Quick Recap: Your Meetings, Summarized in Seconds

Naveen Nayak, Navya Nayak, Madhu Kiran
Saranya Chandu, Brunai, Aakruthi Reddy, Vishwas Mamidi
Seidenberg School of Computer Science and Information Systems
Pace University, New York, NY, USA

Abstract - Quick Recap is a web-based tool designed to streamline the process of consuming long audio and video content by generating concise, personalized summaries. Traditional methods like manual notetaking or reviewing recordings can be time-consuming and ineffective. Quick Recap addresses this issue by utilizing advanced technologies such as automatic speech recognition (ASR) with tools like Google Speech-to-Text API, alongside natural language processing (NLP) to create context-aware summaries tailored to the user's specific role and preferences. Users can upload various recordings, and the tool transcribes and summarizes the content in a way that is relevant. Powered by modern technologies like React, Firebase, and cloud-based services, the platform ensures secure media handling, scalable storage, and real-time performance. This paper explores the design, architecture, and implementation of Quick Recap, emphasizing its potential to enhance productivity, improve accessibility, and transform the way users engage with recorded content, making it more efficient and effective for professionals, students, and content consumers alike.

Keywords - Audio Summarization, Speech-to-Text, Natural Language Processing, Machine Learning, Personalized Content, Google Speech-to-Text API.

I. INTRODUCTION

In today's fast-moving digital environment, audio and video content have become essential tools for communication, education, and collaboration. From online meetings and virtual classrooms to podcasts and webinars, the volume of recorded material continues to grow at an incredible rate. Yet, reviewing this content in its entirety presents a significant challenge due to time limitations, the difficulty of maintaining focus, and the lack of personalized filtering to cater to different user needs. Traditional methods, like manually taking notes or rewatching long sessions, are often ineffective and time-consuming.

Quick Recap was created to address these issues by offering an intuitive platform that generates concise, tailored summaries of lengthy recordings. Users can upload audio or video files, and the system produces summaries relevant to their specific role or needs. This is accomplished through the integration of Automatic Speech Recognition (ASR) technologies, such as Google Speech-to-Text, combined with Natural Language Processing (NLP) and Machine Learning (ML) models to personalize the output based on individual preferences.

Quick Recap is set apart by its ability to identify and condense key information, helping users manage content more efficiently whether they're professionals attending multiple meetings, students revisiting lectures, or journalists conducting interviews. The platform boosts productivity and accessibility by delivering focused, relevant summaries.

Built on a modern tech stack with React for the frontend, Firebase for backend services, and cloud functions for tasks like transcription and summarization, the system is scalable, fast, and responsive. This paper explores the system's design, technology architecture, and implementation process, while also addressing challenges such as transcription accuracy, summary generation, and the importance of user personalization.

II. LITERATURE REVIEW

The increasing reliance on multimedia content in educational, corporate, and journalistic settings has led to the need for efficient tools to manage and extract value from large volumes of audio and video recordings. In recent years, the fields of automatic

speech recognition (ASR), natural language processing (NLP), and text summarization have made significant strides in enabling intelligent media analysis and summarization.

A. Automatic Speech Recognition (ASR)

ASR technology serves as the foundation for converting spoken language into text. Early systems such as IBM's ViaVoice and Dragon Naturally Speaking relied heavily on rule-based models with limited vocabulary and accuracy. Modern ASR systems like Google Speech-to-Text API, Microsoft Azure Speech Services, and OpenAI's Whisper use deep learning, particularly recurrent neural networks (RNNs) and transformer-based architectures, to significantly enhance transcription accuracy and real-time performance. Research [1] has introduced Deep Speech, an end-to-end ASR system that laid the groundwork for current speech models. Whisper, introduced by OpenAI in 2022, offers open-source multilingual transcription with robust handling of accents and background noise, making it an ideal choice for diverse recording environments.

B. Natural Language Processing (NLP) and Text Summarization

Text summarization, a subset of NLP, aims to reduce text length while preserving key information. It is categorized into extractive and abstractive methods. Extractive summarization identifies and compiles the most relevant sentences from the source, while abstractive summarization involves generating novel sentences that paraphrase the content. Algorithms like TextRank [2], based on Google's PageRank, are commonly used in extractive methods. On the abstractive side, transformer models such as BART [3], T5, and PEGASUS [4] have shown remarkable performance in generating human-like summaries. Research work on PEGASUS demonstrated the model's ability to produce summaries that closely mirror human comprehension across multiple domains.

