

THEME TO ACCENT TRANSFORMATION APPLICATION

*project report submitted in partial fulfillment of
the requirements for the award of the degree of
MASTER OF COMPUTER APPLICATIONS*

Awarded By

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY KAKINADA,
KAKINADA.**

Submitted by

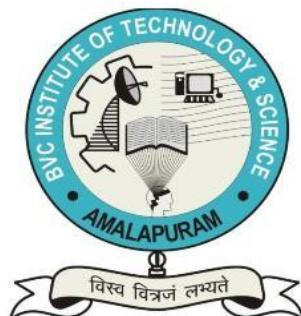
**Ms. TEJASWI MATTAPARTHI
(21H41F0042)**

Under the esteemed guidance of

Mrs. Y.S.N.CHANDESWARI, MCA,

Assistant Professor,

Department of Computer Applications



DEPARTMENT OF COMPUTER APPLICATIONS

BONAM VENKATA CHALAMAYYA INSTITUTE OF TECHNOLOGY & SCIENCE(A)

(Approved by A.I.C.T.E, New Delhi, Accredited by NAAC & Permanently Affiliated to J.N.T.U.K, Kakinada)

BATLAPALEM, AMALAPURAM-533221

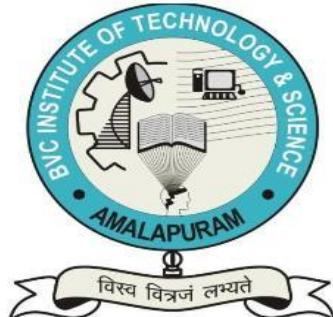
2023

DEPARTMENT OF COMPUTER APPLICATIONS

BONAM VENKATA CHALAMAYYA INSTITUTE OF TECHNOLOGY & SCIENCE(A)

(Approved by A.I.C.T.E, New Delhi, Accredited by NAAC & Permanently Affiliated to J.N.T.U.K, Kakinada)

BATLAPALEM,AMALAPURAM-533221



CERTIFICATE

This is to certify that the project work on "**THEME TO ACCENT TRANSFORMATION APPLICATION**" submitted by Ms. TEJASWI MATTAPARTHI(21H41F0042) is examined and adjudged as sufficient as a partial requirement for the **Master of Computer Applications** at Jawaharlal Nehru Technological University Kakinada, Kakinada is a bonafide record of the work done by her under my guidance and supervision.

Internal Guide

Mrs. Y.S.N.CHANDESWARI, MCA,

Assistant Professor,

Department of Computer Applications.

Head of The Department

Mr. A.V.S.M.GANESH, MCA,M.Tech,MISTE,

Associate Professor & Head,

Department of Computer Applications.

PROJECT EXTERNAL EXAMINER

DISSERTATION CERTIFICATE

This is to certify that the dissertation entitled "**THEME TO ACCENT TRANSFORMATION APPLICATION**" by **Ms. TEJASWI MATTAPARTHI (21H41F0042)** student of **Master of Computer Applications** of **Bonam Venkata Chalamayya Institute of Technology & Science(A), Amalapuram**, permanently affiliated to **Jawaharlal Nehru Technological University Kakinada, Kakinada** is hereby accepted and approved as a credible work. Her work has been found satisfactory for the partial fulfillment of the award of the degree of **MCA**.

INTERNAL EXAMINER

EXTERNAL EXAMINER

Head of the Department
Department of Computer Applications
Bonam Venkata Chalamayya Institute of Technology & Science(A)
Batlapalem, Amalapuram - 533 221

DECLARATION BY THE CANDIDATE

I, **Ms. TEJASWI MATTAPARTHI**, hereby declare that the project work entitled "**THEME TO ACCENT TRANSFORMATION APPLICATION**" is an authenticated work carried out by me at Bonam Venkata Chalamayya Institute of Technology and Science(A) Amalapuram, under the guidance of **Mrs. Y.S.N.CHANDESWARI, MCA**, for the partial fulfillment of the award of the degree of **Master of Computer Applications** and this work has not been submitted for similar purpose anywhere else except to **Bonam Venkata Chalamayya Institute of Technology &Science, Amalapuram** permanently affiliated to **Jawaharlal Nehru Technological University Kakinada, Kakinada.**

(Signature)
Tejaswi Mattaparthi
Regd.No: 21H41F0042

Date:

Place:

ACKNOWLEDGEMENT

I would like to express my heartiest concern of words to all those people who have helped me in various ways to complete my project.

I was highly indebted to **Mrs. Y.S.N.CHANDESWARI, MCA** my internal guide. She has been a constant source of encouragement and has inspired me in completing the project and helped us at various stages of project work

My sincere thanks to respectable **Mr. A.V.S.M. GANESH, MCA, M.Tech, MISTE, Associate Professor** and Head of the Department of Master of Computer Applications, for his timely suggestions and co-operation for my project completion.

I would like to express my heartfelt gratitude to our Principal, **Dr. G. M. V. PRASAD, M.Tech, PhD, FIETE, FIE, MIEEE, MSEMC, MISTE**, for forecasting an excellent academic environment and support.

I would like to extend my sincere thanks to all of our department faculty members, technicians and my family members for their help in completing the project.

Tejaswi Mattaparthi

Reg.No:21H41F0042

BONAM VENKATA CHALAMAYYA INSTITUTE OF TECHNOLOGY AND SCIENCE::BATLAPALEM

DEPARTMENT OF COMPUTER APPLICATIONS (MCA)

VISION

To become a center of excellence in the field of Computer Applications that produces innovative, skillful and socially responsible professionals who can contribute significantly to industry.

MISSION

1. To impart technical skills in the field of Computers and IT to meet the current trends and future challenges.
2. To offer high level skills and impart righteous attitude to succeed in the field of computer applications.
3. To provide knowledge of emerging trends in computers and related sphere to bridge the gap between industry and academia to suit social needs.
4. To produce well equipped professionals who can investigate, research and find feasible Solutions to complex industrial problems with an awareness and concern for the ever changing society.

PROGRAM EDUCATIONAL OBJECTIVES (PEO)

PEO 1: Graduates will demonstrate and apply principles of Sciences, Mathematics, and computer application fundamentals to solve and analyze IT problems.

PEO 2: Graduates able to take up positions as system analysts, system engineers, software engineers and programmers or become entrepreneurs.

PEO 3: Graduates exhibit social, ethical and professional responsibility work in multi – disciplinary projects with their communication and soft skills.

**BONAM VENKATA CHALAMAYYA INSTITUTE OF
TECHNOLOGY AND SCIENCE:: BATLAPALEM**

DEPARTMENT OF COMPUTER APPLICATIONS (MCA)

PROGRAMME OUTCOMES (PO)

PO 1: Computational Knowledge: Apply knowledge of computing fundamentals, computing specialization, mathematics, and domain knowledge appropriate for the computing specialization to the abstraction and conceptualization of computing models from defined problems and requirements.

