Table of Contents

1. Problem

1.1. Background Research

The challenges faced by individuals with visual impairments in accessing reading materials remain significant and multifaceted. Despite advancements in assistive technologies such as Braille, large print, and digital formats, these resources are not universally available. For instance, less than **10%** of books are accessible to blind individuals, which highlights a critical gap in the availability of reading materials tailored for this population. Furthermore, the American Printing House for the Blind reports that only **63,357 blind students** requested free reading materials in Braille, large print, or audio format, indicating that demand far exceeds supply in many educational settings.

Additionally, while Braille literacy is essential for many blind individuals, the current statistics reveal a troubling trend. Estimates suggest that only **10-25%** of blind people know how to read Braille. The situation is even more dire, with many regions lacking adequate resources for teaching and providing Braille materials in their own regional language. The high cost of Braille displays—often running into thousands of dollars—further limits access.

To address these challenges materials in large print, Braille, or digital formats allowing users to customize text size, contrast, and background colour according to their individual needs were introduced. However, the availability of these materials is also insufficient; a study indicated that **69.9%** of libraries did not have readily available Braille materials or large print options for visually impaired users. This scarcity perpetuates a cycle of exclusion and limits opportunities for education and employment among those with visual impairments. As technology advances and awareness grows, it is crucial to address these disparities by ensuring that all individuals with visual impairments have equal access to reading materials and educational resources that meet their needs.

1.2. Problem Statement

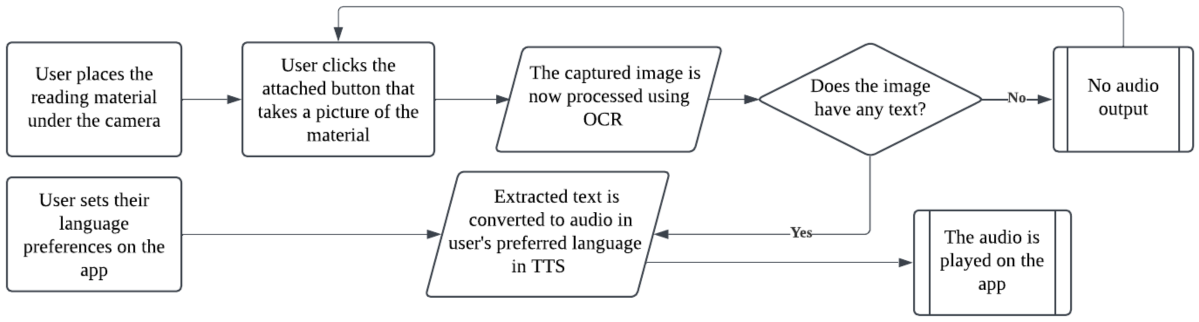
Thus, to solve this problem, we came up with the following Problem Statement:
"How might we empower individuals with partial visual impairment to access diverse and inclusive reading materials in a way that is efficient, accessible, and personalized to their needs?"

2. Solution Overview

Our solution is an innovative and user-friendly reading device designed to empower individuals with visual impairments. This device allows users to capture images of printed text effortlessly, which can then be processed by a dedicated app that converts the text into audio. Users can select their preferred language for the audio output, ensuring a personalized and accessible experience tailored to their needs.

We believe this solution will effectively address the significant challenges faced by individuals with visual impairments in accessing reading materials. By combining advanced image recognition technology with multilingual audio capabilities, our device offers a scalable and practical approach to enhancing literacy and information access. This initiative not only promotes independence among users but also fosters inclusivity, enabling them to engage more fully with written content in a way that suits their preferences.

2.1 User Flow


User places the reading material under the camera
User clicks the attached button that takes a picture of the material
The captured image is now processed using OCR
Does the image have any text?
No
No audio output
User sets their language preferences on the app
Extracted text is converted to audio in user's preferred language in TTS
Yes
The audio is played on the app

This diagram illustrates the complete flow of our system. It starts from the process of capturing text from physical reading material using a camera, processing it with OCR (Optical Character Recognition) to extract text, converting the text to audio in the user's preferred language via TTS (Text-to-Speech), and playing the audio on the app. If no text is detected, no audio output is provided.