C. Personalized Summarization

Personalization in summarization is an emerging area that aligns content output with individual preferences or roles. Research in this domain often involves combining user profiles or contextual metadata with semantic analysis. Studies have proposed frameworks for tailoring summaries based on user intent and task relevance. Integrating ML techniques such as collaborative filtering, topic modeling, and context-aware ranking, these approaches pave the way for building dynamic, user-specific summarization systems like Quick Recap.

D. Similar Systems and Gaps

Several commercial tools offer transcription or basic summarization services (e.g., Otter.ai [5], Fireflies.ai, and Sonix), yet they lack robust personalization or full-stack integration for audio/video summarization in one workflow. Additionally, many of these platforms are paid and offer limited customization, creating an opportunity for open, student-built solutions that balance utility, affordability, and user-friendliness.

E. Motivation for Quick Recap

Given the limitations in current solutions such as generic summaries, lack of contextual understanding, or expensive subscription models, Quick Recap seeks to bridge these gaps by combining open-source and cloud-based technologies to deliver a fully integrated, role-based summarization system. It is specifically tailored for diverse user groups such as students, journalists, and working professionals, offering a flexible and intelligent way to digest information quickly.

III. SYSTEM DESIGN AND IMPLEMENTATION

The Quick Recap system is engineered to be user-centric, efficient, and modular. It enables users to effortlessly upload audio or video files in formats such as MP3, MP4, and WAV, which are then transcribed and summarized through a series of intelligent processing steps.

A. Audio Transcription

Once an audio or video file is uploaded via the web interface, it is first processed by Google Speech-to-Text API. This API converts spoken language into accurate text by leveraging state-of-the-art deep learning models. It supports a wide range of languages and dialects, ensuring accessibility for global users. The transcription process is fast,

scalable, and capable of handling long-form audio files.

B. Text Summarization

The resulting transcript is then passed to the Text Summarization module, where the TextRank algorithm is applied. TextRank is an unsupervised, graph-based ranking algorithm like PageRank. It identifies and extracts the most relevant sentences from the transcript, producing a coherent and concise summary. This approach preserves the semantic integrity of the original content while significantly reducing its length.

C. Personalization Engine

To enhance user relevance, a personalization layer tailors the summary based on individual roles or preferences. These preferences are stored and managed via Firebase Firestore, which allows dynamic retrieval and application of user-specific data. By associating summaries with contextual metadata such as profession, interest areas, or preferred detail level the system delivers more meaningful and role-aware insights.

D. System Architecture

The architecture of Quick Recap consists of three major components that communicate seamlessly to ensure end-to-end functionality. Figure 1 presents a high-level overview of the system design.

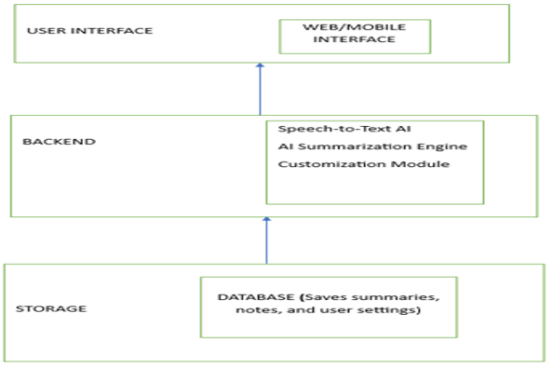

Fig. 1. System Architecture of Quick Recap

The architecture includes three main components:

A. User Interface (Frontend)

Built using ReactJS, the UI allows users to upload files, manage their profile, view transcripts and summaries, and configure personalization settings.

B. Backend Services

Powered by Firebase Functions and RESTful APIs. Handles the integration with Google Speech-to-Text, the TextRank Summarizer, and the Personalization Engine.

C. Data Storage

Firebase Firestore is used to store user metadata, upload file references, transcription results, and personalized summaries. Firebase Storage handles the secure storage of uploaded media files.

IV. DEVELOPMENT METHODOLOGY

Quick Recap is a web-based application designed to simplify and personalize the consumption of long-form audio and video content through automated transcription and intelligent summarization. The system integrates modern web development frameworks, cloud infrastructure, and machine learning technologies to deliver an accessible, performant, and user centric platform. This section outlines the methodologies used across the core modules of the system, reflecting completed user stories and architectural decisions.

