Advice

Lokesh Agnihotri

Contents

Abstract:
This document provides a comprehensive analysis of the technologies and software frameworks for developing a Pronunciation Trainer Application. It explores various options and considers essential factors such as availability, performance, security, and scalability. Based on the evaluation, this document offers recommendations for selecting the most suitable technologies. It covers front-end technologies, back-end frameworks, programming languages, and their comparisons, ensuring informed decision-making to ensure functionality and effectiveness of the application.

Introduction:

The Pronunciation Trainer Application is designed to assist users in improving their pronunciation skills. **In the current project we need to consider multiple technical aspects.**

1. **UI/UX**: We need to provide our users with a unified interface to access our services. We foresee that the UI/UX should be simple, intuitive, and robust as this is the primary method to access and experience our application.
2. **Business Logic/Service**: This is the heart of our application as it is required to serve requests originating from our UI/UX and provided relevant feedback after processing the incoming data. We foresee that Service function needs to be Scalable, Robust and Fault Tolerant to ensure optimum experience for the users.
3. **Cloud Components**: These are the third-party services which are being accessed by our Service/Logic to process data and derive meaningful results. We foresee that this component must be compatible with multiple service providers, and robust to third party failures.
4. **AI For working with Text and Speech**: This is an orthogonal aspect with the development as it provides the key methodology which will be used by our application to provide relevant services. It is important to understand the various existing approaches and their strengths and weaknesses. It is prudent that we choose a cost effective, scalable and including solution to be able to provide effective services to our users.

These aspects are further discussed in detail with the aim of evaluating the existing technologies, framework, methodologies to obtain desired result. In the following pages, we will focus on following aspects of the application.

1. AI Methodologies
2. Front-End Development
3. Backend Infrastructure
4. Programming Languages

AI Methodologies:

Before delving into the AI part and categorizing the activities, it is essential to explore the different methods of evaluating pronunciations and providing feedback. By understanding these approaches, we can select the most suitable one and then proceed with choosing the appropriate libraries, models, and implementation strategies to achieve our goals effectively.

Exploring Pronunciation Assessment Methods:

Human Evaluation: Traditional method involving human experts who assess and provide feedback on pronunciation accuracy manually. This approach offers subjective analysis but can be time-consuming and resource intensive.

Automated Speech Recognition (ASR): Utilizes algorithms and models to convert spoken language into written text. ASR systems can be trained to evaluate pronunciation accuracy by comparing input speech with reference pronunciations. This method provides objective assessments but may require extensive training data and fine-tuning.

Acoustic Analysis: Involves analyzing audio signals to extract acoustic features and measure pronunciation quality based on various metrics such as pitch, intensity, and formants. This method provides quantitative insights into pronunciation but may require advanced signal processing techniques and domain-specific knowledge.

Phonetics-based Approaches: Focuses on phonetic aspects of pronunciation, examining phoneme-level accuracy, stress patterns, intonation, and rhythm. This method requires phonetic expertise and can provide detailed feedback on specific pronunciation aspects.

Considering the information provided, Phonetics-based methods are the recommended choice over the other two methods. This is because the alternative approaches entail extensive fine-tuning and advanced signal processing techniques, both of which can be time-consuming.

Front-end Technologies

Available technologies

- **HTML** (Hypertext Markup Language): A standard markup language for creating the structure and content of web pages.
- **Angular**: A TypeScript-based open-source framework developed by Google for building dynamic web applications.
- **React**: A JavaScript library developed by Facebook for building user interfaces, focusing on component-based development.
- **Vue.js:** A progressive JavaScript framework for building user interfaces, known for its simplicity and ease of integration.

Comparison

Technology	**Cost**	**Availability**	**Performance**	**Security**	**Scalability**	**Learning Curve**
HTML, CSS, JavaScript	Free and open source	Widely	Light and fast rendering	Secure	Scalable for small projects	Low
Angular	Free and open source	Wide community Support	Optimized for high performance	Strong Security Features	Highly	Medium
React	Free and open source	Wide community Support	Efficient rendering	Strong Security Features	Highly	Medium
Vue.js	Free and open source	Wide community Support	Fast rendering	Strong Security Features	Highly	High

Conclusion

Though, Angular, React, and Vue.js offer their own advantages in terms of performance, community support, and scalability. However, they may introduce additional complexity and a steeper learning curve, making them more suitable for larger-scale projects where the benefits outweigh the associated costs.