PO 2: Problem Analysis: Identify, formulate, research literature, and solve complex computing problems reaching substantiated conclusions using fundamental principles of mathematics, computing sciences, and relevant domain disciplines.

PO 3: Design /Development of Solutions: Design and evaluate solutions for complex computing problems, and design and evaluate systems, components, or processes that meet specified needs with appropriate consideration for public health and safety, cultural, social, and environmental considerations.

PO 4: Conduct investigations of complex Computing problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO 5: Modern Tool Usage: Create, select, adapt and apply appropriate techniques, resources, and modern computing tools to complex computing activities, with an understanding of the limitations.

PO 6: Professional Ethics: Understand and commit to professional ethics and cyber regulations, responsibilities, and norms of professional computing practices.

PO 7: Life-long Learning: Recognize the need, and have the ability, to engage in independent learning for continual development as a computing professional.

PO 8: Project management and finance: Demonstrate knowledge and understanding of the computing and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO 9: Communication Efficacy: Communicate effectively with the computing community, and with society at large, about complex computing activities by being able to comprehend and write effective reports, design documentation, make effective presentations, and give and understand clear instructions.

PO 10: Social and Environmental Concern: Understand and assess societal, environmental, health, safety, legal, and cultural issues within local and global contexts, and the consequential responsibilities relevant to professional computing practices.

PO 11: Individual and Team Work: Function effectively as an individual and as a member or leader in diverse teams and in multidisciplinary environments.

PO 12: Innovation and Entrepreneurship: Identify a timely opportunity and using innovation to pursue that opportunity to create value and wealth for the betterment of the individual and society at large.

PROGRAM SPECIFIC OUTCOMES (PSO)

PSO 1: Technical Skills: Capable of integrating several scientific and technical disciplines in the area of Computer Applications.

PSO 2: Design: Ability to analyze problems, design algorithms, identifies and define the computing requirements appropriate to its solution and implement the same.

PROJECT WORK COURSE OUTCOMES

Students will be able to

- C321.1** Simulate or develop a program or prototype for his/her project (Applying)
- C321.2** analyze, compare and discuss their results and models and present his/her work to the panel. (Analyzing)
- C321.3** Utilize conventional or latest technologies for problem solving and identify the future enhancement for the project work. (Applying)
- C321.4** Design models, database and test cases and use tools for testing a project (Creating)
- C321.5** Make use of literature survey and analyze it.(Evaluating)
- C321.6** Build a thesis or report in a required format and present their work to the panel. (Creating)

ABSTRACT

ABSTRACT

In this project, Theme to Accent Transformation Application is developed that converts text into spoken words. The application is developed based on Django for Python and in the form of an interactive web page which is connected to an FPT.AI server through its application programming interface (API). The FPT stands for Financing and Promoting Technology. The application supports conversion of text to seven different Vietnamese speeches. Four out of seven voices can be used to convert up to 500 characters in a single transaction while the others support that of 400 characters. Based on the results obtained, the first conversion time takes up to 10 s to convert 400-character text into speech while the subsequent times, given same text, it takes under 1.8 s for the conversion. This is applicable to all voices. Speech synthesis is the fundamental component of many artificial intelligence systems..

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
1	INTRODUCTION	1
	1.1 Literature survey	5
2	SYSTEM ANALYSIS AND DESCRIPTION	7
	2.1 Existing System	7
	2.1.1 Disadvantages	7
	2.2 Proposed System	7
	2.2.1 Advantages	7
	2.3 System Design	8
	2.4 System Modules	8
	2.4.1 User	8
	2.4.2 Admin	8
	2.4.3 Data Preprocess	9
	2.4.4 FPT.AI Server	9
	2.5 System Architecture	10
	2.6 Data Flow Diagram	11
	2.7 Feasibility Study	11
	2.7.1 Economical Feasibility	12
	2.7.2 Technical Feasibility	12
	2.7.3 Social Feasibility	12
3	REQUIREMENT ANALYSIS	13
	3.1 Requirement Specification	13
	3.1.1 Functional Requirements	14
	3.1.2 Non-Functional Requirements	15
	3.2 System Specification	16
	3.2.1 Hardware Requirements	16
	3.2.2 Software Requirements	16
4	SOFTWARE DESIGN	17
	4.1 Input and Output Design	17
	4.2 Unified Modeling Language	18
	4.2.1 Use case Diagram	19

	4.2.2 Class Diagram	20
	4.2.3 Sequence Diagram	22
	4.2.4 Activity Diagram	23
	4.2.5 Deployment Diagram	23
	4.2.6 Component Diagram	24
5	IMPLEMENTATION	25
	5.1 Machine Learning Technology	25
	5.1.1 Machine Learning	25
	5.1.2 Steps involved in machine learning	25
	5.1.3 Machine Learning vs Traditional Programming	26
	5.1.4 How does Machine Learning work	26
	5.1.5 Machine Learning Algorithms	27
	5.1.6 Types of Machine Learning	27
	5.1.7 Challenges and Limitations	30
	5.1.8 Applications	30
	5.1.9 Why Machine Learning	31
	5.2 Python	32
	5.3 Python Libraries	32
	5.3.1 NumPy	32
	5.3.2 Pandas	32
	5.3.3 Matplotlib	32
	5.3.4 Scikit-learn	33
	5.3.5 Seaborn	33
	5.4 Uses of Python	33
	5.5 Sample Coding	34
6	TESTING	40
	6.1 About Testing	40
	6.2 Types of Testing	40
	6.3 Test Cases	43
7	SAMPLE SCREENS	44
8	CONCLUSION	50
9	FUTURE ENHANCEMENTS	51
10	BIBLIOGRAPHY	52

LIST OF FIGURES

S.NO	NAME OF THE FIGURE	PAGE NO
1	Fig 2.5: System Architecture for text to speech	10
2	Fig 2.6: Data Flow Diagram	11
3	Fig 4.2.1: Use case diagram	20
4	Fig 4.2.2: Class Diagram	21
5	Fig 4.2.3: Sequence Diagram	22
6	Fig 4.2.4: Activity Diagram	23
7	Fig 4.2.5:State Chart Diagram	23
8	Fig 4.2.6: Deployment Diagram	23
9	Fig 4.2.7: Component Diagram	24

LIST OF TABLES

S.NO	NAME OF THE TABLE	PAGE NO
1	Tab: 5.1 Algorithms in Supervised Machine Learning	28
2	Tab: 5.2 Algorithms in Unsupervised Machine Learning	29
3	Tab: 6.1 Test case specification	43

CHAPTER 1

INTRODUCTION

In recent years, the development of automated responding systems (i.e., Chabot, voicebot) has enabled the extensive use of speech-to-text (STT) and text-to-speech (TTS) systems. The latter enables a system to speak out a given input text with a human voice. This provides a close-to-nature response to user who interacts with the system. In addition, user experiences with the system can be significantly improved since the system can provide promptly responses to any requests. However, the quality of the responses depends on multiple factors such as input signal quality, understanding of request and context, and conversion engine. The most important one would be the engine. There are several parameters that can be used for indicating the performance of a TTS system. The most common parameter is mean opinion score (MOS) which is broadly used to measure the naturalness of the generated speech. However, this is not enough to indicate the performance of a system from customer perspective. The reason is that end-users only experience end-to-end TTS conversion, therefore, even if the core engine is very fast, the intermediate communication media may affect how fast the system can perform the conversion. Thus, in this research, end-to-end conversion speed of an FPT.AI-based TTS application is analyzed with the main focus on the relationship between the length of the input text and its end-to-end conversion time.