A. User Interface (Frontend)

The frontend of Quick Recap is developed using React.js and Vite, optimized for fast rendering and developer-friendly configuration. Initial design mockups were created in Figma, with multiple theme options such as light, dark, and playful modes to ensure accessibility and modern aesthetics. Tailwind CSS is used for responsive design, enabling consistent styling across screen sizes and devices.

The user interface allows authenticated users to:

- Register and log in securely
- Upload audio or video files (MP3, MP4, WAV)
- View transcribed text
- Generate and receive personalized summaries
- Toggle between raw transcript and summary views

Development is conducted using VSCode, with version control managed on GitHub. The modular React architecture ensures future scalability and easy integration of new features such as video summarization and keyword highlighting.

B. Backend and API Integration

The backend logic is managed through Firebase Functions and third-party APIs. Upon media upload, the backend initiates a transcription request to the Google Speech-to-Text API, which converts the audio content into accurate, timestamped text. The backend also houses logic for:

- Verifying uploaded files and checking formats
- Triggering cloud-based summarization
- Interfacing with personalization models using user role data

RESTful APIs are exposed to the frontend to abstract complexity and ensure a clean separation of concerns. Error handling is integrated at each step, ensuring users receive informative feedback during failed uploads or processing errors.

C. Cloud Infrastructure

The project leverages the Firebase ecosystem for backend services, ensuring reliability, scalability, and low-cost operation ideal for student and prototype environments.

- Firebase Authentication manages secure login and role-based access control.
- Firebase Firestore stores user preferences and summary history in structured, real-time collections.
- Firebase Storage handles uploaded media files, supporting large audio/video formats with temporary URLs for security.
- Firebase Functions are used for invoking asynchronous background tasks like initiating transcription, storing results, and applying personalization logic.

This cloud-native setup ensures the system can scale and update in real time without the overhead of managing traditional server infrastructure.

D. Summarization and Personalization Engine

The summarization engine is powered by TextRank, a graph based natural language processing (NLP) algorithm that extracts the most important sentences from the transcript. This unsupervised technique ensures that summaries remain faithful to the source while reducing verbosity.

For personalization, the system tailors summaries based on user-selected roles (e.g., Student, Journalist, Project Manager). This is done by filtering or prioritizing content types (e.g., deadlines for managers, insights for students) using metadata and keyword-based logic. User profiles are stored in Firestore and influence the output returned by the summarization engine.

Future improvements include integrating transformer-based summarization models for higher contextual accuracy and support for multilingual content.

E. Workflow

The user workflow in Quick Recap follows a structured and asynchronous model:

- Upload: Users upload audio/video files through the interface.
- Storage: The file is saved securely in Firebase Storage.
- Transcription: A cloud function is triggered by sending the file to the Google Speech-to-Text API.
- Summarization: The resulting transcript is processed by the TextRank algorithm.
- Personalization: The summary is adapted based on user role preferences.
- Presentation: The user is notified and presented with both the transcript and summary via the dashboard.

F. Deployment and Continuous Integration

The application is hosted via Firebase Hosting, ensuring fast global delivery and SSL encryption. GitHub Actions manage CI/CD pipelines, automating tasks like:

- Testing new code on pull requests
- Deploying updated builds to Firebase

- Running test suites to validate frontend and backend integration

This methodology ensures that the system remains modular, reliable, and rapidly deployable as new features are added.

V. RESULTS AND DISCUSSION

The Quick Recap system was evaluated based on its core functionalities: audio transcription, summarization accuracy, and personalized content delivery. The goal was to validate whether the tool could effectively reduce the time and cognitive load required to extract meaningful information from long-form audio content.

A. Transcription Performance

The Google Speech-to-Text API was used to transcribe uploaded audio and video files. Tests were conducted using a variety of formats (MP3, MP4, WAV) and across diverse audio conditions including:

- Varying speaker accents
- Background noise
- Different speaking speeds

Results indicated a high transcription accuracy, especially for clear recordings. For recordings with minor background noise or varied accents, the API still maintained strong performance, showcasing its robustness and multilingual support.

