A Context-Aware Retrieval-Augmented Generation Model for Individualized Concept Learning and Adaptive Evaluation

Emmanuel.A[1], RanjithKumar.S[2], Dharshini.D[3]

[1] Research Scholar, Department of Biomedical Engineering, Anna University, Chennai, Tamil Nadu, India
[2]Research Scholar, Department of Advanced Sports Technology, Tamil Nadu Physical Education and Sports University, Chennai, Tamil Nadu, India
[3]Research Scholar, Department of Biomedical Engineering, GKM College of Engineering and Technology, Chennai, India

---***---

Abstract—*in the evolving landscape of digital education, per- signalized and adaptive learning systems have become crucial for enhancing student engagement and comprehension. This research presents a Retrieval-Augmented Generation (RAG)-based AI tutor that enables students to upload educational materials, interact through dynamic conversations, and receive simplified explanations tailored to their understanding. The system supports customizable assessments, adaptive grading, and personalized study plans, ensuring continuous learning progress tracking. By integrating advanced retrieval and generative AI techniques, this application fosters a highly interactive and efficient self-learning environment, bridging the gap between traditional and AI-driven education methodologies.*

Index Terms—**Retrieval-Augmented Generation, AI tutor, adaptive learning, personalized assessment, educational AI, self- learning system**

I. INTRODUCTION

In recent years, advancements in artificial intelligence (AI) have revolutionized education by enabling personalized learning experiences. Traditional learning methods often struggle to cater to individual student needs, resulting in varying levels of comprehension and engagement. The rise of Retrieval- Augmented Generation (RAG) models has introduced a new paradigm, combining information retrieval with generative AI to deliver highly interactive and adaptive learning environ- mends.

This research presents an AI-powered educational assistant designed to facilitate concept understanding through interacts-tive dialogue. The system allows students to upload educational materials in PDF format, which are then processed to provide simplified explanations, answer queries, and generate tailored assessments. By leveraging AI-driven retrieval and reasoning, the platform adapts to each student's proficiency, providing structured feedback and personalized study plans to enhance learning outcomes.

Beyond conventional question-answering systems, the pro- posed model offers customizable testing features, enabling students to set parameters such as test difficulty, marking schemes, and the number of assessments. Grading mechanisms provide insights into learning progress, while improvement Tracking and study plan recommendations further optimize the learning journey. With AI-driven personalization, this approach aims to bridge gaps in student comprehension, fostering a more efficient and engaging self-learning experience. The integration of retrieval- augmented generation ensures that the responses remain con- textually relevant; making this system a step forward in AI- assisted education.

II. LITERATURE SURVEY

A. Rise of AI in Education

The emergence of Artificial Intelligence in the educational sphere marks a crucial turning point in teaching and learning. AI's capacity to analyze large amounts of data, detect patterns, and make real-time informed decisions has opened up exciting new opportunities in education. With digital devices and internet access becoming more widespread, students now have unprecedented access to information. The challenge lies in effectively leveraging this digital environment to enhance the overall learning experience.

The shift from traditional classroom setups to remote, virtual, and blended learning environments is becoming a standard practice. Numerous studies are exploring how Technology-Enhanced Learning (TEL) can boost both efficiency and engagement in education. Xie et al. (2019) provided an overview of how various technologies are integrated into learning ecosystems [1]. In addition, there are specialized reviews that focus on individual subjects, such as flipping mathematics [2] and chemistry [3], illustrating the advantages of incorporating technology into these fields.

As more instructional courses transition online, managing the resulting data has become increasingly complex. The adoption of big data mining and artificial intelligence has gained traction in analyzing information technology within educational settings. Yousuf and Wahid (2021) discussed various areas and applications where AI can offer valuable insights and pave the way for future research [4].

Integrating AI into tutoring systems can pose significant challenges, particularly when it comes to personalization in the learning process [**?**], [**?**] or interventions [**?**]. This type of integration requires careful consideration of system architecture and adaptability, as the AI algorithms are deeply embedded within the system, resembling adaptive learning approaches [**?**]. Intelligent Tutoring Systems (ITSs) that incorporate AI are better equipped to adjust to students' immediate learning needs, which greatly enhances their potential to provide timely support and intervention.

B. Personalized Content Delivery

AI plays a key role in improving personalized content delivery. By analyzing various student data, such as quiz performance, time spent on assignments, and specific interests, AI algorithms can recommend targeted learning materials, modify task difficulty, and suggest additional resources that align with the learner's current knowledge and goals. A notable example of this is Khan Academy, which offers personalized learning paths for math students. Its AI systems continuously monitor each student's progress and adjust the difficulty of exercises to ensure an optimal challenge level.

