**Chapter 3: Design of the Proposed System**

* 1. **Introduction**

This chapter describes the analysis and design of PDF to Audio Converter system and the type of the methodology used in the system development. The requirements of the system are identified in this chapter, which includes functional and non-functional requirements. The tools that are used when developing the system are identified, which consists of hardware and software tools as well as the technologies used.

In addition, this chapter describes the model of the system which is simplified, complete, and consistent abstraction of the system, that is created for better understanding of the system.

The PDF to Audio Converter system is designed as a web-based solution that allows users to upload PDF files and convert their textual content into audio using speech synthesis. The application uses a combination server-side technology and client-side technologies to accomplish this. This chapter discusses the system architecture, technologies used, system components, data flow, and the user interface design.

The primary goals of the system are to:

* Provide an intuitive and easy-to-use interface for users to upload and convert PDF files.
* Efficiently extract text from PDF documents.
* Convert the extracted text into speech using the Web Speech API.
  1. **Proposed Methodology**

The proposed methodology for developing the PDF to Audio Converter Application follows a structured approach based on the Software Development Life Cycle (SDLC), specifically the Waterfall Model. This methodology ensures that each phase is completed before moving on to the next, providing a clear and organized path from requirements gathering to deployment.

This methodology ensures that the application is built efficiently while maintaining a clear focus on meeting user requirements.

* 1. **System Design**

System design is the process of defining the architecture, components, interfaces, and data for the system to satisfy the specified functional requirements. This section provides the system architecture, the activity diagram and data flow.

The architecture of the PDF to Audio system follows a client-server model, where the user interacts with the web interface (client), and the server performs text extraction from the PDF. Once the text is extracted, it is sent back to the client, where speech synthesis is performed using the browser's Web Speech API.

* + 1. **Overview of System Components**
* **Frontend (Client-side)**:
  + The frontend provides the user interface where users can upload PDF documents, control speech synthesis (play, pause, resume), and adjust speech settings like rate and pitch.
  + It also handles the conversion of extracted text into speech using the Web Speech API.
* **Backend (Server-side)**:
  + The backend is responsible for processing the uploaded PDF files, extracting the text, and sending the extracted text back to the frontend.
  + PHP is used as the server-side scripting language, utilizing the PDF2Text library for text extraction.
    1. **Use Case Diagram**

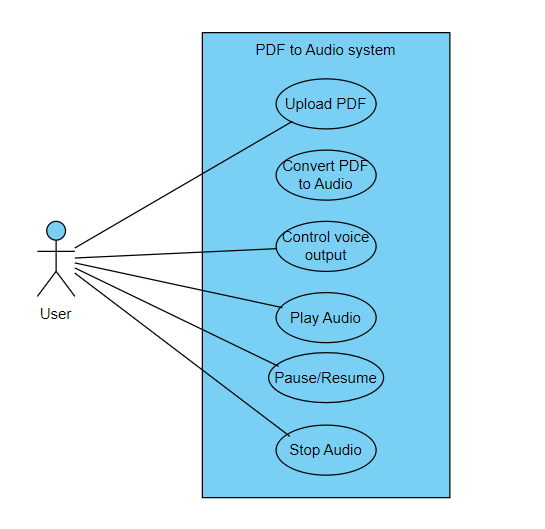


Figure 3.1 Use Case Diagram

The models the interaction between a user and the system. In the diagram from the left is a user. Within the rectangle which represents the system are what are called use cases. The use cases are essentially the actions an actor (user) can perform on the system.

* + 1. **Data Flow Diagram**

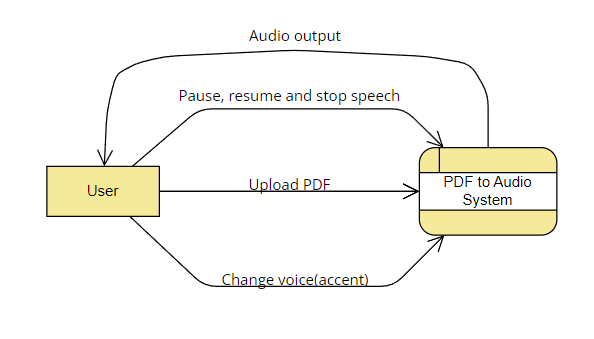
****

Figure 3.2 Data Flow Diagram

1. The client uploads the PDF to the server.
2. The server processes the PDF, extracts text using PDF2Text, and sends the extracted text back to the client.
3. The client renders the text and uses the Web Speech API to convert it into speech. Users can pause, resume, or adjust the speech parameters.
   1. **System Requirement**

The system is anticipated to operate efficiently on a server with a minimum RAM capacity of 4GB and a CPU speed of 1.0 GHz or higher. An initial setup will need a disc space of 1GB. The system is designed to be compatible with and support the system files of any of the following operating systems.