Considering the cost aspects, availability, performance, security, and scalability, HTML, CSS, and JavaScript are recommended for the front-end development, ensuring a cost-effective, widely supported, performant, and secure solution that can be scaled for small to medium-sized projects.

Back-end Technologies

Available technologies

Flask: A lightweight and flexible Python-based web framework that offers simplicity, compatibility, and rapid development capabilities.

Django: A high-level Python web framework that provides a robust set of tools and features for building web applications.

Express.js: A fast and minimalist web application framework for Node.js, known for its simplicity and flexibility.

Ruby on Rails: A full-stack web application framework written in Ruby that emphasizes convention over configuration and promotes rapid development.

Laravel: A PHP-based web framework that offers a clean and elegant syntax, along with powerful tools and features for building scalable applications.

ASP.NET: A web framework developed by Microsoft that allows developers to build dynamic web applications using .NET programming languages such as C#.

Node.js: A runtime environment that allows developers to run JavaScript on the server-side, providing a scalable and efficient backend solution.

Comparison

Technology	Language	Framework	Cost	Availability	Performance	Security	Scalability	Le Cu
Flask	Python	Light Weight	Low	High	Good	Good	Good	Lo
Django	Python	Full Featured	Medium	High	Excellent	Excellent	Excellent	Lo
Express.js	JavaScript	Minimalist	Low	High	Good	Good	Good	Me
Ruby On Rails	Ruby	Full featured	Medium	High	Good	Good	Good	Hi
Laravel	PHP	Full featured	Low	High	Good	Good	Good	Me
ASP.NET	C#J	Full Featured	High	High	Excellent	Excellent	Excellent	Hi
Node.js	JavaScript	Runtime	Low	High	Excellent	Good	Excellent	Me

In this comparison table, the technologies are evaluated based on the following parameters:(Deed, 2023): The ability of technology to handle increased workloads and scale with growing demands.

Conclusion

Flask is recommended as the backend technology for the Pronunciation Trainer App due to its cost-effectiveness, availability, good performance, security features, and scalability capabilities. Despite other options like Django, Express.js, Node.js, Ruby on Rails, and Laravel offering excellent features, Flask proves to be a more suitable choice. It aligns well with the project's

budget, provides extensive resources and community support, exhibits satisfactory performance, and offers adequate security measures. Flask's lightweight nature, simplicity, and compatibility with Python make it an optimal choice for the Pronunciation Trainer App's development, outweighing the higher costs and complexities associated with other technologies.

Programming Language

Languages:

Python: versatile and powerful language known for its simplicity, readability, extensive library support, and strong community, making it ideal for rapid application development and integration of various functionalities.

Ruby: dynamic, object-oriented language known for its simplicity and developer-friendly syntax, often used in web development with frameworks like Ruby on Rails.

C#: versatile language developed by Microsoft, commonly used for building Windows applications, web services, and game development.

Go: also known as Golang, is a statically typed language designed for efficiency and scalability, with built-in support for concurrent programming, making it suitable for networked and distributed systems.

Swift is a programming language developed by Apple, designed for building applications for iOS, macOS, watchOS, and tvOS, known for its safety, performance, and modern syntax.

Comparison of Programming Languages

Criteria	**Python**	**Ruby**	**C#**	**GO**	**Swift**
Simplicity	High	High	Moderate	Moderate	High
Library Support	Extensive	Moderate	Extensive	Moderate	Moderate
Community	Strong	Active	Active	Active	Active
Integration	Excellent	Good	Good	Excellent	Excellent
Scalability	Good	Moderate	Good	Excellent	Excellent
Learning Curve	Low	High	Medium	High	High

Conclusion:

Based on the comparison above, Python has been selected as the primary programming language for the Pronunciation Trainer App. Python stands out for its simplicity, extensive library ecosystem, developer productivity, and strong community support, making it an excellent choice for rapid application development. Its integration capabilities are well-suited to the app's requirements, allowing for easy incorporation of speech recognition libraries and other necessary functionalities.

While Ruby, PHP, C#, Go, and Swift are also noteworthy programming languages, they do not offer the same level of simplicity, extensive library support, and community engagement found in Python. Each language has its own merits and strengths, suited to specific use cases. However, when considering the requirements of the Pronunciation Trainer App, Python proves to be the optimal choice. It strikes a balance between functionality, development efficiency, and integration capabilities, making it the most suitable option for the project.