2.2 Functional Requirements

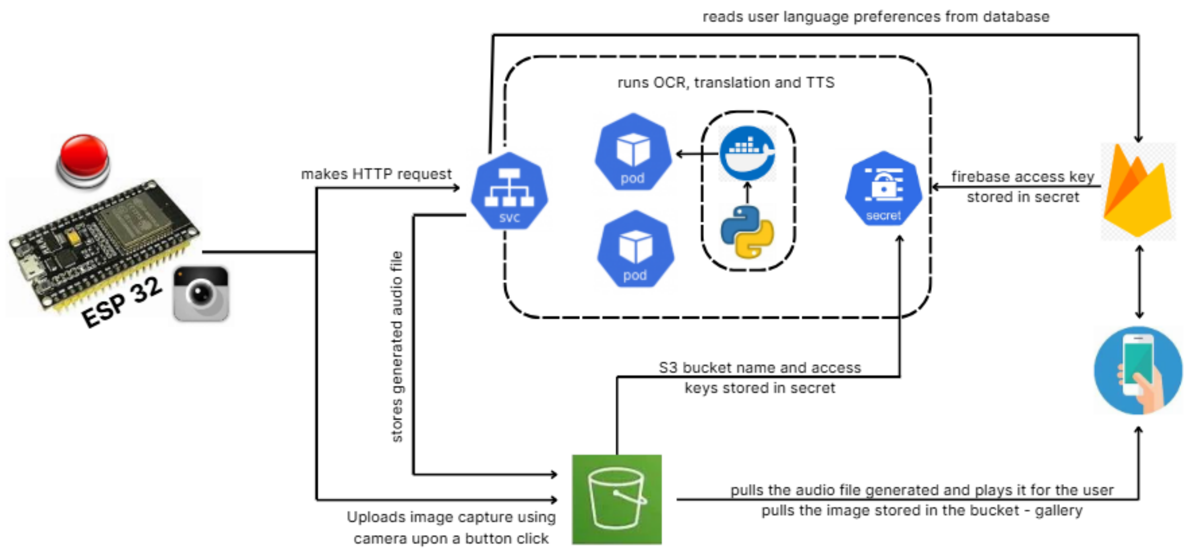
After understanding the flow of our application and its features, we now look at the functional requirements needed for the same. They are:

Image Capture: The system should allow users to capture an image of a document using a camera module and a button setup. The button press should trigger the image capture.
Text Processing: The system should process the captured image using Optical Character Recognition (OCR) to extract text. It should translate the extracted text into the user's preferred language, as stored in the database.
Audio Generation: The system should convert the processed text to audio using Text-to-Speech (TTS) and store the generated audio file in a database.
Data Storage and Retrieval: The system should allow users to retrieve stored images (for a gallery feature) and the generated audio file should be retrieved and played for them via the application.
User Preferences: User language preferences and login credentials should be fetched securely in a database. The user should be able to change these as and when necessary. These changes should also be reflected on our database.

3. System Design

After understanding the functional requirements, we thought about how this could be best achieved and came up with our system design and architecture which is discussed in this section.

3.1. System Architecture


reads user language preferences from database
runs OCR, translation and TTS
makes HTTP request
svc
pod
pod
secret
firebase access key stored in secret
ESP 32
stores generated audio file
S3 bucket name and access keys stored in secret
pulls the audio file generated and plays it for the user
pulls the image stored in the bucket - gallery
Uploads image capture using camera upon a button click

The system architecture leverages an ESP32 module for capturing images and making HTTP requests to a backend service. The backend processes the image using OCR, handles text translation, and converts the text to audio using TTS. Outputs (audio files and images) are stored in an S3 bucket, and a mobile app is used for playback and gallery access. Firebase is integrated for secure storage of user preferences, with credentials managed via secrets for enhanced security.

We believe that this architecture is scalable, robust and ensures the system is both reliable and future-proof, with the ability to adapt to evolving user needs and demands.

