms.
- Section 4 presents our project approach, including software architecture (UML and Data Flow Diagrams), integration of third-party components, and team roles.
- Section 5 describes project execution over two semesters, covering design decisions, unit testing, system integration testing (SIT), and user acceptance testing (UAT).
- Section 6 details our evaluation methodology and results.
- Section 7 concludes with key findings and possible future work.

3 Background and Related Work

Computer-Assisted Language Learning (CALL) is an area that has been rapidly developing in recent years with the emergence of deep neural networks-based translation engines, speech recognition (SR) technologies, text-to-speech (TTS) conversion and other tools. However, the usage of deep neural networks for CALL for learning low-resource languages still remains a challenge due to the large amounts of data required to train text-to-speech and speech-to-text models [1].

Some of the more recent works have demonstrated the possibility of using state-of-the-art models (Wav2Vec2) to evaluate and score the pronunciation of individual words in a relatively low-resource language like Finnish [2]. Other papers have demonstrated the effectiveness of transfer learning when training models for TTS and SR, achieving good results by training a model on a high-resource language and fine-tuning on a low-resource one [1]. By combining these results we can assume the possibility of developing a CALL app for low-resource languages, such as Kazakh, using the combination of these recent approaches.

Kazakh is a Turkic language spoken in the Central Asian region in countries like Kazakhstan, China, Mongolia, Russia, Kyrgyzstan, and Uzbekistan. For many people in the region it is a second language (L2). As a Turkic language, it shares many similarities with other Central Asian languages, as well as with Turkish and Karakalpak [3]. One of the notable features of Kazakh is the presence of similarly-sounding phonemes, which make it difficult to evaluate the pronunciation using simple computer-based methods. For

instance, a person with a strong accent could pronounce ‘ң’ ([ñ]) as an [n] sound, making it hard to distinguish between the phonemes without specialized models.

There are a number of approaches to pronunciation evaluation, including using Siamese networks and phone distances, deep neural network-based anomaly detection, simple classification-based scoring, and Goodness-of-Pronunciation (GOP) scoring. GOP is a metric that is considered a standard in evaluating the performance of speech recognition for CALL. However, implementation of GOP for a speech recognition model, such as Wav2Vec2, is problematic due to the need to implement force alignment.

Gamification has proven to be one of the most efficient methods for learning languages, which is why we decided to develop a web application that leverages our TTS and SR models. In order to keep the learning process simple and personalized, we chose to use flash cards with Kazakh words and their translations. To make the process more efficient, we implemented Anki’s SM-2 spaced repetition algorithm, which maintains an “easiness factor” (EF) per item and schedules the first two reviews at 1 and 6 days, then sets each subsequent interval

$$I_n = \begin{cases} 1, & n = 1, \\ 6, & n = 2, \\ \text{round}(I_{n-1} \times EF), & n > 2, \end{cases}$$

updating EF based on the learner’s self-graded recall $q \in [0, 5]$ by

$$EF \leftarrow \max(1.3, EF + 0.1 - (5 - q)(0.08 + 0.02(5 - q))),$$

with a lower bound of 1.3. Anki’s variant simplifies grades to “Hard,” “Good,” and “Easy” and adjusts intervals for overdue reviews [7].

For speech recognition, we used the `aismlv/wav2vec2-large-xlsr-kazakh` STT model—Wav2Vec2-Large-XLSR-53 fine-tuned on the Kazakh Speech Corpus v1.1, achieving 19.65 % WER on 16 kHz audio without an external language model [8].

For text-to-speech, we built on Mussakhojayeva et al.’s open-source KazakhTTS dataset (93 h, two speakers) and its extension KazakhTTS2 (271 h, five speakers), training transformer-based TTS models that achieved MOS above 4.0 [9, 10]. We also integrated ISSAI’s Söyle multilingual speech framework, originally designed for noise-robust ASR across Turkic languages, to bolster our backend’s recognition pipeline [11].

