Application for Effective Memorization of Learning Material

Ihor Rohatiuk[1], Radoslav Vargic[1]
[1] Faculty of Electrical Engineering and Information Technology, Slovak University of Technology in Bratislava, Slovakia
xrohatiuk@stuba.sk

Abstract - **In this paper, the development of a web-based application designed for effective memorization of learning material using flashcards is presented. The study addresses the problem of knowledge retention and memory decay, referencing the widely recognized Ebbinghaus forgetting curve. A comparative analysis of existing spaced repetition algorithms and applications was conducted, including manual spaced repetition systems, absolute, relative, memory-model-based, and neural network-based approaches. Based on the findings, a suitable algorithm was selected and implemented within the application to optimize the learning process. Additionally, the paper explores modern approaches to automatic flashcard generation from text using large language models (LLMs). The proposed solution integrates AI-based generation as an optional feature, allowing users to create personalized flashcards from arbitrary text content.**

Keywords **- Effective Learning; Spaced Repetition; Application**

I. Introduction

In every age, society has valued every man for his knowledge. With the rapid development of science, the ability to memorize large amounts of information quickly and reliably has become a key part of everyday life, and the equally rapid development of technology is helping everyone acquire this ability.

Effective memorization of large volumes of educational material remains a significant challenge for students, particularly in preparation for exams and assessments. Despite the availability of various applications and learning techniques, many existing solutions are limited in flexibility, adaptability, or scientific foundation. This has led to growing interest in developing more effective, evidence-based tools for knowledge retention.

This paper focuses on the study and implementation of methods for efficient memorization using spaced repetition algorithms. The research includes an analysis of existing memorization techniques and applications, evaluating their strengths, limitations, and technical solutions. Special attention is paid to scientifically validated algorithms and their practical implementations.

Furthermore, the study explores contemporary technologies, including the use of artificial intelligence, to enhance the learning experience. The proposed solution involves the development of a custom web-based application implementing selected memorization techniques and integrating AI-driven flashcard generation.

The paper is divided into two main parts: the analytical section, which provides a detailed review of existing methods, algorithms, and tools; and the practical section, which describes the architecture, implementation, and functionality of the developed application.

II. Problem Analysis

Fast and effective memorization of learning material is essential for successful studying. As early as 1885, German psychologist Hermann Ebbinghaus laid the foundations of memory theory and the concept of spaced repetition. Through his experiments, he examined the rate of forgetting and created the well-known "forgetting curve," illustrating how quickly information fades from memory without review. Ebbinghaus also proposed that regular repetition with gradually increasing intervals can significantly slow this process **Chyba! Nenašiel sa žiaden zdroj odkazov.**, **Chyba! Nenašiel sa žiaden zdroj odkazov.**.

In 1978, T. Landauer and R. Bjork expanded on his theory, demonstrating that expanding intervals between repetitions lead to better long-term retention compared to uniform repetition. These findings paved the way for modern spaced repetition algorithms, which are now evolving from static schemes to flexible systems that account for individual memory characteristics **Chyba! Nenašiel sa žiaden zdroj odkazov.**.

A. Spaced Repetition Algorithms

Ebbinghaus's experiment demonstrated that spaced repetition improves long-term retention. However, it did not account for the complexity of the material, as different types of information decay at different rates. This makes manual tracking of review intervals inefficient and impractical **Chyba! Nenašiel sa žiaden zdroj odkazov.**.

To address this, spaced repetition algorithms were developed to automatically monitor the memorization process and optimize review intervals for individual pieces of information. In this section, I aim to analyze various spaced repetition algorithms in order to select the most suitable one for my own application.

Since the classification of these algorithms in open sources **Chyba! Nenašiel sa žiaden zdroj odkazov.**, **Chyba! Nenašiel sa žiaden zdroj odkazov.Chyba! Nenašiel sa žiaden zdroj odkazov.** remains fragmented, this study attempts to systematize them based on the literature and the specific features of individual approaches. It should be noted that this classification is provisional and reflects the author's perspective, shaped by the reviewed research. It may be revised or expanded as further studies emerge in the field.