Research has significantly focused on enhancing student engagement and preventing dropout rates. For instance, Leony et al. (2013) utilized facial expression analysis through cam- eras, employing machine image recognition to gauge students' emotional states [**?**]. Similarly, Pereira et al. (2019) harnessed historical system log data to train a machine learning model that predicts student dropouts in online courses, effectively aiding teachers in boosting engagement [**?**].

The full potential of AI in education is likely still unfolding. While Explainable AI (XAI) gains traction in computer science, its application in education remains in early stages. For instance, Afzaal et al. (2021) implemented a counterfactual method to help instructors understand how AI provides pre- dictions, enabling them to offer targeted assistance to students [**?**]. In another study, Pereira et al. (2021) applied a recent XAI framework, SHAP, to elucidate the results of AI predictions [**?**].

C. Learning Analytics

Learning analytics involves gathering and examining the data produced by students as they interact with educational platforms. AI systems monitor various metrics including quiz scores, the duration spent on tasks, and how engaged students are with course materials. This information is then transformed into actionable insights, allowing instructors to pinpoint students' strengths and weaknesses and offer timely support. For example, if a student repeatedly struggles in a specific area, the AI system can suggest targeted practice exercises or ad- additional resources, illustrating an effective proactive approach to personalized learning.

D. Adaptive Testing

Adaptive testing leverages algorithms to continually assess a student's knowledge and skills. As students move through the assessment, the system modifies the questions based on their performance. This approach improves the accuracy of the assessment, ensuring that each question provides meaningful insights into the student's understanding. A prime example is the Graduate Record Examinations (GRE) adaptive test, which varies question difficulty according to the test-taker's previous responses, enabling a more precise evaluation of their abilities. AI platforms adapt instruction accordingly. For instance,if a learner is struggling with a particular language concept, the system might revise the content, adjust the complexity level, or change the learning mode to visual, auditory or kinesthetic depending on the learners' preference. In addition, AI platforms provide real-time feedback, identifying areas of strength and improvement [**?**], [**?**]. This instantaneous feed- back mechanism enables learners to improve their language skills more quickly and efficiently than through traditional feedback methods. The use of AI, therefore, provides a highly targeted, responsive, and effective approach to personalized learning, revolutionizing the way we view education and instruction.

III. RELATED WORK

Implementing AI in E-Learning platforms for analyzing students Data is very common, it can be used to retrieve insights on students learning type, their method of grasping information or their category, churn predictions to sell a course, study materials, boot camps etc., for segmentation of the students data into categories a unsupervised learning algorithms are used, they can be light weighted algorithms like gaussian mixture models [GMM], K-Means Clustering, K Nearest neighbors etc., neural network algorithms and Transfer learning has become much popular with the advantages of handling huge amount of training data and higher competency and robust performance to real world problems.

A. Manual Work

Tutoring involves teaching with simpler and understandable methods and analyzing whether the student has learnt what the things that are being taught, this is done with tests, assignments and project work. this can track a student's performance, adaptation, learning ability on a regular basis. this can be manual and not a personalized solution when it comes to the students. they strive to understand based on their own preferred source of education like live experiments, video lectures, reading books, solving problems etc.,

B. Automations

For tutoring systems the research exists for so many years with designing smart tutoring systems with Mamdani Fuzzy inference systems and LLM based approach on designing tutors, quantitative cognitive diagnosis systems. this can become a hectic and expensive as the usage becomes higher and higher. for this usage of monolith system architecture does not considered as a best option, microservices receives a greater appreciation to run the entire application as separate small services on their own contributing to achieve a single solution

IV. RESEARCH DESIGN

For the development of the system, the research starts with analyzing Edu Tech platforms that are being used by millions of students for personalized learning, another field is the CRMs i.e., customer relationship management systems where these types of software's analyses every movement, usage, likings and disliking, behavior with the product and they propose analytics on which product to sell to be sold to which user. the tutoring system with a personalized plan can be derived from the daily involvement of the user with the software, so our LLMs create a personalized learning dashboard within a certain short amount of time.

A. Database Design

The database is designed with MariaDB collections made with app write, the user authentication is managed by session based authentication, which implements the Argon 2 hashing algorithm, authentication of the user generates a unique identifier key which can be used in creating and manipulating data in the database with respect to the user. the collections defined in the database are

B. Adaptive Learner design

Adaptive learner design is built on the foundation of con- tenuous assessment and progressive content delivery, ensuring that each student receives a personalized learning experience. The system constantly evaluates a learner's knowledge level by analyzing test results, quiz attempts, and interaction pattern switch is continuously updated based on the student's test performance. If a student's proficiency score in a subject drops due to repeated incorrect answers, the system identifies related prerequisite topics that might be causing the difficulty and recommends a revision path to strengthen foundational know- edge. In addition, the adaptive system incorporates spaced repetition techniques, ensuring that students revisit weaker topics at optimal intervals to reinforce their understanding. The assessment engine dynamically adjusts the weightage of different topics, prioritizing those where improvement is needed while maintaining a balance with newly introduced subjects.