Scalability:

- The architecture is cloud-based, utilizing scalable components such as S3 buckets for storage and containerized services (pods) for processing OCR, translation, and TTS. This ensures the system can handle increasing workloads (e.g., more users or larger files) by scaling up services and storage as needed.
- Using container orchestration (e.g., Kubernetes for pods) allows horizontal scaling, ensuring efficient resource utilization.

Security:

- Secure management of sensitive data (e.g., Firebase and S3 credentials) using secrets ensures the system remains reliable and protected against unauthorized access.

Robustness:

- Fault tolerance is enhanced through the use of distributed storage (S3 bucket) and containerized processing, minimizing the impact of individual component failures.
- The modular design, with distinct roles for ESP32, backend services, and mobile apps, enables easier debugging and system maintenance.

3.2. Kubernetes Component

We use Kubernetes to orchestrate and manage our containerized backend services, that processes the image and generates the audio file in user preferred language by running OCR, language translation and TTS.

Load Balancer Service (SVC): Exposes the cluster to handle public requests, ensuring external access to our backend services (OCR, translation, TTS). Load balancers distribute traffic across replicas, maintaining high availability and preventing overload on a single pod.

Replicas of Pods: Each pod is deployed with 2 replicas for redundancy and fault tolerance. If one pod fails, the second ensures uninterrupted service, enhancing reliability.

Secrets: Manage sensitive information, such as S3 bucket access keys and Firebase credentials. Protecting sensitive data ensures security and prevents unauthorized access.

By leveraging Kubernetes, the system ensures scalability, reliability, and security for backend operations.

3.3. Cloud Component – Multi Cloud Providers

We utilize **AWS S3** for storage and **Firebase** for authentication.

AWS S3: Stores images captured by the ESP32 and the generated audio files. S3 provides scalable, durable storage, ensuring data availability and accessibility.

Firebase: Used for user authentication and database access. Firebase simplifies authentication setup and securely manages user preferences and data.

Multi-Cloud Approach: Combines AWS and Firebase for their respective strengths. This enhances reliability, flexibility, and cost effectiveness.

3.4. Mobile App Component

We use a mobile app to interact with users and provide outputs. We built the mobile app using React Native and Expo making it compatible with both iOS and Android.

It has the following functionalities:
Language Preference Setting: Allows users to customize audio output in their preferred language.
Audio Playback: Provides a platform to play the audio files generated by TTS.
Gallery Access: Retrieves stored images from the S3 bucket for user convenience.

A mobile app offers an intuitive and portable interface for seamless user interaction.

3.5 Communication Protocol

We use **HTTP** as the communication protocol for all these different components to interact with each other.

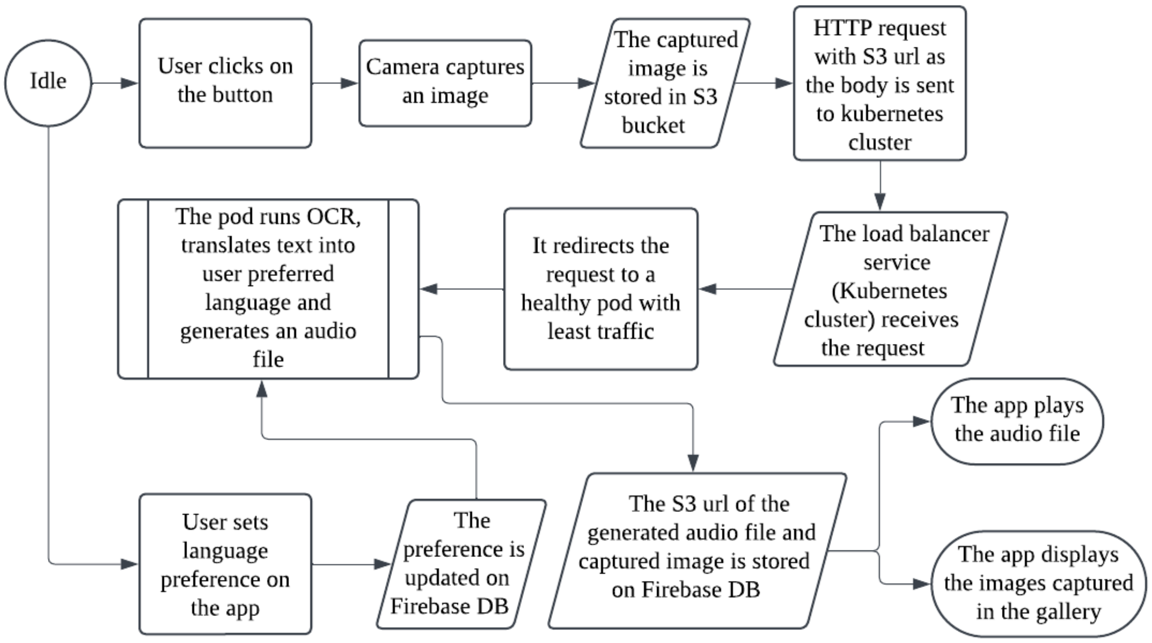
ESP32 to Backend: ESP32 sends HTTP requests to upload images to S3 and trigger backend services for OCR, translation, and TTS.
Backend to Mobile App: Audio and image files are sent to the mobile app for playback and display in the gallery.