In general, spaced repetition algorithms can be grouped into five categories:

1) Manual Spaced Repetition Systems

Manual Spaced Repetition Systems are methods where learners manage the repetition process themselves without automated interval scheduling. Learners manually separate cards into "learned" and "unlearned" categories, repeating the latter until mastered. Although this requires manual effort, it can still be considered an algorithm, as it follows a step-by-step sequence leading to the desired result **Chyba! Nenašiel sa žiaden zdroj odkazov.**.

2) Absolute SRS Algorithms

Absolute SRS algorithms (sometimes referred to as "heuristic" in literature) assign exact dates for when a card should be reviewed, using fixed time intervals between repetitions. Their main limitation is the inability for users to review material earlier than scheduled, restricting opportunities for extra practice or catching up **Chyba! Nenašiel sa žiaden zdroj odkazov.**.

3) Relative SRS Algorithms

Relative SRS algorithms determine the review timing based on the user's self-assessed confidence rating (usually on a scale from 1 to 5), rather than setting fixed intervals like Absolute SRS algorithms **Chyba! Nenašiel sa žiaden zdroj odkazov.**.

4) Neural Network-Based Algorithms

Neural network-based spaced repetition algorithms use machine learning models to predict optimal review moments based on user behavior. Unlike traditional approaches, these algorithms can consider factors like self-assessed confidence (1–5), past review intervals, answer accuracy, perceived material difficulty, and contextual topic relationships. The model outputs the predicted probability of recall success and recommends the next review interval. Despite their adaptability, these methods require large, high-quality datasets for training and significant computational resources for deployment **Chyba! Nenašiel sa žiaden zdroj odkazov.**.

5) Memory Model-Based Algorithms

Unlike other algorithms, these methods build a mathematical memory model (exponential or power-law decay) that calculates the probability of recalling specific material after a given period. Review intervals are determined based on this model, considering the individual forgetting dynamics of each user **Chyba! Nenašiel sa žiaden zdroj odkazov.**.

B. Existing solutions

As part of the analysis, I found 10 random articles that advise different tops of essential solutions for effective memorization of learning material. These were used to compile the top apps that were advised most often. In Fig. 1, I have indicated a histogram of the frequency of mentions of the app in articles for more visual statistics:

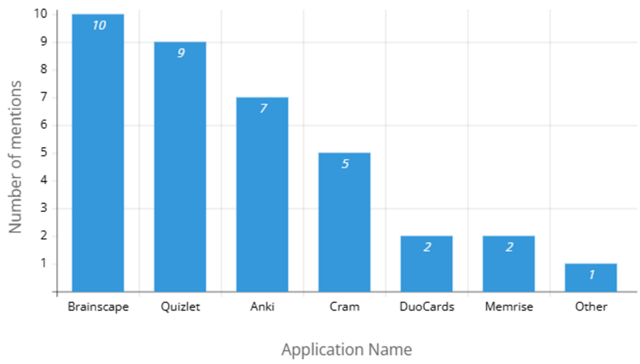

Figure 1. Graph showing the frequency of repetition of a particular application in articles

In the detailed analysis, I will consider applications that have been mentioned more than once. This approach stems from the need to focus on the most popular and widely adopted solutions that have demonstrated practical significance and recognition among users. The following solutions were taken into account: Brainscape Quizlet, Anki, Cram, DuoCards, Memrise.

During the analysis, I focused on both the positive and negative aspects of each solution, as well as on their technological side, which includes the technologies used in the development of the solution and the AI model, in case the solution supports AI.