Conclusion

Based on the analysis and comparisons, it is recommended to use HTML, CSS, and JavaScript for the front-end development of the Pronunciation Assessment Application. Flask is suggested as the back-end framework, and Python as the primary programming language. These choices ensure simplicity, compatibility, rapid development, extensive library support, and integration capabilities. By considering factors such as availability, performance, security, and scalability, these technologies provide a solid foundation for creating a user-friendly, feature-rich, and efficient Pronunciation Assessment Application.

Software development process:

The software development process is of utmost importance in ensuring the success of a project. After carefully considering the project requirements, the next step is to define a robust development process. To strike a balance between productivity and adaptability, it is advisable to adopt a 3-day sprint basis for software development. This approach aligns with the principles of iterative development and frequent feedback, allowing for timely course corrections and adjustments as necessary.

During each sprint, it is imperative to include a comprehensive testing process to uphold the quality and reliability of the software. Two types of tests that should be incorporated into the development process are unit tests and user tests.

Unit tests are specifically designed to verify the functionality of individual components or units of code within the software. By conducting these tests, bugs, errors, and unexpected behaviors can be identified and rectified at an early stage, ensuring a stable and error-free codebase.

User tests, on the other hand, involve gathering feedback from actual users of the software. This feedback provides valuable insights into the usability, user experience, and any potential shortcomings of the product. By regularly conducting user tests, developers can gather actionable feedback early in the development process, enabling them to make informed decisions and improve the software based on user needs and preferences.

Advice for Data and Privacy

In the context of the current application, which does not require user sign-in or store any user-specific sensitive or non-sensitive information in a database, privacy concerns are irrelevant. However, it is prudent to consider potential future scenarios where the application may gain widespread popularity and attract a large user base.

In such a scenario, it is advisable to implement a load balancer to efficiently handle the increased traffic from a substantial number of users accessing the pronunciation trainer. A load balancer distributes incoming network traffic across multiple servers, helping to optimize performance and prevent overload. By proactively addressing potential scalability challenges, It can be ensured, a seamless user experience even with a significant increase in user activity.

Conclusion

Based on the analysis and comparisons conducted, the recommended approach for the development of the Pronunciation Assessment Application involves utilizing HTML, CSS, and JavaScript for the front-end development. These technologies offer simplicity, compatibility, and extensive library support, ensuring the creation of a user-friendly and visually appealing interface. In addition, Flask is suggested as the back-end framework, leveraging the power of Python as the primary programming language. By adopting a 3-day sprint basis for the software development process, a balance between productivity and adaptability can be achieved. This iterative approach, along with frequent feedback, allows for course corrections and adjustments as needed, ultimately ensuring the success of the project. Considering factors such as availability, performance, security, and scalability, these technology choices provide a solid foundation for the development of a Pronunciation Assessment Application that is efficient, feature-rich, and capable of meeting user needs.

Bibliography

Deed, I. (2023, 05 08). *Highest Performing Web Framework Benchmarks.* Retrieved from pangea.ai: https://www.pangea.ai/resources/web/

Fabien, M. (n.d.). *Introduction to ASR.* Retrieved from https://maelfabien.github.io/machinelearning/speech_reco/#evaluation-metrics

Gandhi, Y. (2022). Automatic Speech Recognition: Types and Examples. Analytic Steps. Retrieved from https://www.analyticssteps.com/blogs/automatic-speech-recognition-types-and-examples

Karczewski, D. (2022, December 27). Retrieved from Ideamotive: https://www.ideamotive.co/blog/best-frontend-frameworks

Klatt, D. H. (1978). *Speech perception: a model of acoustic-phonetic.* Retrieved from https://pdf.sciencedirectassets.com/272464/1-s2.0-S0095447019X34002/1-s2.0-S0095447019310599/main.pdf?X-Amz-Security-Token=IQoJb3JpZ2luX2VjEGUaCXVzLWVhc3QtMSJIMEYCIQD9busBokYN22OEBa5Yb8JKXwhV2Q3H6lv%2BBDrqNsbgcQIhAIdi%2BmzYmUCDEO4Ppx%2BBFIH%2FXGeBkstL8bpY

Papastratis, I. (2022, 07 12). *Speech Recognition: a review of the different deep learning approaches.* Retrieved from https://theaisummer.com/speech-recognition/

Parabattina, B. &. (2016). Acoustic Phonetic Approach for Speech Recognition: A Review. .

Tiobe. (2023, 06). *Tibe Index.* Retrieved from Tiobe Index: https://www.tiobe.com/tiobe-index/

Veerarraghavan, S. (2023, 06 1). Retrieved from https://www.simplilearn.com/best-programming-languages-start-learning-today-article

Classified

Classified