Beyond assessment-based adaptation, the system also cons- siders learning patterns and behavioral insights to refine content recommendations. If a student consistently struggles with specific question types, such as problem-solving or conceptual reasoning, the AI adjusts the teaching method accordingly—switching between visual, textual, or interactive explanations to find the most effective approach for that individual. Over time, the system constructs a personalized study trajectory, gradually shaping the learning experience to maximize retention, engagement, and conceptual mastery. By dynamically modifying test structures, adapting content complexity, and reinforcing weak areas, adaptive learner de- sign ensures that students' progress at their own pace while maintaining steady academic growth.

C. Rag model Pseudo code

With study materials. Every test score is recorded and categorized based on topic difficulty, question complexity, and accuracy rate, allowing the system to determine a learner's strengths and weaknesses. If a student performs exceptionally well in a particular area, the system gradually increases the complexity of future assessments, introducing more challenging concepts to stimulate deeper learning. Conversely, if test results indicate difficulty in understanding a topic, the system simplifies subsequent content, providing additional explanations, alternative problem-solving approaches, and supplement-tray learning resources such as videos or interactive exercises.

Algorithm 1 Adaptive Learning Path

Require: Student proficiency level, past performance
Ensure: Personalized learning path
Step 1 - Initialize *studentProfile*
Step 2 - Retrieve past *performanceData*
Step 3 - Identify *weakAreas* based on *performanceData*
Step 4 **- if** *weakAreas* ≠ Empty **then**
Step 5 - Recommend topics from *weakAreas*
Step 6 **- else**
Step 7 - Advance to the next module
Step 8 **- end if**
Step 9 - Generate adaptive assessment =0

The core mechanism of adaptive learning lies in real- time progress tracking and intelligent content adaptation. The system assigns a learning proficiency score to each topic,

Algorithm 2 Retrieval-Augmented Generation (RAG) Work- flow **Require:** Query from student, knowledge database
Ensure: AI-generated response with relevant context

Step 1 - Retrieve *query Embedding*
Step 2 - Search knowledge base for *relevant Documents*
Step 3 - Select top *k* most similar documents
Step 4 - Combine *query* with *retrieved Documents*
Step 5 - Generate response using *LLM*
Step 6 - Return generated response =0

D. Progress Tracking & insights

Progress tracking in adaptive learning monitors test scores, topic engagement, and learning patterns to provide real-time insights into student performance. If a learner struggles with a concept, the system recommends revisions and additional exercises, while strong performance leads to more advanced challenges. By continuously analyzing learning trajectories, the system ensures personalized growth, targeted improve- mends, and optimized study plans for each student.

E. Event Driven Architecture

An event-driven architecture using Apache Kafka enhances the efficiency of an AI-powered tutor service by enabling real-time data processing and asynchronous communication between system components. When a student completes a

Algorithm 3 Student Progress Tracking
Require: Student ID, performance history
Ensure: Continuous learning insights
Step 1 - Initialize *progressData*
Step 2 - Retrieve latest *testScores* and *completedModules*
Step 3 - Compute *averagePerformance*
Step 4 **- if** *averagePerformance* ¡ Threshold **then**
Step 5 - Identify weak areas
Step 6 - Recommend additional resources
Step 7 **- else**
Step 8 - Proceed to next learning phase
Step 9 **- end if**
Step 10 - Generate progress report =0

test, asks a question, or uploads study material, these actions generate events that Kafka captures and processes. The Kafka Producer publishes these events to specific topics like ”student- events”, while multiple Kafka Consumers listen for relevant updates. For instance, when a ”Test Completed” event is received, the system updates progress tracking, analyzes the student’s performance, and adjusts future lesson recommendations. Similarly, an event like ”Material Uploaded” trig- gets a document processing pipeline, ensuring newly added resources are indexed and made searchable for retrieval- augmented learning. By decoupling services and allowing real- time streaming, Kafka ensures that the tutor system remains scalable, responsive, and optimized for dynamic educational interactions.