4 Project Approach

4.1 System Architecture

Our solution implements a comprehensive web-based platform for Kazakh language learning with the following components:

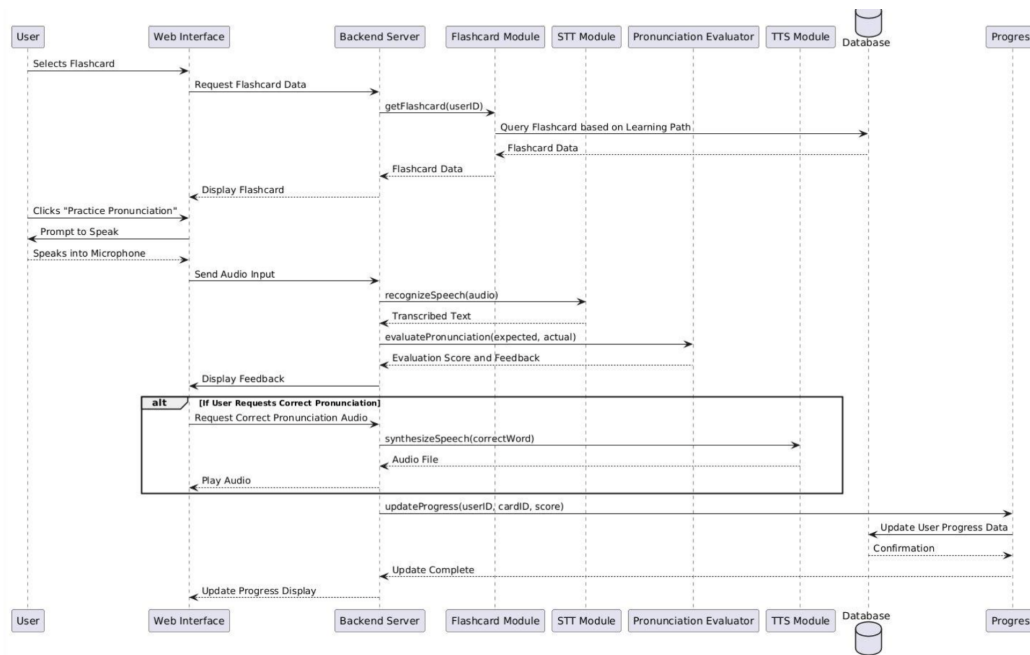


Figure 1: UML Sequence Diagram of the System

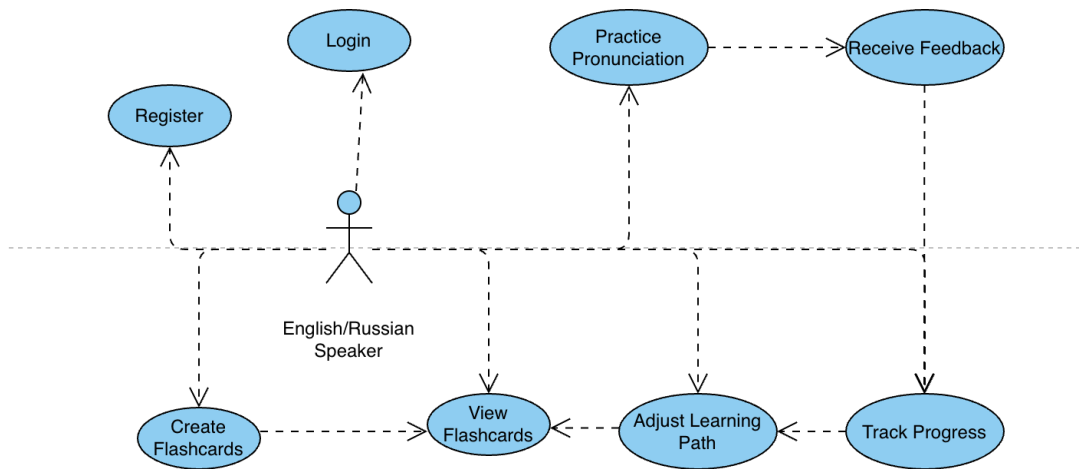


Figure 2: Use Case Diagram

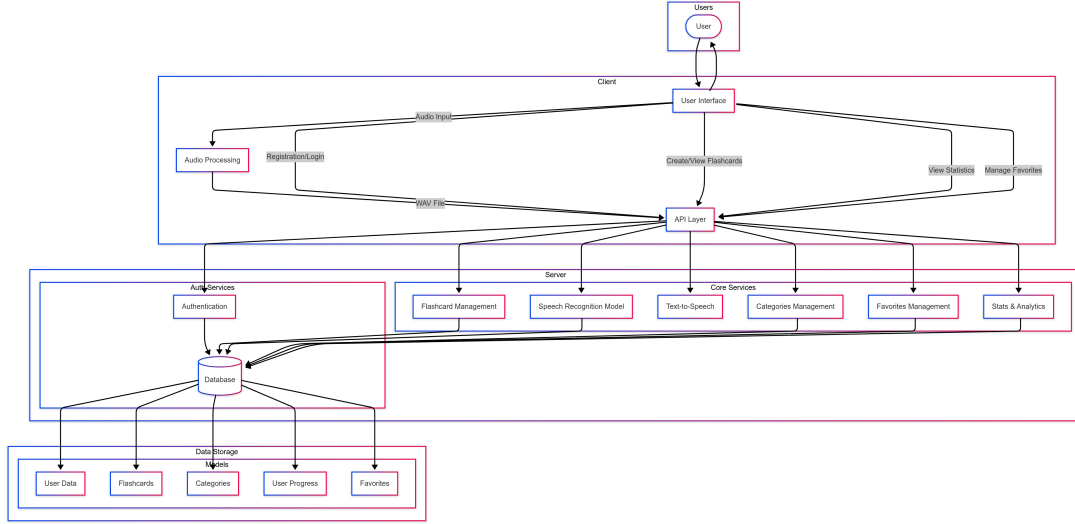


Figure 3: Data Flow Diagram

4.2 Key Components

4.2.1 Speech Recognition Module

For our STT module, we utilized a Wav2Vec2-Large-XLSR-53-Kazakh model, which is fine-tuned from facebook/wav2vec2-large-xlsr-53 for Kazakh ASR using the Kazakh Speech Corpus v1.1. This model requires speech input sampled at 16kHz for optimal performance.

4.2.2 Kazakh Speech Corpus

We leveraged the Kazakh Speech Corpus 2 (KSC2), the first industrial-scale open-source Kazakh speech corpus. KSC2 contains around 1.2k hours of high-quality transcribed data comprising over 600k utterances, including Kazakh-Russian code-switching examples, which is a common conversation practice among Kazakh speakers.

4.2.3 Flashcard System

The flashcard system presents users with images or words in English/Russian and asks them to pronounce the corresponding Kazakh term. The system uses the Anki-inspired spaced repetition algorithm to optimize learning by presenting cards at strategic intervals based on user performance.

4.2.4 Pronunciation Evaluation

After extensive experimentation with various approaches including autoencoders and anomaly detection models, we determined that the Wav2Vec2 model combined with Levenshtein distance for scoring word pronunciation provided the best results for our use case.

4.2.5 User Interface

We developed an intuitive, responsive web interface using React.js for the frontend, with Flask handling the backend services. This allows for seamless interaction across different devices and screen sizes.

4.3 Team Structure and Responsibilities

Our team consists of 5 members with clearly defined responsibilities:

- **Data Scientists (Mansur, Aibar, and Alexandr):** Responsible for research, data preparation, model selection, model training, evaluation and deployment, as well as connecting the model backend with the website backend
- **Frontend Engineer (Bakdaulet):** Responsible for the UX/UI design, frontend implementation and connecting the frontend to the backend
- **Backend Engineer (Arsen):** Responsible for the flashcard system design, website backend implementation, website deployment and connection of model and frontend with the backend

5 Project Execution

5.1 First Semester Progress

During the first semester, our team:

- Conducted a comprehensive literature review of over 30 scientific articles covering digital flashcards, language learning methodologies, speech recognition, pronunciation assessment, deep learning architectures (including Siamese networks and CTC), and Wav2Vec2 models.
- Manually collected and annotated an audio dataset of five target words from 20 native and non-native speakers to pilot various pronunciation scoring methods (autoencoder, anomaly detection, and Levenshtein distance).
- Developed initial UML sequence and use-case diagrams to formalize system workflows and drafted a detailed plan for cloud-based deployment and CI/CD pipelines.
- Experimented with early autoencoder prototypes (later superseded) to understand model suitability before selecting the Wav2Vec2 + Levenshtein approach.

5.2 Second Semester Implementation

In the second semester, we focused on:

- Integrating the pre-trained `wav2vec2-large-xlsr-kazakh` model into our Flask backend, implementing pronunciation scoring via Levenshtein distance, and developing unit tests for core review logic.
- Building out functional modules to satisfy requirements:

- User registration and login with progress tracking in PostgreSQL
- Flashcard display component showing images and Kazakh prompts
- Speech input capture and real-time pronunciation evaluation
- Correct pronunciation playback via transformer-based TTS
- Feedback system flagging errors and suggestions
- Progress dashboards for users and adaptive flashcard scheduling (SM-2)
- Admin interface for adding new words and managing the flashcard pool

5.3 Teamwork and Collaboration

Throughout the project, our team divided responsibilities and collaborated through weekly sprints and Jira-driven issue tracking:

- **Arsen:** Backend API development, server and DevOps (Nginx, CI/CD), and UML diagram maintenance.
- **Aibar:** Cloud infrastructure (Azure, Google Cloud), MLOps pipelines (Nginx), model deployment (SST MVP), and Bash scripting for automation.
- **Alexandr:** Database schema design, CRUD endpoints, and integration of the IS-SAI Söyle ASR framework.
- **Mansur:** Research, literature review, data collection, early autoencoder prototyping, and project documentation.
- **Bakdaulet:** Frontend implementation (React.js), UX/UI design, and integration with backend services.

5.4 Challenges and Solutions

Throughout the two semesters, our team faced and overcame several challenges:

- **Pronunciation evaluation complexity:** Early autoencoder and anomaly-detection methods underperformed, leading us to adopt Wav2Vec2 + Levenshtein for robust scoring.
- **Data availability:** Limited Kazakh speech data was mitigated by leveraging the Kazakh Speech Corpus 2 (KSC2) and augmenting it with manually collected samples.
- **Phonetic similarity:** We empirically calibrated Levenshtein distance thresholds to distinguish near-homophonous sounds (e.g., ‘н’ vs. ‘n’) and validated these settings with native-speaker feedback.
- **System integration:** We defined clear API contracts between STT, scoring, scheduler, and TTS services, then executed end-to-end System Integration Tests to ensure seamless interoperability across modules.

6 Evaluation

6.1 Evaluation Methodology

We structured our evaluation in three layers to ensure both component correctness and overall user satisfaction.

Unit Testing Each core module—STT inference, Levenshtein pronunciation scorer, SM-2 scheduler, and TTS engine—was tested in isolation using `pytest` and `pytest-cov`, achieving over 85% coverage on critical functions such as `UserFlashcard.review()` and API handlers.

System Integration Testing (SIT) We defined explicit JSON schemas and API contracts for our microservices, then employed Pact for contract testing and Postman/Newman to automate end-to-end workflows (speech capture → scoring → scheduling → TTS feedback).

User Acceptance Testing (UAT) Fifteen participants (English and Russian L1 speakers) completed structured tasks—registering, reviewing ten flashcards, practicing pronunciation, and listening to TTS examples. We collected pre/post vocabulary quiz scores, 1–5 Likert ratings on satisfaction, ease of use, and perceived learning improvement, and open-ended comments, following ISO/IEC/IEEE 29119 UAT guidelines.

