Voice-Controlled TaskManager

Name: Divyanshu Pushpad
UID: 24BCS10223
Section: 24KGR_711-B
Subject: Full Stack Development-I
Subject Code: 24CSP-210
Faculty: Er. Juned Alam

Introduction:

In the modern digital landscape, voice-command technology is revolutionizing how users interact with productivity tools. Traditional task management requires manual data entry, which can be slow and prone to user fatigue during busy schedules.

To address these challenges, the Voice-Controlled Task Manager project is designed as a modern, web-based application that allows users to manage their to-do lists hands-free. The system enables users to create, update, and track tasks using voice commands, which are processed in real-time to improve efficiency. Developed using Spring Boot for backend services, React for the frontend interface, and MySQL for data storage, the project emphasizes accessibility and seamless user interaction.

Objectives of the Project:

The primary objectives of the Voice-Controlled Task Manager are:

- To automate the process of task creation and management using voice input.
- To reduce the need for manual typing and physical interaction.
- To provide a hands-free and accessible platform for all users.
- To implement a high-accuracy speech-to-text mapping engine for task parameters.
- To enable users to efficiently manage their daily schedules via natural language.
- To store and retrieve task-related data securely using a relational database.
- To ensure system scalability through a modular full-stack architecture.

Scope of the Project:

The scope of this project includes the design and development of a web-based task management platform with voice recognition capabilities. The system focuses on personal task management but can be extended to include collaborative team tasks, shared project boards, and cross-platform mobile support.

Technology Stack:

The Voice-Controlled Task Manager is built using the following technologies:

Frontend :

- React.js: For component-based UI architecture.
- Web Speech API: For real-time voice-to-text conversion and command processing.
- Responsive UI: Ensuring the dashboard works across various devices.

Backend :

- Spring Boot: For creating robust RESTful web services.
- Business Logic: To parse voice transcripts into actionable task data.
- Security: To handle user authentication and data protection.

Database :

- MySQL: For structured and persistent data storage.
- JPA/Hibernate: For efficient object-relational mapping.

System Architecture:

The system follows a three-tier architecture:

- Presentation Layer (Frontend): Built with React; captures audio via the Web Speech API and handles user visualization.
- Application Layer (Backend): Built with Spring Boot; contains the logic for interpreting voice commands and managing task states.
- Data Layer (Database): MySQL database storing user profiles, task details, and reminder settings.

High-Level Design (HLD):

The system architecture ensures clear separation of concerns:

- Presentation Layer: Handles user interaction and microphone input.
- Application Layer: Processes transcribed text and validates commands.
- Data Layer: Stores tasks, status updates, and user credentials.

Data Flow Overview:

1. User speaks a command into the interface.
2. Frontend converts speech to text and sends it to the backend.
3. Backend parses the intent (e.g., "Add", "Delete") and updates the database.
4. The updated task list is returned and displayed on the dashboard.

Low-Level Design (LLD):

Backend Components

- Controller Layer: Handles HTTP requests for task operations.
- Service Layer: Contains logic for parsing dates and priorities from voice transcripts.
- Repository Layer: Manages database CRUD operations via JPA.
- Security Layer: Ensures only authorized users can access their specific tasks.

Database Design

Entities:

- **User:** Stores login credentials and preferences.
- **Task:** Stores descriptions, deadlines, and completion status.
- **Category:** Groups tasks by type (Work, Personal, etc.).
- **VoiceLog:** Stores transcription history for accuracy monitoring.

Functional Modules:

Voice Input Module

This module allows users to interact with the system using spoken language.

- Key Features: Microphone toggle, real-time transcription, and visual signal processing.
- Inputs: Audio signal from the user.
- Outputs: Transcribed text forwarded for command mapping.

Task CRUD Module

Manages the fundamental lifecycle of to-do items.

- Key Features: Automated creation, reading, updating, and deletion of tasks.
- Benefits: Instant synchronization between voice input and the visual dashboard.

Reminder Module

Notifies users of upcoming deadlines.

- Key Features: Scheduled alerts and status tracking for time-sensitive tasks.
- Actions: Triggering notifications when a task's scheduled time is reached.

Module Breakdown (8 Modules):

The project is divided into 8 independent modules for modularity:

Frontend Modules (3 Modules)

- Module 1: User Auth Module (Frontend): Handles signup, login, and secure session management.
- Module 2: Voice Recognition Module (Frontend): Integrates Web Speech API to capture and convert voice to text.

- Module 3: Task Dashboard Module (Frontend): Provides a visual interface to view and interact with the task list.

Backend Modules (5 Modules)

- Module 4: User Account Module (Backend): Manages profile data and role-based access.
- Module 5: Task Management Module (Backend): Handles the logic for creating and retrieving tasks.
- Module 6: Command Parser Module (Backend): Uses logic to identify actions (Add/Delete) from text transcripts.
- Module 7: Notification Service (Backend): Manages the scheduling and delivery of task reminders.
- Module 8: Security & API Module (Backend): Protects endpoints using JWT and filters incoming requests.

Use Case Description:

Primary Use Case: Voice Task Entry

1. User logs into the Task Manager.
2. User clicks the microphone and says, "Add task: Meeting at 5 PM".
3. System converts speech to text and evaluates the command.
4. The task is saved to the database.
5. The dashboard updates to show the new "Meeting" task.
6. User is notified of the successful addition.

Advantages of the System:

- Increased Efficiency: Faster task entry compared to manual typing.
- Accessibility: Enables users with physical disabilities to manage tasks easily.
- Hands-Free Operation: Allows users to organize while engaged in other activities.
- Scalable Architecture: Modular design supports adding new command types easily.
- Centralized Storage: Securely keeps all tasks synced across sessions.

Limitations:

- Environmental Noise: Background noise can reduce speech recognition accuracy.
- Internet Dependency: Web Speech API often requires an active connection.
- Accent Sensitivity: The system may struggle with diverse regional accents.

Future Enhancements:

- AI/ML Integration: For better intent understanding and natural language processing.
- Multi-Language Support: Expanding beyond English voice commands.
- Mobile App: Developing native iOS/Android apps for on-the-go use.
- Smart Home Sync: Integration with Google Assistant or Alexa.

Conclusion:

The Voice-Controlled Task Manager is a robust and efficient solution that modernizes personal organization. By utilizing Spring Boot, React, and the Web Speech API, the system ensures high performance and a user-centric experience. This project demonstrates the practical application of full-stack development and voice-processing technologies in real-world productivity tools.