Algorithm 4 Event Processing with Apache Kafka **Require:** Event stream from AI tutor system **Ensure:** Real-time processing and updates
Step 1 - Initialize Kafka Producer
Step 2 - Publish event to *student-events* topic
Step 3 - Initialize Kafka Consumer
Step 4 **- while** New event received **do**
Step 5 - Process event data
Step 6 **- if** Event type == ”Test Completed” **then**
Step 7 - Update progress tracking system
Step 8 **- else if** Event type == ”Material Uploaded” **then**
Step 9- Trigger document processing pipeline
Step 10 **- end if**
Step 11 **- end while**=0

F. Data Privacy

Ensuring robust data privacy in the AI tutor service, all user information is securely stored on App write, which provides end-to-end encryption and access control. Ar- gon2 hashing, an award-winning encryption algorithm from 2021, is used to protect user credentials, making unauthorized access nearly impossible. The system utilizes MariaDB, a reliable and secure database, to store structured learning data while maintaining high availability. To enhance security further, reverse proxy servers manage

AI API requests, preventing direct exposure of backend services. Google Gemini powers AI-driven responses, en- suring highly accurate and context-aware outputs while maintaining compliance with data protection regulations. This combination of secure storage, advanced hashing, and controlled API access guarantees the privacy, integrity, and confidentiality of student and tutor data.

G. Implementations

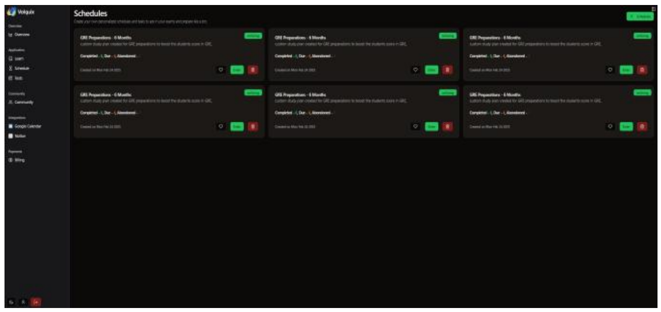

Fig. 1. AI-Assisted Study Schedule Management

a) *The scheduling interface enables users to create, track, and manage personalized study plans. It provides an overview of completed, due, and abandoned tasks, ensuring structured learning for optimized exam preparation:*

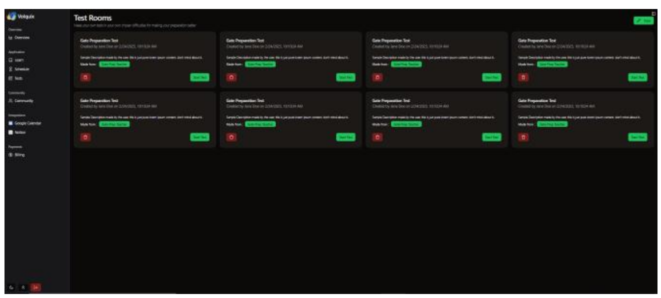

Fig. 2. AI-Powered Test Room for Exam Preparation

b) *The test room interface allows users to create, man- age, and take customized tests for structured learning. Each test entry includes metadata such as creation date, test source, and description, ensuring an organized approach to exam preparation.:*

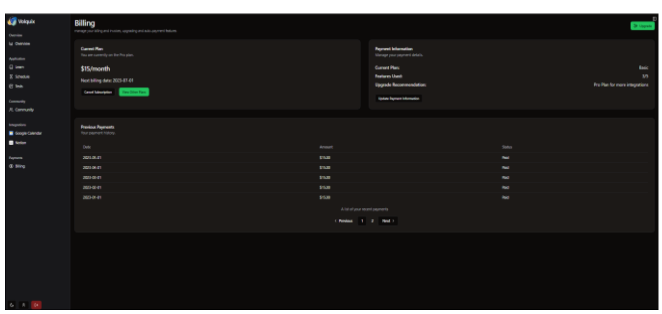

Fig. 3. Subscription and Billing Management Interface

c) *The* billing *dashboard provides users with an overview of their current subscription plan, payment history, and up- grade options. It ensures transparency in financial transactions by displaying past payments, next billing dates, and available features.*

V. LOW LEVEL DESIGN

Low-level design (LLD) is a crucial phase in the software development lifecycle that focuses on the internal architecture and components of an application. In building an application, the LLD phase breaks down the high-level features and requirements into specific, manageable modules and classes. It defines the structure, methods, data flow, and interactions between individual components within the system. This includes detailed specifications for databases, algorithms, data structures, and user interface elements.

For example, in a web application, the LLD will outline how individual components like APIs, controllers, views, and models interact. It will specify how data is handled through different layers, from the frontend to the backend, and how the database schema is structured for efficient querying and storage. Furthermore, LLD ensures that all edge cases are addressed and that error handling, logging, and security mechanisms (e.g., authentication and authorization) are well- implemented. By providing clear guidance for developers, low- level design serves as a blueprint for coding and testing and helps mitigate risks such as performance bottlenecks, data inconsistency, and security vulnerabilities.

In essence, low-level design forms the technical foundation upon which an application is developed, ensuring that it is both scalable and maintainable while also meeting functional and non-functional requirements.