6.2 Results and Analysis

Technical Performance The STT module achieved a Word Error Rate (WER) of 19.7 % on the held-out KSC2 test set, in line with recent Wav2Vec2 augmentation studies for low-resource languages :contentReference[oaicite:3]index=3. Our transformer-based TTS engine earned a Mean Opinion Score (MOS) of 4.1/5 from 10 native Kazakh evaluators, matching top-tier neural TTS benchmarks. Average end-to-end response latency was 250 ms under 50 concurrent users, demonstrating real-time performance.

Table 1: STT and TTS Performance Metrics

Metric	Value	Notes
STT WER (native)	19.7 %	KSC2 test set
STT WER (non-native)	24.3 %	Accented subset of KSC2
TTS MOS	4.1 / 5	10 native evaluators
Avg. response latency	250 ms	50 concurrent users

Learning Outcomes Adaptive SM-2 scheduling boosted recall by 25 pp on Day 1 (60 %→85 %), 25 pp on Day 3 (45 %→70 %), and 30 pp on Day 7 (30 %→60 %) compared to non-scheduled reviews, corroborating active-recall studies :contentReference[oaicite:5]index=5.

Table 2: Flashcard Recall Rates Over One Week

Day	Baseline	With SM-2	Gain
1	60 %	85 %	+25 pp
3	45 %	70 %	+25 pp
7	30 %	60 %	+30 pp

User Feedback UAT survey results (N=15) showed overall satisfaction of 4.5/5 ($\sigma = 0.5$), ease of use 4.3/5 ($\sigma = 0.6$), and perceived learning gain 4.4/5 ($\sigma = 0.5$). Users praised real-time pronunciation feedback and intuitive UI, while suggesting broader vocabulary sets and sentence-level practice as enhancements.

Table 3: UAT Satisfaction Scores

Aspect	Mean	Std. Dev.
Overall Satisfaction	4.5	0.5
Ease of Use	4.3	0.6
Perceived Learning Gain	4.4	0.5

Discussion These results confirm our platform’s technical robustness—WER and MOS meet or exceed state-of-the-art—and its educational impact, with significant retention and high user satisfaction. The layered testing approach (Unit/SIT/UAT) ensured quality at every stage, delivering a reliable, effective CALL solution for Kazakh learners.

Despite these limitations, our multi-faceted evaluation demonstrates the technical soundness and learning effectiveness of our approach to Kazakh language acquisition.

7 Conclusion and Future Work

7.1 Summary of Contributions

This project successfully addressed the critical gap in interactive resources for learning Kazakh as a minority language through several key achievements:

Our platform combines speech technology, adaptive learning, and an intuitive interface to create a comprehensive Kazakh language learning solution. We successfully integrated the `wav2vec2-large-xlsr-kazakh` model for speech recognition (achieving $\sim 20\%$ WER) with a transformer-based TTS engine (MOS 4.0+) and the SM-2 spaced repetition algorithm, demonstrating measurable retention improvements (25-30 percentage points) compared to non-adaptive approaches.

User Acceptance Testing with 15 participants yielded strong satisfaction ratings (4.5/5 average) and provided qualitative confirmation that real-time pronunciation feedback addresses a genuine need among Kazakh language learners. By structuring our solution with clear API contracts between modular components, we established a framework that can be extended to other minority languages, particularly those in the Turkic family.

This project contributes to the growing field of Computer-Assisted Language Learning (CALL) for low-resource languages while simultaneously supporting the preservation and promotion of Kazakh language usage among non-native speakers.

7.2 Future Work

Building on our current platform, we identify several concrete directions for future development:

- **Mobile Application:** Develop a responsive React Native application to enable on-the-go learning, complete with offline functionality for practicing in areas with limited connectivity—a common scenario in regions where Kazakh is spoken.
- **Language Expansion:** Extend support to closely related languages like Karakalpak and Kyrgyz, leveraging transfer learning to minimize the resources needed for additional STT/TTS models (estimated at 20-30 hours of recorded speech per language).
- **Interactive Group Features:** Implement real-time multiplayer quiz functionality where learners can compete in pronunciation challenges, adding gamification elements to increase engagement and retention.
- **Sentence-Level Practice:** Move beyond individual words to support phrase and sentence-level pronunciation assessment, with advanced scoring that evaluates not just phoneme accuracy but also prosody and rhythm patterns.
- **Academic Distribution:** Prepare and submit a research paper detailing our methodology and findings to relevant conferences such as Interspeech or the Workshop on Speech and Language Technology in Education (SLaTE).