• Windows 7, 8, 10, and 11

• Linux

* + 1. **Hardware Requirements**

In computer terminology, the term hardware configuration is used to describe the kind of components assembled together to make a complete (whole) computer physically. The system configuration required for the proposed system is as follows:

1. Minimum of Intel Dual core processor.
2. Minimum of 2.0GHz of Processor speed
3. Minimum of 4GB of RAM (Random access memory).
4. Minimum of 250GB HDD (HARD Disk Drive).
   * 1. **Software Requirements**

The following are the system software requirements;

1. Operation System (OS): Windows 7/8/8.1/10.
2. Web Browser: Latest version of Google Chrome, Mozilla Firefox, Microsoft Edge.
3. XAMPP Local host Server (Support PHP & MySQL).
   * 1. **Choice Of Programming Language**

**PHP**: PHP allows for easy integration of dynamic functionality into web pages. Assigning the .php suffix to pages gives them immediate and direct access to the scripting language. PHP was used in this project to author the programs that provide instructions to the software, so enabling it to function as a dynamic website. PHP is a robust tool for creating web pages that are dynamic and interactive. It is extensively used, free, and efficient compared to rivals like Microsoft's ASP. PHP is a programming language that is embedded in HTML. It incorporates syntax from C, Java, and Perl, along with some distinctive characteristics particular to PHP. The objective of the language is to enable web developers to efficiently create dynamically generated pages. PHP is used for establishing the database connection and executing queries in this project.

**MySQL** is a relational database management system. Data is stored in distinct tables instead of being consolidated in a single repository. This enhances speed and flexibility. MySQL database server exhibits exceptional speed, reliability, affordability, and user-friendliness, making it highly accessible and easily comprehensible. Currently, MySQL provides a diverse and valuable range of functionalities. Its exceptional connection, rapid speed, and robust security make it very well-suited for accessing databases on the internet. Additionally, it offers full support for clustering technologies. MySQL is used for database creation in this project.

**JavaScript**: JavaScript was used to provide further functionalities to the webpage. The website offers features such as image slides and enhanced security measures.

**Web Speech API**: The Web Speech API is used to handle speech synthesis on the client side. It allows real-time conversion of text into speech and offers options to adjust speech rate and pitch.

**HTML/CSS (Bootstrap)**: Used for building the user interface, ensuring it is responsive and user-friendly.

* 1. **User Interface Design**

The user interface is designed to be simple and intuitive. Below are the key elements of the interface:

* **File Upload Section**:
  + A form for users to upload their PDF files. It includes a file input element and a submit button.
* **Text Display**:
  + The extracted text is displayed in a text area where users can view the content that will be converted to audio. This helps users verify the text before playing it.
* **Playback Controls**:
  + The application provides buttons to control the audio playback:
    - **Play**: Starts reading the text aloud.
    - **Pause**: Pauses the audio playback.
    - **Resume**: Resumes the audio from where it was paused.
* **Speech Controls**:
  + Sliders allow users to adjust the speech rate (speed) and pitch of the voice.
* **Error Messages**:
  + If an invalid file is uploaded or if text extraction fails, the user is notified with appropriate error messages.
  1. **Process analysis**

Text-to-speech synthesis takes place in several steps. The TTS systems get a text as input, which it first must analyze and then transform into a phonetic description. Then in a further step, it generates the prosody. From the information now available, it can produce a speech signal.

The structure of the text-to-speech synthesizer can be broken down into major modules:

* **Natural Language Processing (NLP) module:** It produces a phonetic transcription of the text read, together with prosody.
* **Digital Signal Processing (DSP) module:** It transforms the symbolic information it receives from NLP into audible and intelligible speech.
* The major operations of the NLP module are as follows:
* **Text Analysis:** First the text is segmented into tokens. The token-to-word conversion creates the orthographic form of the token. For the token “Mr” the orthographic form “Mister” is formed by expansion, the token “12” gets the orthographic form “twelve” and “1997” is transformed to “nineteen ninety-seven”.
* **Application of Pronunciation Rules:** After the text analysis has been completed, pronunciation rules can be applied. Letters cannot be transformed 1:1 into phonemes because correspondence is not always parallel. In certain environments, a single letter can correspond to either no phoneme (for example, “h” in “caught”) or several phoneme (“m” in “Maximum”). In addition, several letters can correspond to a single phoneme (“ch” in “rich”).
  1. **Design Justifications**

The design of the PDF to Audio Converter System is centered around accessibility and ease of use. Below are some key justifications for the chosen design:

* **Simple User Interface**: The UI is designed to be user-friendly, allowing individuals with limited technical skills to upload PDFs and convert them into speech without difficulty.
* **Cross-platform Compatibility**: By using web technologies such as HTML, JavaScript, and PHP, the application can run on any modern browser, making it accessible across different devices and platforms.
* **Customization Options**: The ability to adjust speech rate and pitch enhances user experience, catering to different preferences and needs.
* **Text-to-speech Integration**: By utilizing the Web Speech API, the application benefits from the native capabilities of modern browsers, which ensures high-quality and real-time speech synthesis.
  1. **Limitations of the Design**

While the system is designed to meet the objectives of converting PDFs to audio, there are some limitations:

1. **Handling Complex PDFs**: The text extraction module may struggle with complex PDFs that contain images, tables, or non-textual content.
2. **Browser Dependency**: The Web Speech API is not fully supported on all browsers, and some users may experience limitations depending on the browser they are using.
3. **Multilingual Support**: The current design focuses on English-language text extraction and speech synthesis. Future versions could explore adding support for multiple languages.
   1. **Summary**

This chapter has outlined the design of the PDF to Audio Converter Application. The system architecture is built around a client-server model where text extraction is handled by the server, and speech synthesis is performed on the client side. The frontend provides a simple, user-friendly interface, and the backend ensures that PDFs are processed efficiently. By leveraging technologies like PHP, Web Speech API, and MySQL, the system aims to provide a flexible, accessible solution for converting PDFs to audio.