A. Design Patterns

Django, Flask, and FastAPI applications due to its high-speed data retrieval and support for multiple data structures such as strings, lists, hashes, and sets. In a system handling teacher services, testing modules, and study plan generation, caching plays a vital role in reducing database load, minimizing API latency, and enhancing the overall user experience. By utilizing key-value storage, Redis ensures that frequently accessed data, such as study materials, quiz results, and student progress reports, can be retrieved instantaneously without repeatedly querying the backend database.

In a Django-based teacher service, caching helps store preprocessed lecture materials, AI-generated explanations, and frequently accessed course content. When a student requests a study topic, instead of querying the Post greSQL or Mon- goDB database, Redis serves the data directly from memory, cutting down on response time. Django provides built-in caching mechanisms through Django Cache Framework, al- lowing seamless integration with Redis. For dynamic content, template fragment caching stores portions of HTML templates, ensuring that elements such as navigation bars, recent activity feeds, and frequently used queries load instantly.

In the Flask-based testing service, caching improves performance by storing quiz structures, previous test attempts, and frequently used question banks. Redis enables session-based caching where user authentication tokens and exam progress are temporarily stored, reducing the need for repeated authentication checks. Flask's Flask-Caching extension integrates with Redis to implement function-level caching, ensuring that expensive computations, such as AI-powered test evaluations and scoring mechanisms, do not overload the server. Addition- ally, Redis supports publish-subscribe (Pub/Sub) mechanisms, enabling real-time notifications about test completion, grade updates, or upcoming assessments.

For the Fast API-driven study plan generator, caching optimizes the retrieval of personalized learning paths and rec-

Algorithm 5 Factory Pattern for API Services

Require: API request type
Ensure: Correct service instantiation
Step 1 **- function** SERVICEFACTORY(requestType)
Step 2 **- if** requestType == "User" **then**
Step 3 - Return *UserService*
Step 4 **- else if** requestType == "Auth" **then**

Step 5 - Return *AuthService*
Step 6 **- else**
Step 7 - Return ”Invalid request”
Step 8 **- end if**
Step 9 **- end function**
Step 10 - service = **ServiceFactory**(”User”)
Step 11 - service.**processRequest**() =0

B. Caching

Caching is a critical optimization technique in web ap- plications, significantly improving response times by storing frequently accessed data in memory. Redis (Remote Dictionary Server), an in-memory data store, is widely used for caching inommendation models. Since Fast API excels in asynchronous processing, Redis is used for storing precomputed AI model outputs, allowing study plans to be generated faster. Utilizing TTL (Time-To-Live) caching, Fast API ensures that temporary recommendations are refreshed periodically, preventing out- dated study plans. Redis also serves as a distributed cache, allowing multiple instances of the study plan generator to access shared cache data, ensuring consistency across user sessions.

A robust caching strategy with Redis involves implementing write-through caching for frequently updated data, where changes are written both to the cache and the database simultaneously. In contrast, lazy caching (cache-aside) fetches data only when necessary, reducing unnecessary cache storage. Additionally, Redis supports eviction policies, such as Least Recently Used (LRU) or Least Frequently Used (LFU), en- suring that memory is efficiently managed without excessive cache buildup.

Beyond API-level caching, Redis also enhances search functionality within the AI tutoring system. Storing pre- indexed search results in Redis drastically reduces lookup

Algorithm 6 Caching Mechanism for API Requests
Require: Request data, cache store (Redis)
Ensure: Faster API responses
Step 1 **- function**
Step 2 **- if** request in CACHE
Step 3 **- then** Return
Step 4 **- else**
Step 5 - response = Query Database
Step 6 - Return response
Step 7 **- end if**
Step 8 **- end function**
Step 9 - FetchData(”Get User Profile”) =0

Times, allowing students to retrieve relevant content instantly. Furthermore, session caching in Redis eliminates the need for repeated authentication, maintaining persistent user sessions across Django, Flask, and FastAPI services.

By integrating Redis caching into the teacher, testing, and study plan generator services, the system achieves faster response times, reduced server load, and improved scalability.

Stances. For Django’s teacher service, multiple Gunicorn work- ers are deployed to manage concurrent requests, with Nginx distributing the load among them. Flask’s tester service, being lightweight, is deployed using uWSGI or Gunicorn, where Nginx ensures an even traffic flow. Fast API, known for its asynchronous capabilities, runs multiple Uvicorn workers, al- lowing high-speed request handling. When a student accesses the AI tutor, Nginx directs the request to the corresponding service based on predefined routing rules. If a teacher-related request is detected, it is routed to Django; if it pertains to a test or quiz, Flask handles it; and if it involves study plan generation, Fast API processes the request.