The main contributions of this research are: (i) an FPT.AI based TTS application using Django for Python, (ii) performance analysis of the application for seven different supported voices and several lengths of input text.

About FPT.AI API

In this work, FPT.AI API is used for interfacing between local host and remote FPT TTS server. This represents an actual working condition when an external user needs to use the API for converting text to speech. The TTS API takes four arguments for forming an Hypertext Transfer Protocol (HTTP) request before posting it to the server: (i) POST Data which contain text to be converted to speech; (iiiv) voice, speed and prosody to form HTTP header. The latter three arguments are not required by the TTS server during a request. The reason is that by default, normal speed (speed = 0) and female voice (see Table I) (Thu Dung - Northern Female) are used. The prosody is only available for question working with male voice. Hence, in this work, it is not considered for performance analysis.

The available voices supported by FPT.AI engine. These voices are provided for trial users to convert up to 10,000 characters to speech monthly. For each request, a response will be returned to host application by the server. It has JavaScript Object Notation (JSON) format and contains a static HTTP link to download the converted audio file in *.mp3 format. In addition, the response has an error-or-success indicator plus a message detailing the system's feedback. Since, it may take some time for the server to generate the audio file, this work aims to assess the end-to-end conversion timing of the system.

Proper use of expression makes speech more interesting and aids understanding of content by adding nuances and information beyond the pure text content. Expressiveness in speech includes emotions (e.g. angry, sad), speaking style (e.g. whisper, boasting), and the “character voices” often used in story reading (e.g. “old man”). While state-of-the-art TTS systems achieve high intelligibility and naturalness for reading isolated sentences in a relatively neutral style, the synthesis of expressive speech is still a challenge. The first key components in expressive text-to-speech synthesis (ETTS) are the speech corpus used and the annotation of “expression” in that corpus. In most approaches to date a separate speech corpus was recorded for each expression of interest. However, creating such dedicated speech corpora is costly and time-consuming, and typically limited to a handful of expressions, lacking generalization. Researchers have started to explore using audiobooks e.g. as a source of training data for TTS due to the wide variety of expressions contained. This paper presents an approach to derive ETTS from audiobook data. Unlike dedicated emotional corpora, the emotions and speaking styles are mixed in audiobooks. There is no standard set of classes for annotating such data so manual annotation is highly subjective, with poor inter-annotator agreement. The size of such corpora (10-20 hours) also makes annotation of multiple books impractical in terms of time and cost. Therefore, an unsupervised clustering approach is taken here to produce automatic expression annotations.

The aim of the clustering is to place similar (emotion and style) short book units into the same or proximate clusters. Recently Sleekly et al. have proposed self-organizing feature maps to cluster audiobook data based on voice quality (but no TTS was built). For this work, hierarchical k-means clustering is used based on acoustic features derived from work in emotion recognition. This approach was chosen due to its simplicity and linear scalability. Once the expressive clusters are learnt, ETTS can be trained. HMM-based

speech synthesis is used to provide a flexible framework to model the varying expressions. Two approaches are investigated: incorporating the cluster labels as context features within the decision tree creation process cf. ; and average voice speech synthesis with the expressive clusters acting as “speakers”. The final step in realizing ETTS is to determine the appropriate expression at synthesis. There are 3 main scenarios for expression derivation: from text (audiobook reading); given by external sources (in a dialog system or manually specified); and from audio (speech-to-speech translation). The unsupervised clusters do not have a human-readable label such as “sad”. However, a machine learner could be trained to predict the correct expressive cluster. Alternatively, for the manual specification case a user might get a “feel” for the clusters by listening to synthesized examples, possibly assigning their own, informal labels, prior to selecting the cluster for a specific utterance. In this paper the issue of how to choose the appropriate cluster is not investigated. It is assumed that the cluster can be chosen reliably Since there is no agreement of what the optimum set of expressive annotation labels for an audiobook is, unsupervised clustering is used to place similar (emotion, speaking style and character voices) book units into the same or proximate clusters. To cluster the training data into expressions three questions, need to be addressed: (i) the linguistic level at which the features will be labelled, (ii) the nature of the features, and (iii) the clustering approach. Emotion recognition and classification is typically performed at the sentence level. This is because the longer the unit of analysis is, the more stable the result provided the emotion does not change. A lot of the expressive speech in audiobooks occurs within direct speech whereas the associated carrier phrases, such as ‘he said’, tend to be neutral in style. Therefore, for clustering, the audiobook sentences were sub-divided into three types of units: narration, carrier and direct speech. The presence of double quotation marks in the audiobook text was used to detect direct speech. For example, a sequence such as: He said angrily, "Yes! I know. Bye!", and hung up. Then he left. would be split into the 6 underlined units: an initial carrier unit (a part not in quotes of a sentence containing quotes), three direct speech units, another carrier, and a narration unit (a complete sentence not inside or containing quotes). In this example, the reader would be expected to convey the angry tone only in the direct speech units. There are three primary options in terms of the feature sets that can be used for the unsupervised clustering: acoustic-only; text only; acoustics and text. In synthesis generally only the text will be available. Typically for emotion recognition and classification only acoustic features are used,

although there has been some very recent work on combining with text features. For simplicity only acoustic features were considered in this work. The feature extraction followed the commonly used supra-segmental modelling approach. In this a single, high-dimensional feature vector is extracted for a speech unit of variable length (e.g. a sentence) e.g. Low-level acoustic features are extracted on a frame level and mapped to the unit level via functional (mean, standard deviation, etc.). To derive the feature vector, the Inter speech 2011 Speaker State Challenge set was used as a starting point. This has 4 ,368 features. The set includes a number of items which are not (or poorly) related to expressiveness, such as spectral features. Therefore, the list of low-level descriptors was reduced to the following prosodic descriptors: F0, voicing probability, loudness, voice quality (local jitter and shimmer, logarithmic HNR). For these descriptors, functional including arithmetic mean, flatness, standard deviation, and skewness were applied to yield a set of 163 features, called feat A. Initial experiments using feat A indicated a link between utterance length and cluster assignment. This was found to be due to some features in feat A being highly correlated with the utterance length. A new set of features was produced by removing all features with a correlation coefficient (wrt. utterance length) higher than 0.2. A total of 69 features remained in this set, feat B. Motivated by, two manually selected feature sets were also created with a minimal set of prosodic and voice quality descriptors and their functional: feat C 8 features: mean of F0, voicing probability (pv), local jitter and shimmer, and logarithmic HNR; standard deviation of F0; mean of absolute delta of F0 and pv. Feat D 4 features: mean of F0 and pv; mean of absolute delta of pv; standard deviation of F0. Before clustering all feature vectors were standardized to have zero mean and unit variance. This ensures that all features are equally accounted for and that badly scaled features do not bias the clustering. An unsupervised clustering approach is required. For this work hierarchical k-means clustering, similar to the x-means algorithm described in, was applied in a cascade of hierarchical binary The majority of quoted material is direct speech. However, there are also “other” uses of quotes. In the present work, these uses were not distinguished. splits. The number of leaf clusters was controlled using BIC and a minimum cluster occupancy criterion of 20. A maximum tree depth of 5 levels limited the maximum number of leaf clusters to 32. A Euclidean distance metric was used to determine which clusters to split. To improve the stability of the algorithm, the initial cluster centers in each split were