The analyzed solutions, such as Quizlet, Brainscape, Anki, Cram, DuoCards, and Memrise, are platforms for learning via flashcards with various features. Quizlet offers multi-platform support, spaced repetition, and user interaction, though it lacks AI integration. Brainscape uses AI tools to generate flashcards and organizes material by subjects but is limited by its narrow functionality. Anki, an open-source solution, supports extensive customization and offers two algorithms for spaced repetition but has a complex interface. Cram, while free and versatile, lacks advanced repetition algorithms and has a dated interface. DuoCards focuses on language learning, uses AI for creating flashcards, and employs a modified Leitner system for spaced repetition. Memrise, similar to DuoCards, focuses on language learning and uses multiple-choice questions rather than flashcards. These platforms offer valuable insights for developing a new solution, with a focus on multi-platform support, AI integration, and effective repetition algorithms.

Based on the analysis of existing solutions, we can conclude that although there are many applications using spaced repetition, most of them have limited free functionality, do not utilize AI to enhance learning, and do not consider users' individual memory characteristics. These shortcomings highlight the need to create a custom solution.

C. Technologies to create custom solution

For this project, C# and the .NET platform were chosen for application logic due to their wide capabilities, high performance, and cross-platform support. The visual part will use standard web technologies like HTML and CSS, with C# replacing JavaScript for consistency in the codebase.

AI can automate the process of creating flashcards by extracting key information from text. To enhance the learning process, the solution will include functionality for automatically generating flashcards from user-uploaded text using AI models. This will save users time and improve memory efficiency. Possible solutions include using NLP algorithms or integrating large language models (LLMs) via APIs for better content generation.

III. Proposed Solution

The proposed solution is the development of a web-based application for effective information memorization using spaced repetition techniques, enhanced by artificial intelligence. Unlike most existing tools, this application will offer users the ability to select or customize a memorization algorithm that fits their individual learning preferences.

A. Architecture

In Fig. 2, I have indicated how the system architecture will look like. On the client side, an ASP.NET Core Blazor application will be responsible for the user interface and the implementation of the spaced repetition algorithm. This will allow users to interact with their study material, review flashcards, and track their progress in an intuitive environment.

On the server side, an ASP.NET Core WebAPI will manage interactions with the database and handle AI-based operations. A local LLM (Large Language Model) will be integrated into the server to automatically generate flashcards from uploaded texts, reducing the user's workload and improving content relevance. The server will also provide endpoints for user management, data storage, and API services for AI integration.

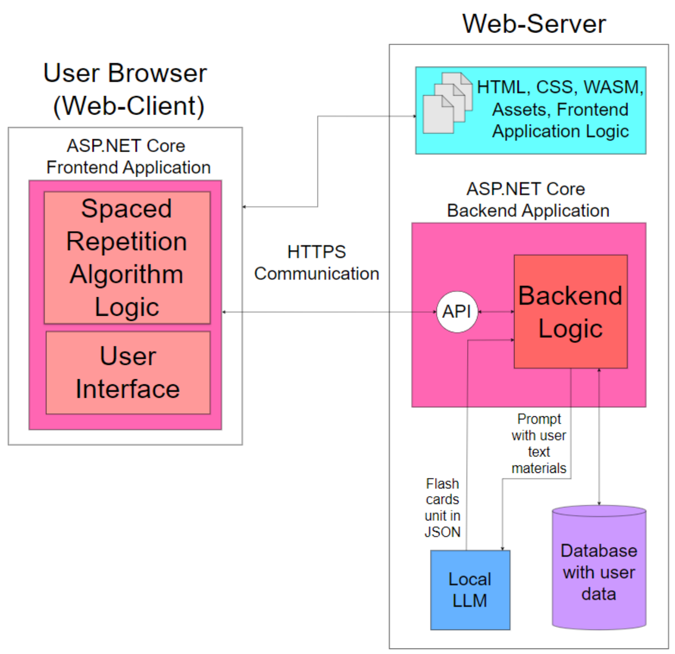

Figure 2. System diagram

Using a client-server architecture offers several key advantages for this solution. It ensures a clear separation of concerns: the client-side Blazor application focuses on user interface and interaction, while the server-side WebAPI handles business logic, data management, and AI operations. This makes the system scalable, maintainable, and easier to extend in the future.