To implement load balancing, Nginx upstream configu- rations define backend servers running these services. The round-robin method ensures that requests are distributed se- quentially, whereas the least connections method directs traffic to the least burdened server. If one instance becomes un- responsive, Nginx automatically reroutes traffic to a healthy instance, ensuring fault tolerance. Additionally, rate limiting and caching mechanisms within Nginx reduce latency, improv- ing response times for frequently accessed data such as quiz results, study plans, and educational resources.Caching ensures that AI-powered tutoring remains efficient,seamless, and responsive, ultimately enhancing the student learning experience while optimizing backend performance.

C. Load Balancers

Load balancing is an essential mechanism in distributed applications to ensure high availability, scalability, and optimal resource utilization. In a system handling multiple micro services, such as a teacher service, tester service, and study plan generator, load balancing efficiently distributes incoming requests among multiple application instances. Nginx, a high-performance reverse proxy and load balancer, plays a critical role in handling these API services by evenly distributing traffic, preventing server overload, and enhancing fault tolerance. By leveraging round-robin, least connections, or IP hash algorithms, Nginx ensures that each request is routed to the appropriate backend service, optimizing response times and server health.

In a typical micro service-oriented architecture, where Django powers the teacher service, Flask manages the tester service, and FastAPI runs the study plan generator, Nginx is configured to balance the traffic between multiple instances of each service. The teacher service, built with Django, handles functionalities such as retrieving study materials, assisting in lectures, and generating AI-powered explanations. The tester service, developed using Flask, is responsible for creating, managing, and grading assessments. Meanwhile, the study plan generator, powered by FastAPI, dynamically designs personalized learning schedules based on student progress. Since these services cater to different workloads and user requests, an efficient load balancing strategy prevents perfor- mance bottlenecks.

Nginx operates as a reverse proxy, receiving client requests and forwarding them to the appropriate backend server in-

Algorithm 7 Load Balancer for Microservices

Require: Incoming API Requests
Ensure: Efficient request distribution
Step 1 - Initialize *ServerList* with active servers
Step 2 - **function** DistributeRequest(request)
Step 3 - selectedServer = **RoundRobin**(ServerList)
Step 4 - Forward request to selectedServer
Step 5 - Return response
Step 6 - **end function**
Step 7 - DistributeRequest(”User Login”) =0

By integrating SSL termination, Nginx secures communication between clients and the backend services, encrypting sensitive student and teacher data. It also compresses API responses, reducing bandwidth usage and enhancing efficiency. Moreover, with WebSocket support, real-time updates on test results and study plan modifications can be efficiently managed. The implementation of auto-scaling policies allows dynamic provisioning of additional backend instances during peak usage, ensuring seamless operation even during high- traffic periods such as exam seasons.

The combination of Nginx load balancing, Django, Flask, and FastAPI ensures that the AI tutoring system remains scalable, reliable, and optimized. By effectively distributing traffic, handling failures gracefully, and improving request efficiency, Nginx plays a crucial role in enhancing the teacher, tester, and study plan generator services, thereby creating a robust and efficient AI-powered educational ecosystem.

VI. HIGH-LEVEL DESIGN

The high-level design of the application involves three core services, each dedicated to specific functionalities, along with the management of authentication and data storage. The architecture incorporates multiple technologies and frameworks to ensure a scalable, maintainable, and efficient system.

A. Cloud hosted Functions as a Service

The function as a service was built using Python and lever- aged Gemini, an AI model, to provide advanced functionalities within the system. This service is responsible for executing key application functions such as processing user inputs, triggering events, and managing task executions. Flasks offer lightweight and flexible handling of requests, ensuring that the service is fast and responsive.

- **Timeline Generator Service**: Built using Flask, this service generates personalized timelines for users based on their input. It uses Gemini to generate intelligent predictions and dynamic content. The service employs a highly efficient backend to process data, thereby ensuring that users receive accurate and timely information.

- **Teacher Service**: This service, built with Django and powered by Langchain, Chroma DB (a vector database), and Gemini, is responsible for handling teacher-related operations. Using the Retrieval-Augmented Generation (RAG) approach, the service leverages the power of Chroma DB for storing and retrieving embeddings and Langchain for managing workflows. Gemini is used for generating responses based on the retrieved information, enhancing the service's performance by offering context- aware and relevant content to teachers.

- **Tester Service**: Built using Fast API, Cassandra DB, Lang chain, and Gemini, the tester service is focused on testing and validating various components of the application. Cassandra DB offers high scalability and availability, making it suitable for handling large amounts of data generated during testing. Lang chain orchestrates the logic of the testing workflows, whereas Gemini pro- vides intelligent capabilities to analyze and generate test results in real time.

B. Authentication and Backend Management

Authentication is managed by App write, a Backend-as-a- Service (BaaS) solution. App write handles user authentication, cloud storage, and SQL databases for structured data management. Additionally, App write functions are used to execute the server-side logic. The system also incorporates Gzip compression to ensure efficient file storage and faster transmissions.