initialized heuristically with a small perturbation to the left and right of the original centroid.

LITERATURE SURVEY

1) Content-Oriented User Modeling for Personalized Response Ranking in Chabot's

AUTHORS: B. Liu et al

Automatic Chabot's (also known as chat-agents) have attracted much attention from both researching and industrial fields. Generally, the semantic relevance between users' queries and the corresponding responses is considered as the essential element for conversation modeling in both generation and ranking based chat systems. By contrast, it is a nontrivial task to adopt the users' information, such as preference, social role, etc., into conversational models reasonably, while users' profiles play a significant role in the procedure of conversations by providing the implicit contexts. This paper aims to address the personalized response ranking task by incorporating user profiles into the conversation model. In our approach, users' personalized representations are latently learned from the contents posted by them via a two-branch neural network. After that, a deep neural network architecture is further presented to learn the fusion representation of posts, responses, and personal information. In this way, the proposed model could understand conversations from the users' perspective; hence, the more appropriate responses are selected for a specified person. The experimental results on two datasets from social network services demonstrate that our approach is hopeful to represent users' personal information implicitly based on user generated contents, and it is promising to perform as an important component in Chabot's to select the personalized responses for each user.

2) Ensemble-based deep reinforcement learning for Chabot's

AUTHORS: H. Cuayáhuitl et al

Trainable Chabot's that exhibit fluent and human-like conversations remain a big challenge in artificial intelligence. Deep Reinforcement Learning (DRL) is promising for addressing this challenge, but its successful application remains an open question. This article describes a novel ensemble-based approach applied to value-based DRL Chabot's, which use finite action sets as a form of meaning representation. In our approach, while dialogue actions are derived from sentence clustering, the training datasets in our ensemble are derived from dialogue clustering. The latter aim to induce specialised agents that learn to interact in a particular style. In order to facilitate neural chatbot training using our proposed approach, we assume dialogue data in raw text only – without any

manually-labelled data. Experimental results using chitchat data reveal that (1) near human-like dialogue policies can be induced, (2) generalisation to unseen data is a difficult problem, and (3) training an ensemble of Chabot's agents is essential for improved performance over using a single agent. In addition to evaluations using held-out data, our results are further supported by a human evaluation that rated dialogues in terms of fluency, engagingness and consistency – which revealed that our proposed dialogue rewards strongly correlate with human judgements.

CHAPTER - 2

SYSTEM ANALYSIS AND DESCRIPTION

2.1 Existing System:

In the existing system there are several parameters that can be used for indicating the performance of a TTS system. The most common parameter is mean opinion score (MOS) which is broadly used to measure the naturalness of the generated speech. However, this is not enough to indicate the performance of a system from customer perspective. The reason is that end-users only experience end-to-end TTS conversion, therefore, even if the core engine is very fast, the intermediate communication media may affect how fast the system can perform the conversion.

2.1.1 DISADVANTAGES:

- Cloud Based solution not available for the MOS based TTS.
- The Converted file into mp3 is not downloadable to the user.
- It is simple Speech engine gtts
- **Algorithm:** mean opinion score (MOS)

2.2 PROPOSED SYSTEM:

The proposed System end-to-end conversion speed of an FPT.AI-based TTS application is analyzed with the main focus on the relationship between the length of the input text and its end-to-end conversion time. The main contributions of this research are: (i) an FPT.AI based TTS application using Django for Python, (ii) performance analysis of the application for seven different supported voices and several lengths of input text. In this work, FPT.AI API is used for interfacing between local host and remote FPT TTS server. This represents an actual working condition when an external user needs to use the API for converting text to speech.

2.2.1 ADVANTAGES:

- Each registered user is allowed to use the TTS service for multiple requests amounting a total of 10,000 characters monthly.
- There are three main input parameters that user can key into the system: text to convert to speech, the desired speed of generated speech and voice type.
- For each request, a response will be returned to host application by the server. It has JavaScript Object Notation (JSON) format and contains a static HTTP link to

download the converted audio file in *.mp3 format. In addition, the response has an error-or-success indicator.

- **Algorithm:** FPT.AI core engine

2.3 SYSTEM DESIGN:

Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of engineered product or system. The designer's goal is to produce a model or representation of an entity that will later be built. Beginning, once system requirement has been specified and analysed, system design is the first of the three technical activities –design, code and test that is required to build and verify software. The importance can be stated with a single word “Quality”.

2.4 SYSTEM MODULES:

- **User**
- **Admin**
- **Data Preprocess**
- **FPT.AI Server**

User:

The User can register the first. While registering he required a valid user email and mobile for further communications. Once the user register then admin can activate the customer. Once admin activated the customer then user can login into our system. The text data size is 500 characters. The user can enter the data; he can select voices which is configured in the project. There are 7 voices of Vietnamese user can select anyone. The user also selects the speed. Then our programs will make a post request to the fpt server. The sever will check the access key if access key valid then fpt.ai server will create a .mp3 file and that file stored in cloud. The response will send to the user is link. That link we can configure in client side with html5 tag.

Admin:

Admin can login with his credentials. Once he login he can activate the users. The activated user only login in our applications. The admin can configure the access key in the program. Without access key the fpt server will not give the response. This information for each user we will add to header file. The header file contains the access

key and we are making a post request to fpt server. Later admin can check all the user speech data and time taken data.