Additionally, this architecture allows for centralized management of data and AI models, enhancing security and simplifying updates. It also improves system performance by distributing workloads between client and server, and enables cross-platform compatibility, since the ASP.NET Core stack is fully cross-platform **Chyba! Nenašiel sa žiaden zdroj odkazov.**.

B. Spaced repetition algorithm

In this project, the SM-2 algorithm, developed by Piotr Wozniak as part of the SuperMemo method, was chosen for implementing spaced repetition. SM-2 has proven to be a simple and effective way to optimize review intervals based on the individual difficulty of memorization. It is widely used in popular applications such as Anki and Mnemosyne, thanks to its reliability and demonstrated effectiveness.

SM-2 allows the repetition intervals to be adjusted according to the user's performance in recalling information, which leads to more efficient learning. Its simplicity and efficiency make it a preferred choice for spaced repetition systems **Chyba! Nenašiel sa žiaden zdroj odkazov.**.

C. Using Artificial Intelligence

In this project, a local Large Language Model (LLM) will be used to automate the generation of flashcards from user-provided text materials. The process will work as follows: the user uploads their text, which is then sent to the LLM along with a predefined system prompt that instructs the model to extract key information and generate question-answer pairs. As a result, the model returns a JSON file containing the ready-to-use flashcards, which the user can immediately incorporate into their learning routine.

For this purpose, LLaMA 3.1 7b will be utilized — an advanced open-source LLM developed by Meta. This model provides a strong balance between performance and hardware requirements, offering high-quality natural language understanding and generation capabilities while being efficient enough to run on local servers. Its main advantages include improved instruction-following, better reasoning capabilities compared to previous versions, and full independence from external paid APIs, ensuring data privacy and system autonomy **Chyba! Nenašiel sa žiaden zdroj odkazov.**.

Key benefits of LLaMA 3.1 7B are:

- Open-source and locally deployable.
- Enhanced instruction-tuning and context understanding.
- Competitive performance for text generation and summarization tasks.
- No dependence on third-party cloud services, ensuring privacy and cost control.

The advantages of LLaMA 3.1 7B are essential for this project because its open-source, locally deployable nature ensures full control over the system without reliance on third-party services, improving privacy and reducing operational costs. Its advanced instruction-following and context understanding capabilities guarantee higher-quality flashcard generation from user materials, directly enhancing learning efficiency. Additionally, local deployment eliminates data security risks and dependency on external providers, making it an ideal solution for sustainable, private, and high-performance educational applications.

D. Outlook and Future Improvements

At the current stage, the application is partially implemented, with core functionalities such as the spaced repetition algorithm and AI-assisted flashcard generation still under development. In the next phases, the primary focus will be on completing these essential modules and refining the integration between the client-side Blazor interface and the ASP.NET Core WebAPI.

Planned improvements include the ability to attach images to flashcards, enhancing the memorization process through visual associations. Additionally, there are plans to optimize the performance and accuracy of the locally deployed LLaMA 3.1 7B model, improving the relevance and structure of generated flashcards. Future updates may also introduce adaptive learning strategies, personalized spaced repetition algorithms, and improved user experience based on collected feedback and usage data. In Fig. 3 and Fig. 4 you can see what the application looks like at the current stage of development:

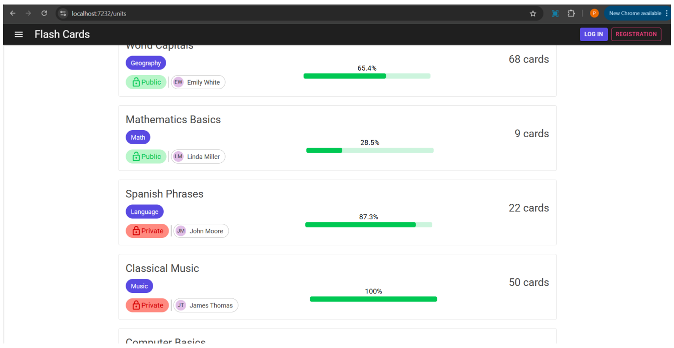

Figure 3. Tab with user units in the application

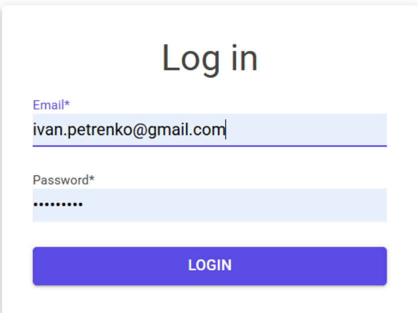

Figure 4. Login page in the application

IV. Conclusion

In this paper, a web-based application for effective memorization of learning material was proposed and partially implemented, combining proven spaced repetition techniques with modern AI-driven flashcard generation. By selecting the SM-2 algorithm for its simplicity and reliability, and integrating a locally deployed LLaMA 3.1 7B model, the solution ensures both high learning efficiency and user data privacy. The client-server architecture, based on ASP.NET Core Blazor and WebAPI, provides a scalable, maintainable, and cross-platform framework. Although the application is still under development, initial results demonstrate the potential for an effective and autonomous learning tool. Future improvements will focus on enhancing flashcard content with images, refining AI generation quality, and implementing adaptive learning features to create a more personalized and engaging educational experience.

Acknowledgement

This paper was supported by NEXT (ERASMUS-EDU-2023-CBHE-STRAND-2), EULiST (ERASMUS), CYB-FUT (ERASMUS+).

References

[1] EduBlox Tutor, “Hermann Ebbinghaus and the Forgetting Curve.” [Online]. Available: https://www.edubloxtutor.com/hermann-ebbinghaus/. [Accessed: 11-May-2025].

[2] Wikipedia, “Forgetting curve.” [Online]. Available: https://en.wikipedia.org/wiki/Forgetting_curve. [Accessed: 11-May-2025].

[3] Wikipedia, “Spaced repetition.” [Online]. Available: https://en.wikipedia.org/wiki/Spaced_repetition. [Accessed: 11-May-2025].

[4] Open-Spaced-Repetition, “Spaced Repetition Algorithm: A Three-Day Journey from Novice to Expert.” [Online]. Available: https://github.com/open-spaced-repetition/fsrs4anki/wiki/spaced-repetition-algorithm:-a-three%E2%80%90day-journey-from-novice-to-expert. [Accessed: 11-May-2025].

[5] Wikipedia, “Spaced repetition: Research and application.” [Online]. Available: https://en.wikipedia.org/wiki/Spaced_repetition#Research_and_application. [Accessed: 11-May-2025].

[6] Expertium, “Benchmark of Spaced Repetition Algorithms.” [Online]. Available: https://expertium.github.io/Benchmark.html#intro. [Accessed: 11-May-2025].

[7] Brainscape Academy, “Comparing Spaced Repetition Algorithms.” [Online]. Available: https://www.brainscape.com/academy/comparing-spaced-repetition-algorithms/. [Accessed: 11-May-2025].

[8] SuperMemo Guru, “Neural Networks in Spaced Repetition.” [Online]. Available: https://supermemo.guru/wiki/Neural_networks_in_spaced_repetition. [Accessed: 11-May-2025].

[9] Open-Spaced-Repetition, “FSRS4Anki Wiki.” [Online]. Available: https://github.com/open-spaced-repetition/fsrs4anki/wiki. [Accessed: 11-May-2025].

[10] Pandac, “Client-Server Architecture.” [Online]. Available: https://pandac.in/blogs/clientserver-architecture. [Accessed: 11-May-2025].

[11] SuperMemo, “The SuperMemo Method.” [Online]. Available: https://www.supermemo.com/en/supermemo-method. [Accessed: 11-May-2025].

[12] Meta AI, “Meta LLaMA 3.1 Launch Announcement.” [Online]. Available: https://ai.meta.com/blog/meta-llama-3-1/. [Accessed: 11-May-2025].