C. Containerization and Scalability

The entire application was containerized using Docker to ensure scalability and ease of deployment. Each of the services (function and timeline generator, teacher, and tester services) runs in isolated Docker containers, providing portability and consistency across different environments. Docker's lightweight nature allows us to quickly spin up or scale down services based on demand, ensuring that we can handle varying workloads efficiently. Containerization also supports a micro service architecture, where each service is independently deployable and upgradable. This modular approach allows easy scaling and rapid iteration of specific components without affecting the entire system. This architecture provides a com- prehensive and modular approach, allowing each service to be independently scalable, while ensuring seamless integration across applications. Each component is designed to handle specific tasks with minimal overhead, ensuring a smooth and responsive user experience.

VII. IMPROVING UI/UX

To enhance the user experience and ensure that students are motivated, engaged, and productive, our application leverages a variety of advanced UI/UX design principles. These include micro-interactions, progress charts, an awards-based learning system, and a gamified experience, all of which work in tandem to create an engaging, intuitive, and effective learning environment.

1) *Micro-Interactions:* Micro-interactions play a crucial role in improving the overall user experience by providing sub- tle yet impactful feedback for user actions. In our application, micro-interactions are used throughout the interface to guide students in their learning journey. For example, when a student completes a task or quiz, a smooth animation and a brief confirmation message appear, confirming their achievement. These micro-interactions help to keep users informed and engaged without overwhelming them, providing a sense of progression and accomplishment.

2) *Progress Charts:* Progress tracking is an essential fea- ture for motivating students and keeping them on track with their learning goals. Our application integrates visually appeal- ing progress charts that display key metrics such as completed lessons, quiz scores, and overall course performance. These charts are interactive and provide students with real-time insights into their progress. By offering a clear visualization of their achievements, students are encouraged to continue their efforts, improving their engagement and productivity. Additionally, the progress charts are customizable, allowing students to focus on the areas where they need improvement.

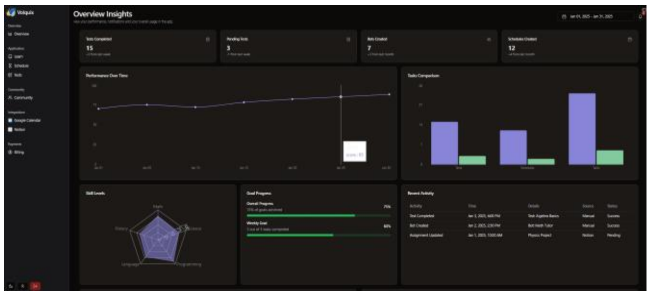

Fig. 4. Overview Dashboard of AI-Assisted Tutoring System : The dashboard provides insights into user performance, test completion, and scheduling activities. It visualizes performance trends, skill levels, task comparisons, and goal progress, enabling data-driven tutoring enhancements.

3) *Awards-Based Learning System:* To further drive mo- tivation and engagement, the application incorporates an awards-based learning system. As students complete various milestones, such as finishing lessons, completing assignments, or scoring high on quizzes, they earn badges, certificates, or other forms of recognition. These awards not only provide students with a sense of achievement but also create an environment where learning is rewarded. The awards-based system encourages students to maintain consistent effort, promoting continuous learning and improvement.

4) *Gamified Experience:* Our application integrates gamification techniques to create an interactive and enjoyable learn- ing experience. Students can unlock achievements, progress through levels, and compete in challenges that align with their learning goals. The gamified experience includes ele- ments such as leaderboards, time-based challenges, and re- ward systems that make the learning process both fun and productive. By transforming the learning environment into a game-like experience, students are more likely to stay engaged, motivated, and driven to complete their learning tasks. This gamification approach fosters a sense of friendly competition and achievement, turning learning into an exciting and rewarding adventure.

5) *Conclusion:* By incorporating micro-interactions, progress charts, an awards-based learning system, and a gamified experience, our application aims to create a more engaging and rewarding educational environment. These design features not only enhance the overall UI/UX but also contribute to boosting student productivity, motivation, and long-term engagement with the learning content.

Acknowledgment

We would like to extend our sincere thanks to all the indi- viduals and organizations who have supported the development of our application. The successful completion of this project would not have been possible without the collaborative efforts of our talented team and the invaluable contributions of our stakeholders.

Special thanks to the research community for their con- tinued advancements in AI, machine learning, and education technology, which served as the foundation for many of the features in our app. We also appreciate the extensive resources provided by Appwrite, LangChain, and other third- party services that enabled us to implement powerful backend functionalities, such as authentication, data storage, and inte- gration with AI models.

Additionally, we acknowledge the support of our aca- demic advisors and mentors, whose guidance has been crucial throughout the development process. Their insights into ed- ucational methodologies and user-centered design principles were instrumental in shaping the app's features, such as the personalized learning experience and real-time feedback system.