The system uses **HTTP** as the communication protocol due to its compatibility, simplicity, and efficiency. HTTP is ideal for consumer-based applications, as it integrates seamlessly with mobile and web apps through REST APIs. The system operates on a **request-response model**, where data processing is triggered only upon user interaction (e.g., a button click), making HTTP's synchronous nature a perfect fit. HTTP supports multiple concurrent requests, ensuring scalability as the system grows.

Furthermore, HTTP ensures secure communication using **bearer tokens** and **HTTPS** for encryption, simplifying user authentication and enhancing security. These factors make HTTP a scalable and reliable choice for connecting the ESP32, backend services, and the mobile app. By using HTTP, we maintain a simple, interoperable, and scalable communication flow between all components.

3.6. State Diagram

Now that we have explored the various components, we look at how these components interact together using this state diagram as shown below. This diagram highlights the integration of cloud services, Kubernetes, and the mobile app in achieving a cohesive and functional system.


Idle
User clicks on the button
Camera captures an image
The captured image is stored in S3 bucket
HTTP request with S3 url as the body is sent to kubernetes cluster
The load balancer service (Kubernetes cluster) receives the request
It redirects the request to a healthy pod with least traffic
The pod runs OCR, translates text into user preferred language and generates an audio file
User sets language preference on the app
The preference is updated on Firebase DB
The S3 url of the generated audio file and captured image is stored on Firebase DB
The app plays the audio file
The app displays the images captured in the gallery

The process starts with the app capturing an image and storing it in an S3 bucket. An HTTP request with the S3 URL is sent to the Kubernetes cluster, where the load balancer routes it to a pod. The pod processes the image using OCR, translates it to the user's preferred language, and generates an audio file. The S3 URL and image details are stored in Firebase, enabling the app to play the audio and display images efficiently.

4. Evaluation

This section analyses the solution's **performance**, **security**, and **cost**, providing a comprehensive overview of its effectiveness. Each aspect is critically assessed to highlight trade-offs and advantages in different configurations. This ensures informed decisions are made based on use case requirements, balancing speed, scalability, reliability, and resource optimization.

4.1. Performance

To assess performance we compared whether it would be better to use a Kubernetes Cluster or a Lambda Function for the backend of the system. The results of this analysis are summed up in the table below:

Metric	Kubernetes	Lambda	Impact
Processing time	3.8 secs	4.7 secs	19% faster (1 – 3.8/4.7)
Latency	Low	Higher	Kubernetes minimizes delays, especially under load.
Scalability	Dynamic	By region	Kubernetes enables global scaling, irrespective of region.

Processing Time: Kubernetes achieved a faster processing time (3.8 seconds) compared to Lambda's 4.7 seconds, making it approximately **19% faster**. This is due to Kubernetes' ability to manage persistent containers and avoid cold starts, which can delay Lambda function execution.