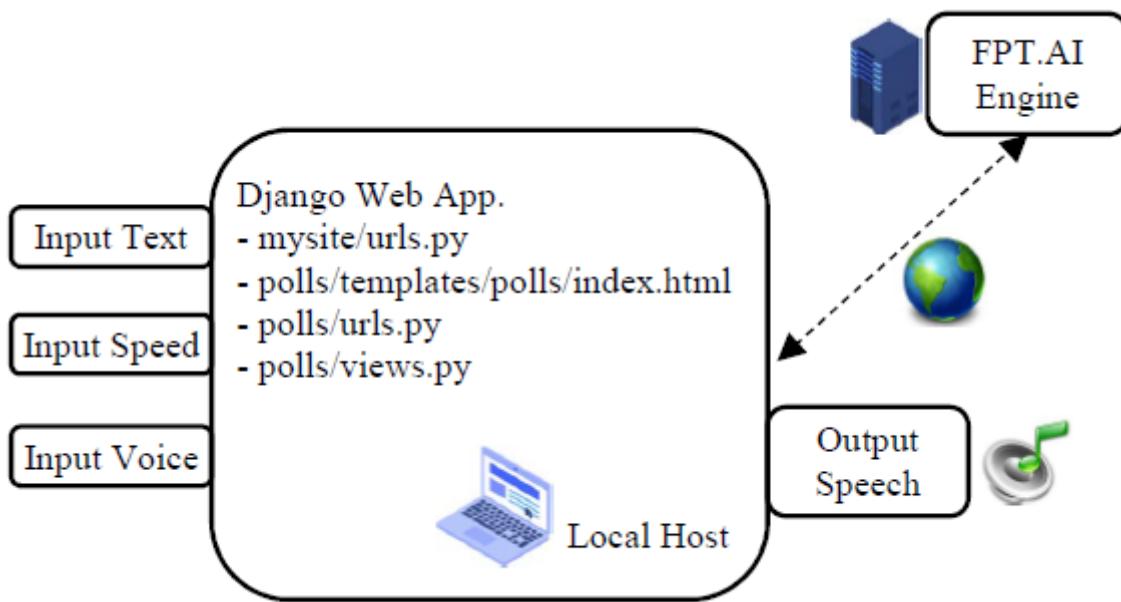
Data Preprocess:

It is seen that there are three main input parameters that user can key into the system: text to convert to speech, the desired speed of generated speech and voice type. These inputs are fed into the web application developed based on Django for Python. This application is used at local host. There are four main files which are at most important to the application. The urls.py file located in my site folder is used to initialize a localhost which address is <http://127.0.0.1:8000/>. When user runs the application, it will either return an address to admin folder containing administration information or synchronize with file urls.py located in Text to Speech folder to connect server and client by their addresses.

FPT.AI Server:

FPT.AI Server is used for interfacing between local host and remote FPT TTS server. This represents an actual working condition when an external user needs to use the API for converting text to speech. The TTS API takes four arguments for forming an Hyper Text Transfer Protocol (HTTP) request before posting it to the server POST Data which contain text to be converted to speech; voice, speed and prosody to form HTTP header. The latter three arguments are not required by the TTS server during a request. The reason is that by default, normal speed (speed = 0) and female voice (see Table I) (Thu Dung – Northern Female) are used. The prosody is only available for question working with male voice. Hence, in this work, it is not considered for performance analysis. These voices are provided for trial users to convert up to 10,000 characters to speech monthly

2.5 SYSTEM ARCHITECTURE:



2.6 DATA FLOW DIAGRAM:

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

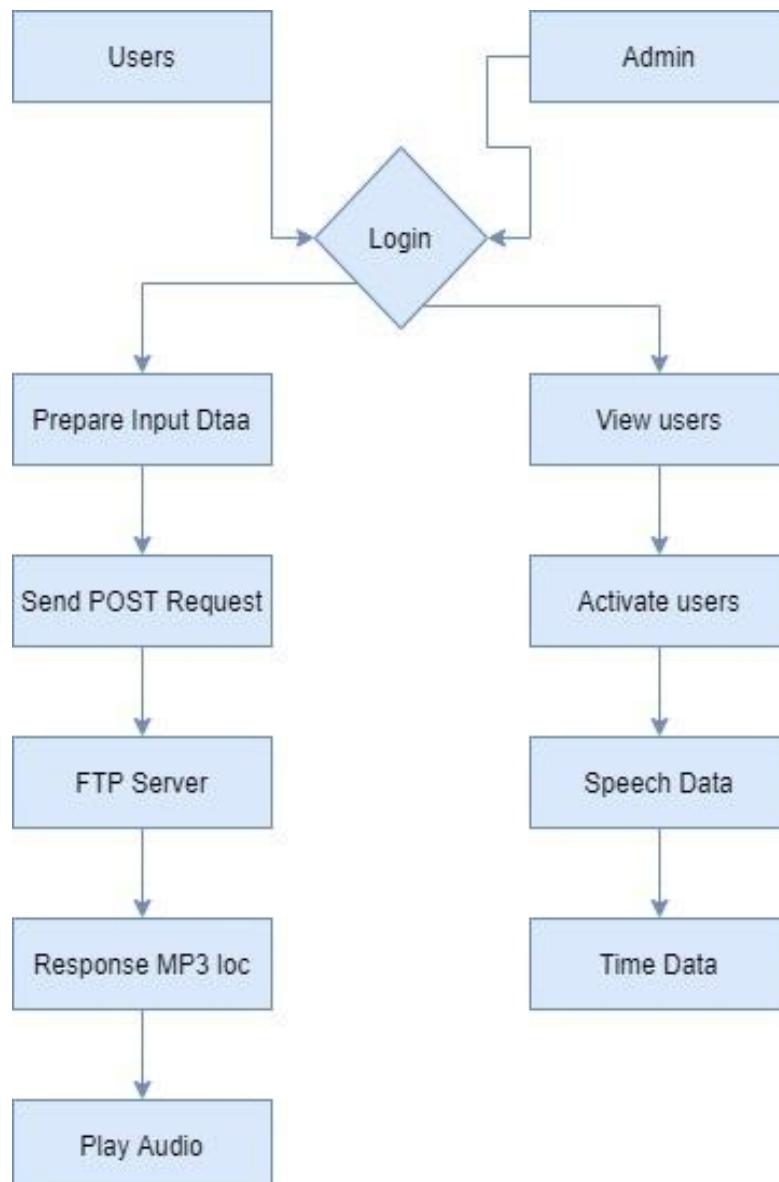


Fig 2.6: Data Flow Diagram

2.7 FEASIBILITY STUDY:

The feasibility of the project is analysed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are,

- ◆ **ECONOMICAL FEASIBILITY**
- ◆ **TECHNICAL FEASIBILITY**

◆ **SOCIAL FEASIBILITY**

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system was kept well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement; as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

CHAPTER - 3

REQUIREMENT ANALYSIS

3.1 REQUIREMENT SPECIFICATION:

A Software Requirements Specification (SRS) – A requirements specification for a software system – is a complete description of the behavior of a system to be developed. It includes a set of use cases that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. Nonfunctional requirements are requirements which impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A business analyst, sometimes titled system analyst, is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the systems development life cycle domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

- **Business requirements** describe in business terms what must be delivered or accomplished to provide value.
- **Product requirements** describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)
- **Process requirements** describe activities performed by the developing organization. For instance, process requirements could specify specific methodologies that must be followed, and constraints that the organization must obey.