B. Summarization Accuracy

The TextRank algorithm was applied to the transcribed text to generate extractive summaries. The system was able to capture key ideas and reduce content length by up to 60–70% without losing essential meaning. We noted that the summaries preserved the logical flow and context of the original discussions.

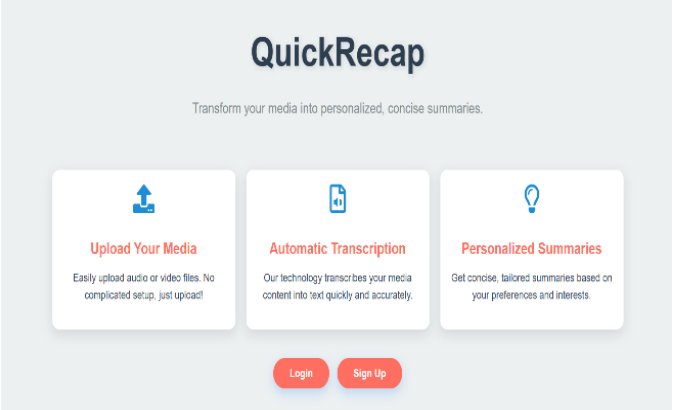

Fig. 2. Home page

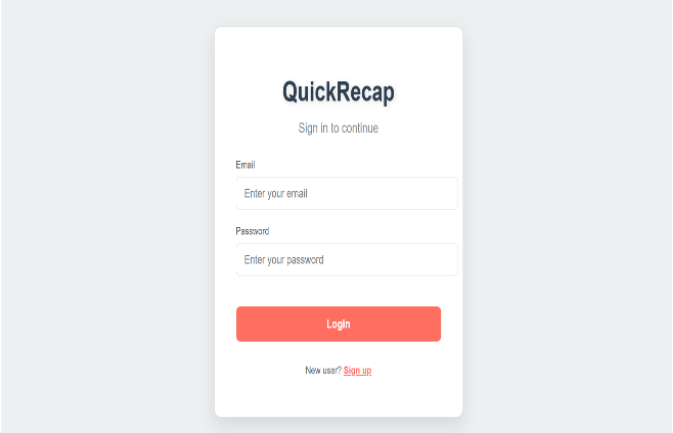

Fig. 3. User Sign-up page

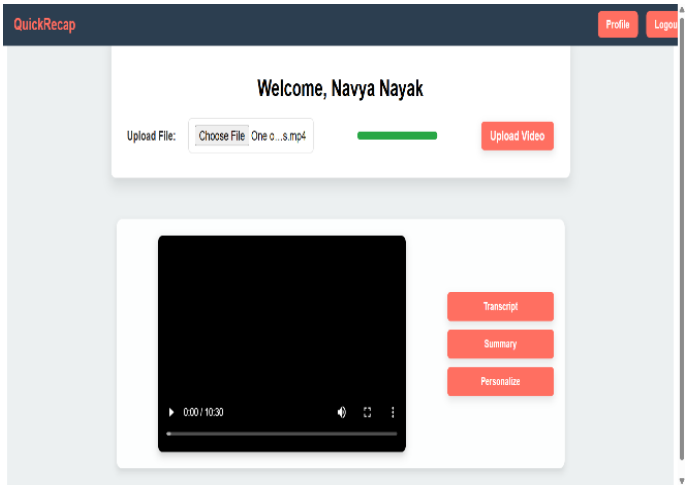

Fig. 4. Dashboard page

VI. CONCLUSIONS

Quick Recap demonstrates a practical application of modern AI and machine learning techniques to improve information accessibility through personalized audio summarization. The use of the Google Speech-to-Text API, combined with the Text Rank algorithm, enables effective and efficient summarization. Future improvements could include expanding the personalization features to incorporate more detailed user preferences or integrating with additional platforms.

VII. REFERENCES

[1] Amodei, D., Ananthanarayanan, S., Anubhai, R., et al. (2016). "Deep Speech 2: End-to-End Speech Recognition in English and Mandarin". ICML.

[2] Mihalcea, R., & Tarau, P. (2004). "TextRank: Bringing order into texts". Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing.

[3] Lewis, M., Liu, Y., Goyal, N., et al. (2020). "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension". ACL.

[4] Zhang, J., Zhao, Y., Saleh, M., & Liu, P. J. (2020). "PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization". ICML.

[5] Otter.ai. Otter Voice Notes. https://otter.ai