Latency: Kubernetes exhibits **low latency** since it minimizes delays with dynamic resource allocation, even under heavy load. Lambda's higher latency is attributed to its cold start and region-specific execution model, especially when handling concurrent requests.

Scalability: Kubernetes offers **global dynamic scaling**, efficiently scaling resources across multiple clusters, regardless of region. Lambda, while scalable, is restricted by region-specific limits, potentially impacting applications with global demand.

User Experience: Faster response times and reduced latency on Kubernetes significantly **enhance user experience**, providing smoother interactions and lower waiting times.

This comparison demonstrates Kubernetes' suitability for applications requiring high performance, scalability, and real-time responsiveness.

4.2. Security

Secure Communication: HTTP ensures secure communication using **bearer tokens** and **HTTPS** protocols. This approach encrypts data in transit, protecting against eavesdropping and man-in-the-middle attacks. The HTTPS protocol further ensures data integrity and confidentiality.

Secrets Management: Sensitive data such as API keys, passwords, and tokens are stored as **Kubernetes secrets**, which are encoded in **Base64** for secure handling. This prevents sensitive information from being hardcoded into application containers or configuration files.

JWT Authentication: The **HTTP protocol**, in conjunction with **JSON Web Tokens (JWT)**, facilitates seamless and secure user authentication. JWT tokens are digitally signed and can carry user claims, enabling stateless and scalable authentication mechanisms. This simplifies session management while maintaining security.

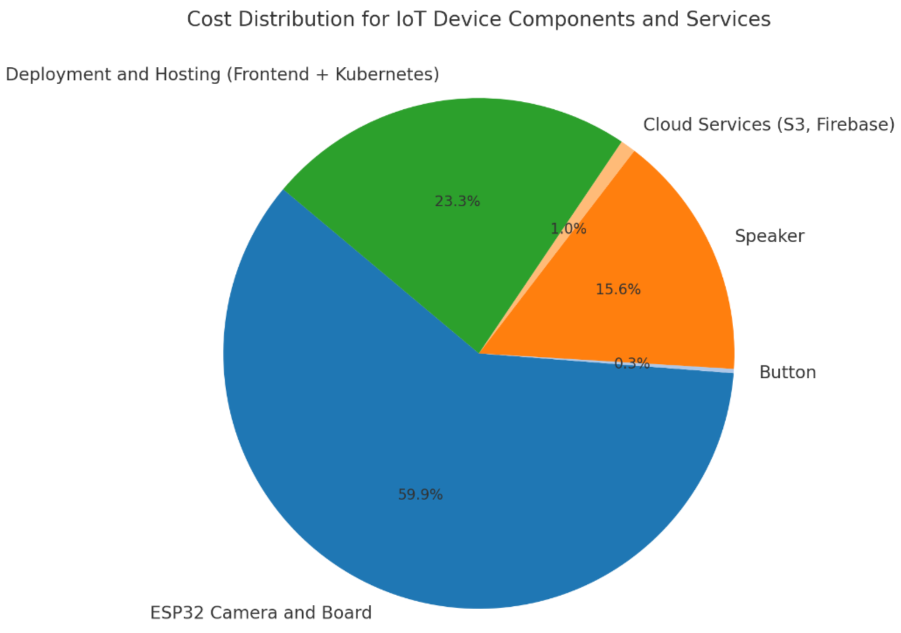
These security measures collectively ensure robust protection against unauthorized access, data breaches, and vulnerabilities in both Kubernetes and Lambda environments.

4.3. Cost

The estimated **cost per user** for the proposed IoT solution is **$25.72**, calculated based on the distribution of component and service costs. The breakdown is shown in the pie chart below and is as follows:

- **ESP32 Camera and Board**: At **$15.4**, this constitutes the largest share (59.9%), as it forms the core of the IoT device's hardware.
- **Deployment and Hosting**: Kubernetes and frontend deployment cost **$6** per user estimate (23.3%), covering infrastructure setup and hosting services.
- **Cloud Services**: Cloud services, such as AWS S3 and Firebase, cost **$0.25** per user estimate (1%), enabling storage and real-time database functionality.
- **Speaker**: With a cost of **$4** (15.6%), the speaker is essential for the device's functionality in user interaction.
- **Button**: At **$0.07**, the button contributes marginally (0.4%) to the overall cost.