Product and process requirements are closely linked. Process requirements often specify the activities that will be performed to satisfy a product requirement. For example, maximum development cost requirement (a process requirement) may be imposed to help achieve a maximum sales price requirement (a product requirement); a requirement that the product be maintainable (a Product requirement) often is addressed by imposing requirements to follow particular development style. A systems engineering, a

requirement can be a description of what a system must do, referred to as a functional requirement. This type of requirement specifies something that the delivered system must be able to do. Another type of requirement specifies something about the system itself, and how well it performs its functions. Such requirements are often called Non-functional requirements, or 'performance requirements' or 'quality of service requirements. Examples of such requirements include usability, availability, reliability, supportability, testability and maintainability. A collection of requirements define the characteristics or features of the desired system. A 'good' list of requirements as far as possible avoids saying how the system should implement the requirements, leaving such decisions to the system designer. Specifying how the system should be implemented is called "implementation bias" or "solution engineering". However, implementation constraints on the solution may validly be expressed by the future owner, for example for required interfaces to external systems; for interoperability with other systems; and for commonality (e.g. of user interfaces) with other owned products. In Software Engineering, the same meanings of requirements apply, except that the focus of interest is the software itself.

3.1.1 FUNCTIONAL REQUIREMENTS:

Functional requirements are very important system requirements in the system design process. These requirements are the technical specifications, system design parameters and guidelines, data manipulation, data processing, and calculation modules etc, of the proposed system. Functional requirements are in contrast to Non-Functional requirements which are descriptive of the parameters of system performance, quality attributes, reliability and security, cost, constraints in design/implementation, etc.

- After checking dataset prepared or not, the user upload dataset.
- After completion of dataset uploading to check it performing the view dataset function.
- The system had the pre-processor and the pre-processor verify the dataset.
- The pre-processor performs feature extraction to give the separation of train data & test data in the form of percentage.
- After separation of test data, the system run the three algorithms Decision Tree, Naïve Bayesian, Hyper-heuristic to show the accuracy of finding intrusions

- Finally, after execution of the three algorithms a plotted graph shown by giving the accuracy on their performance of test data.

3.1.2 NON- FUNCTIONAL REQUIREMENTS:

Non-functional requirements tend to be stated in terms of constraints on the results of tasks which are given as functional requirements (e.g., constraints on the speed or efficiency of a given task), a task based functional requirements statement is a useful skeleton upon which to construct a complete requirements statement. That is the approach taken in this work. It can be helpful to think of non-functional requirements as adverbially related to tasks or functional requirements.

Non-functional requirements are often called qualities of a system. Other terms for non-functional requirements are “constraints”, “quality attributes”, “quality goals”, “quality of service requirements” and “no behavioral requirements”.

Accessibility:

Accessibility is a general term used to describe the degree to which a product, device or service is available to as many people as possible. Accessibility can be viewed as the “ability to access” and benefit from some system or entity. Accessibility is often used to focus on people with disabilities or special needs and their right of access to entities, often to use of assistive technology. This project is accessible for understanding of, finding the cyber-attacks on Big Data.

Availability:

Availability is the degree to which a system, sub-system, or equipment is in a specified operable and committable state at the start of a mission, when a mission is called for at an unknown, i.e., a random, time. Simply put, availability is the proportion of time a system is in a functioning condition. This is often described that svm provides security for the data set from intrusions.

Scalability:

Scalability is the ability of a system, network, or process, to handle growing amount of work in a capable manner or its ability to be enlarged to accommodate that growth. Here the scalability is checked by running the three algorithms effectively to get the intrusions accuracy graph representation.

Portability:

Portability in high-level computer programming is the usability of the same software in different environments. The pre requirement for portability is the generalized abstraction between the application logic and system interfaces. Our model based on machine learning is portable because it runs on different environments.

3.2 SYSTEM SPECIFICATION:**3.2.1 HARDWARE REQUIREMENTS:**

- ❖ System : Intel i3
- ❖ Operating system : Windows 10.
- ❖ Hard Disk : 1 TB.
- ❖ Monitor : 14' Colour Monitor.
- ❖ Mouse : Optical Mouse.
- ❖ Ram : 4GB.

3.2.2 SOFTWARE REQUIREMENTS:

- ❖ Coding Language : Python.
- ❖ Front-End : Html, CSS,javascript
- ❖ Data Base : SQL

CHAPTER – 4

SOFTWARE DESIGN

4.1 INPUT AND OUTPUT DESIGN

INPUT DESIGN:

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

OBJECTIVES:

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.
2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.
3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus, the objective of input design is to create an input layout that is easy to follow.

OUTPUT DESIGN:

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.
2. Select methods for presenting information.
3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

4.2 Unified Modeling Language:

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS OF UML:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