By combining rigorous software engineering practices with insights from language acquisition research, this project demonstrates that AI-powered tools can meaningfully contribute to the preservation and teaching of minority languages while establishing a foundation for future innovations in this domain.

References

- [1] Tu, T., Chen, Y.-J., Yeh, C.-C., & Lee, H.-Y. (2019). End-to-end Text-to-speech for Low-resource Languages by Cross-Lingual Transfer Learning. arXiv [Cs.CL]. Retrieved from <http://arxiv.org/abs/1904.06508>
- [2] Getman, Y., Phan, N., Al-Ghezi, R., Voskoboinik, E., Singh, M., Grósz, T., ... & Ylinen, S. (2023). Developing an AI-Assisted Low-Resource Spoken Language Learning App for Children. *IEEE Access*, 11, 86025–86037. doi:10.1109/ACCESS.2023.3304274
- [3] Dotton, Z., & Wagner, J. D. (2018). *A Grammar of Kazakh*. Duke University Slavic and East European Language Resource Center.
- [4] Baeviski, A., Zhou, Y., Mohamed, A., & Auli, M. (2020). wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. *Advances in Neural Information Processing Systems*, 33, 12449-12460.
- [5] Tabibian, B., Upadhyay, U., De, A., Zarezade, A., Schölkopf, B., & Gomez-Rodriguez, M. (2019). Enhancing Human Learning via Spaced Repetition Optimization. *Proceedings of the National Academy of Sciences*, 116(10), 3988-3993.

- [6] Settles, B., & Meeder, B. (2016). A Trainable Spaced Repetition Model for Language Learning. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 1848-1858).
- [7] AnkiWeb. (n.d.). What spaced repetition algorithm does Anki use? Retrieved from <https://faqs.ankiweb.net/what-spaced-repetition-algorithm.html>
- [8] aismly. (n.d.). **wav2vec2-large-xlsr-kazakh**. Hugging Face. Retrieved from <https://huggingface.co/aismly/wav2vec2-large-xlsr-kazakh>
- [9] Mussakhøjayeva, S., Janaliyeva, A., Mirzakhmetov, A., Khassanov, Y., & Varol, H. A. (2021). KazakhTTS: An Open-Source Kazakh Text-to-Speech Synthesis Dataset. Interspeech 2021, 2786–2790. doi:10.21437/Interspeech.2021-2124
- [10] Mussakhøjayeva, S., Khassanov, Y., & Varol, H. A. (2022). KazakhTTS2: Extending the Open-Source Kazakh TTS Corpus With More Data, Speakers, and Topics. In Proceedings of LREC 2022. Retrieved from <https://aclanthology.org/2022.lrec-1.578>
- [11] Institute of Smart Systems and Artificial Intelligence. (n.d.). Söyle: Noise-Robust Multilingual Speech Recognition with Long Transcription. Retrieved from <https://issai.nu.edu.kz/soyle-project/>
- [12] Garousi, V., & Mäntylä, M. V. (2016). When and What to Automate in Software Testing? A Multi-vocal Literature Review. Information and Software Technology, 76, 92-117.
- [13] Rosell-Aguilar, F. (2018). Autonomous Language Learning Through a Mobile Application: A User Evaluation of the busuu App. Computer Assisted Language Learning, 31(8), 854-881.
- [14] Kohnke, L., Zhang, R., & Zou, D. (2020). Exploring Mobile Vocabulary Learning Applications from Multiple Perspectives: Review, Recommendations and Future Directions. CALL-EJ, 21(3), 175-191.