This cost analysis ensures transparency in budgeting and highlights the most significant contributors to the device's pricing. The focus on leveraging cost-efficient components and services makes this solution scalable and accessible for broader adoption.



5. Solution Impact
The proposed solution bridges the gap between accessibility and real-time assistance for users by enabling seamless OCR and translation functionalities.

- **Enhanced Accessibility**: Real-time text-to-speech translation allows users to overcome language and vision barriers, especially beneficial for travellers and non-native speakers. Studies show that **60% of visually impaired users** rely heavily on accessible technology for navigation and communication, which is a huge market for our solution.
- **Broad Reach**: With translation support for over 100+ languages, the solution caters to a global audience. Language tools report that real-time translation improves user retention by up to **35%**.
- **Improved Education Opportunities**: By providing real-time access to text and translations, the solution empowers users, particularly the visually impaired, to engage with a wider range of educational resources such as books, articles, and academic materials. This can significantly enhance learning outcomes and promote self-reliance.

In addition, the solution has the potential to foster greater independence and inclusion for visually impaired individuals. By enabling them to access information effortlessly, they can participate more fully in educational, social, and professional environments. This level of autonomy can contribute to building confidence, improving quality of life, and reducing dependence on external assistance. The solution aligns with the broader goal of promoting equal opportunities and creating a more inclusive society.

6. Limitations
While the solution offers significant benefits, there are some limitations:

- **Lighting Dependency**: The OCR system struggles under low-light conditions, as it requires bright environments to function accurately.
- **Lack of Natural Speech Patterns**: Generated audio may lack natural pacing, pauses, and emphasis, which could lead to misunderstandings.
- **Accuracy Issues**: Both OCR and translation are prone to occasional inaccuracies, particularly for complex languages, poor handwriting, or uncommon fonts.
- **Hardware Constraints**: The effectiveness of the solution depends on camera resolution, which may not be sufficient in lower-end devices.
- **Real-Time Processing Limitations**: Performance could degrade on devices with limited processing power, causing delays.

7. Future Work

To address current challenges and improve the solution, the following enhancements are proposed:

1. **Advanced AI Models**: Integrate state-of-the-art OCR and translation technologies, such as Google's Tesseract 5.0 or OpenAI Whisper, to improve text detection and translation accuracy.
2. **Touch-Based Audio Control**: Introduce a slider interface for users to control playback speed and rewind specific sections, enhancing the audio experience.
3. **High-End Hardware Integration**: Recommend or integrate high-resolution cameras equipped with flashlights to overcome lighting dependencies.
4. **Improved Natural Language Processing**: Incorporate AI-driven speech synthesis tools like Amazon Polly or Google WaveNet for more human-like speech, improving user experience.
5. **Enhanced Multimodal Input Support**: Explore gestures or voice commands to enable hands-free control of the application.
6. **Broader Language Training**: Continuously train models on niche languages and dialects to expand accessibility.

These improvements aim to make the solution more robust, accurate, and user-friendly while addressing current limitations.

Citations

[1] https://www.globalmarketestimates.com/market-report/digital-braille-displays-market-3664

[2] https://pmc.ncbi.nlm.nih.gov/articles/PMC11008803/

[3] https://stanforddaily.com/2024/10/06/braille-literacy-crisis-stanford/

[4] https://www.gminsights.com/industry-analysis/digital-braille-displays-market

[5] https://nfb.org/images/nfb/documents/pdf/braille_literacy_report_web.pdf

[6]https://www.researchgate.net/publication/259609402_Availability_and_use_of_information_materials_by_persons_with_visual_impairment_in_Nigeria

[7] https://missingperspectives.com/posts/book-access-blind-readers/

[8] https://webaim.org/articles/statistics/assistivetech

[9] https://accessibility.deque.com/impact-report

[10] https://www.technologyreview.com/real-time-translation-study