4.2.1 USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

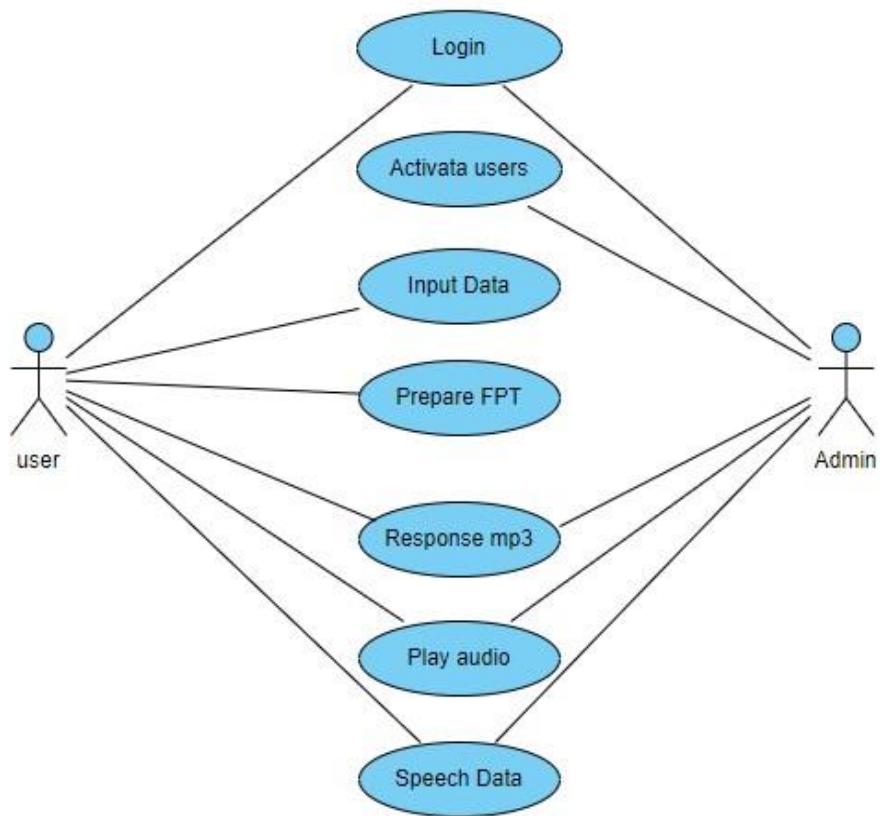


Fig 4.2.1 Use Case Diagram

4.2.2 CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

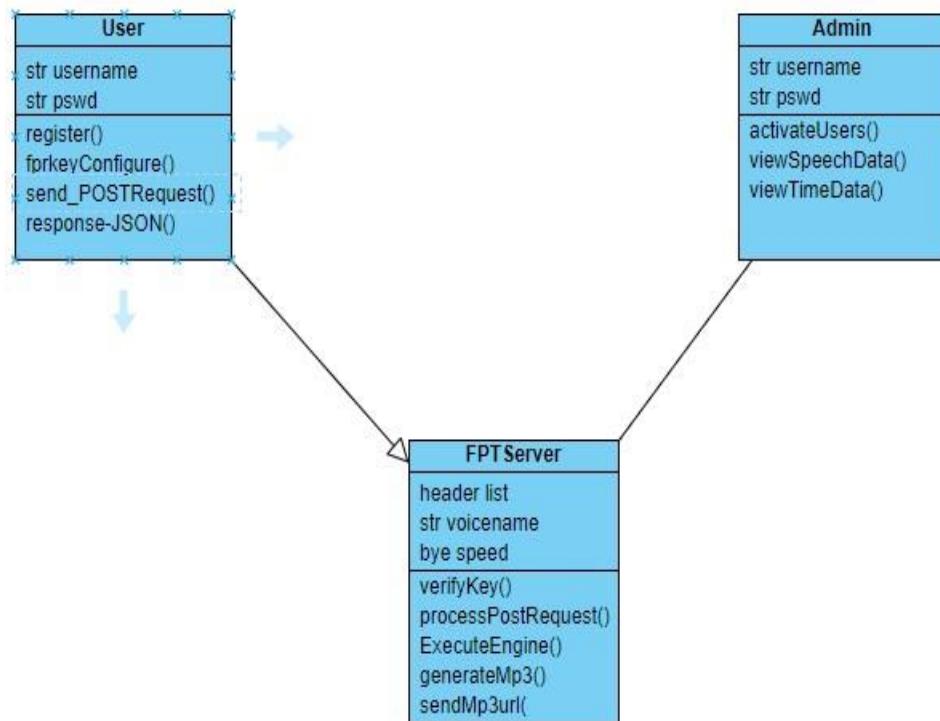


Fig:4.2.2 Class Diagram

4.2.3 SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

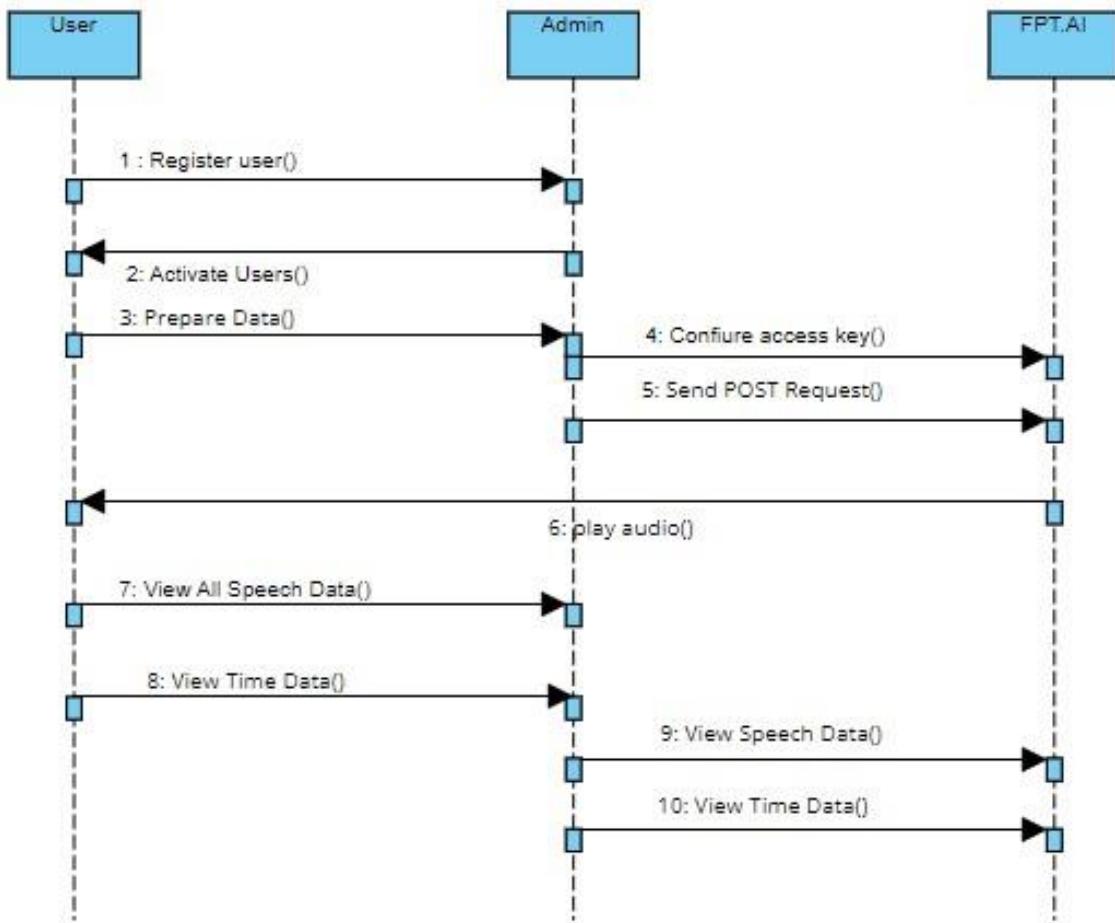


Fig 4.2.3 Sequence diagram

4.2.4 ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

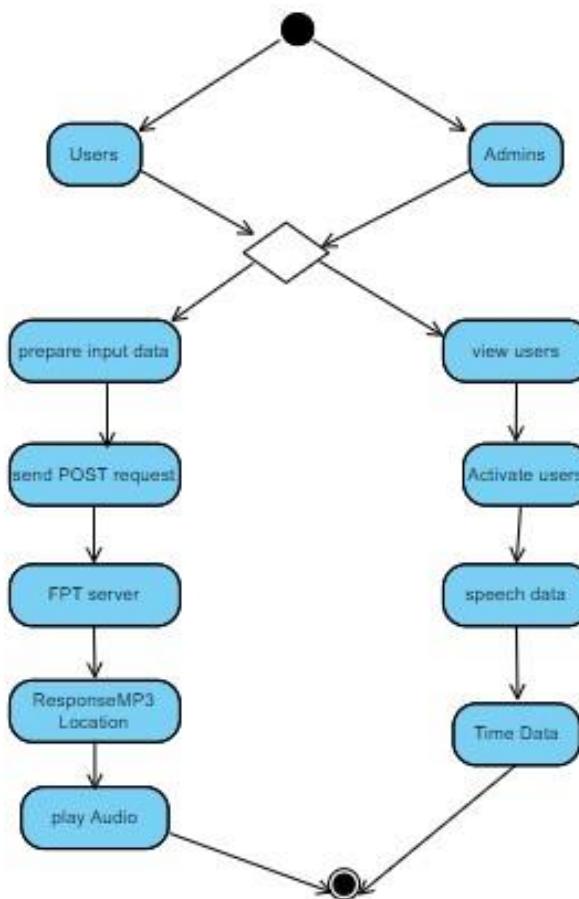
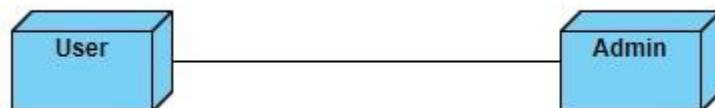