Lastly, we thank our users for their feedback, which has driven the continuous improvement of the app. Their input has been invaluable in refining the user interface, enhancingthe gamification aspects, and optimizing the overall user experience.

VIII. CONCLUSION

The integration of Retrieval-Augmented Generation (RAG) into educational AI systems marks a significant advancement in personalized learning. This research introduces an AI- powered tutor that enables students to interact with educational content dynamically, offering simplified explanations, intelligent question answering, and adaptive assessments. By leveraging AI-driven retrieval and generative techniques, the system personalizes learning experiences, tracks progress, and provides data-driven study plans to optimize student outcomes. Beyond traditional learning methods, this approach fosters self-paced education, ensuring that learners receive tailored guidance based on their individual needs. The ability to generate customized tests with adjustable difficulty levels and grading mechanisms enhances engagement and knowledge retention. As AI-driven education continues to evolve, the proposed system sets a foundation for more adaptive, inter-active, and intelligent learning platforms, bridging gaps in comprehension and making quality education more accessible.

REFERENCES

[1] Agarwal, S., Laradji, I. H., Charlin, L., & Pal, C. (2024). LitLLM: A Toolkit for Scientific Literature Review. arXiv preprint arXiv:2402.01788.https://arxiv.org/abs/2402.01788

[2] Alshammari, S., Basalelah, L., Rukbah, W. A., Alsuhibani, A., & Wijesinghe, D. S.(2023). KNIMEZoBot: Enhancing Literature Review with Zotero and KNIME OpenAIIntegration using Retrieval-Augmented Generation. arXiv preprint arXiv:2311.04310.https://arxiv.org/abs/2311.04310

[3] Arora, D., Kini, A., Chowdhury, S. R., Natarajan, N., Sinha, G., & Sharma, A.(2023). Gar-meets-rag paradigm for zero-shot information retrieval. arXiv preprintarXiv:2310.20158.https://arxiv.org/abs/2310.20158

[4] Boyd, J. (2015). Automatic metadata extraction - The high energy physics use case(Master's thesis, Ecole Polytechnique, Lausanne). Retrieved from https://cds.cern.ch/record/2039361. Presented 04 Sep 2015.

[5] S. Ranjithkumar, G. Nallavan, and S. Sugumar, "Implementation of an Intelligent Real-Time Squat Detection and Performance Evaluation System Using Computer Vision and Pose Estimation Techniques," Iconic Res. Eng. J., vol. 9, no. 6, 2025. https://doi.org/10.64388/IREV9I6-1712786

[6] Duˇsek, R., Galias, C., Wojciechowska, L., & Wawer, A. (2023, September). Improving domain-specific retrieval by NLI fine-tuning. In 2023 18th Conference on Computer Science and Intelligence Systems (Fed CSIS) (pp. 949-953). IEEE. https://doi.org/10.15439/2023F7569

[7] Es, S., James, J., Espinosa-Anke, L., & Schockaert, S. (2023). Ragas: Automatedevaluation of retrieval augmented generation. arXiv preprint arXiv:2309.15217.https://arxiv.org/abs/2309.15217

[8] Gao, T., Yen, H., Yu, J., & Chen, D. (2023). Enabling large language models to generatetext with citations. arXiv preprint arXiv:2305.14627.https://arxiv.org/abs/2305.14627

[9] Gunasekar, S., Zhang, Y., Aneja, J., Mendes, C. C. T., Del Giorno, A., Gopi, S., ...& Li, Y. (2023). Textbooks are all you need. arXiv preprint arXiv:2306.11644.https://arxiv.org/abs/2306.11644

[10] Han, W., Shen, J., Liu, Y., Shi, Z., Xu, J., Hu, F., ... & Ge, M. (2024). LegalAsst: Human-centered and AI-empowered machine to enhance court productivity and legal as-sistance. Information Sciences,679, 121052. https://doi.org/10.1016/j.ins.2024.121052

[11] Sentence Transformers. (2021). all-mpnet-base-v2 [Pre-trained model]. Hug-ging Face. Retrieved from https://huggingface.co/sentence-transformers/all-mpnet-base-v229

[12] Izacard, G., & Grave, E. (2020). Leveraging passage retrieval with generative modelsfor open domain question answering. arXiv preprint arXiv:2007.01282.https://arxiv.org/abs/2007.01282

[13] Kaesberg, L. B., Ruas, T., Wahle, J. P., & Gipp, B. (2024). CiteAssist: A system forautomated preprint citation and BibTeX generation. arXiv preprint arXiv:2407.03192.https://arxiv.org/abs/2407.03192[14] Kamradt, G. (2024). Semantic chunking. FullStack Retrieval Tutor