Fig 4.2.4 Activity Diagram

4.2.5. DEPLOYMENT DIAGRAM:

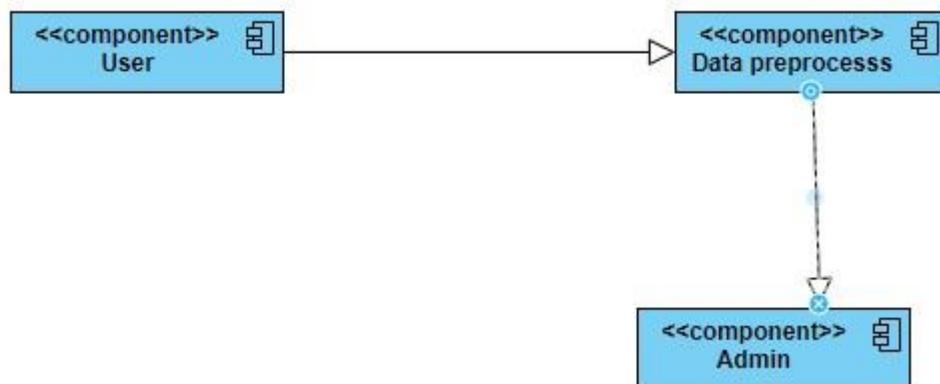
Deployment diagram represents the deployment view of a system. It is related to the component diagram because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware used to deploy the application. Deployment diagrams are useful for system engineers.



4.2.6 COMPONENT DIAGRAM:

Component diagram is a special kind of diagram in UML. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the components used to make those functionalities. Thus,

from that point of view, component diagrams are used to visualize the physical components in a system. These components are libraries, packages, files, etc.



CHAPTER - 5

IMPLEMENTATION

5.1 Machine Learning Technology:

5.1.1 Machine Learning:

Machine learning is a discipline that deals with programming the systems so as to make them automatically learn and improve with experience. Here, learning implies recognizing and understanding the input data and taking informed decisions based on the supplied data. It is very difficult to consider all the decisions based on all possible inputs. To solve this problem, algorithms are developed that build knowledge from a specific data and past experience by applying the principles of statistical science, probability, logic, mathematical optimization, reinforcement learning, and control theory.

5.1.2 Steps Involved in Machine Learning:

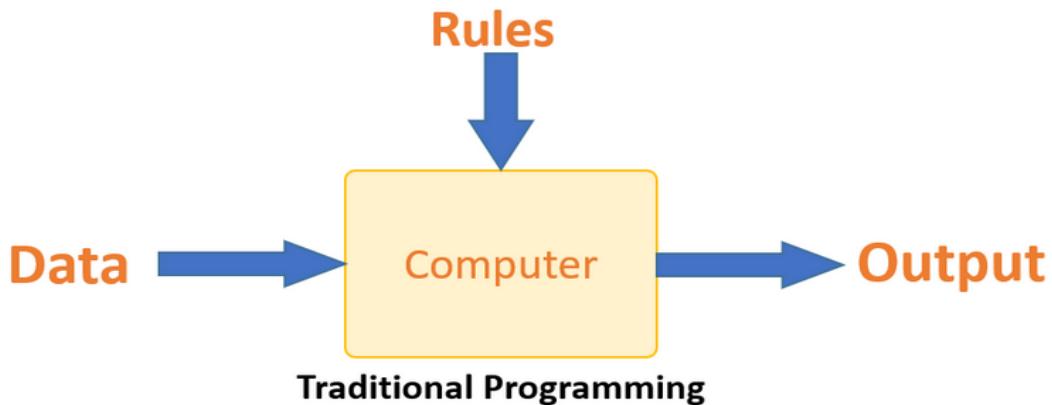
A machine learning project involves the following steps

- Defining a Problem
- Preparing Data
- Evaluating Algorithms
- Improving Results
- Presenting Results

Machine Learning (ML) is an automated learning with little or no human intervention. It involves programming computers so that they learn from the available inputs. The main purpose of machine learning is to explore and construct algorithms that can learn from the previous data and make predictions on new input data.

The input to a learning algorithm is training data, representing experience, and the output is any expertise, which usually takes the form of another algorithm that can perform a task. The input data to a machine learning system can be numerical, textual, audio, visual, or multimedia.

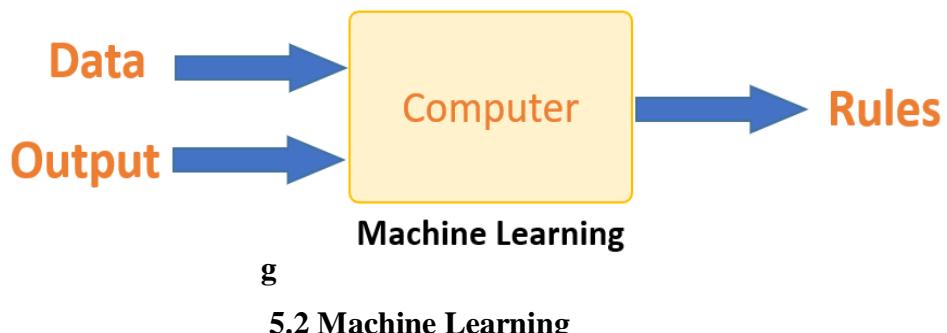
5.1.3 Machine Learning vs. Traditional Programming:



Traditional programming differs significantly from machine learning. In traditional programming, programmers code all the rules in consultation with an expert in the industry for which software is being developed. Each rule is based on a logical foundation; the machine will execute an output following the logical statement. When the system grows complex, more rules need to be written. It can quickly become unsustainable to maintain.

Fig 5.1: Traditional Programming

Machine learning is supposed to overcome this issue. The machine learns how the input and output data are correlated and it writes a rule. The programmers do not need to write new rules each time there is new data. The algorithms adapt in response to new data and experiences to improve efficacy over time.



5.2 Machine Learning

5.1.4 How does Machine learning work?

Machine learning is the brain where all the learning takes place. The way the machine learns is similar to the human being. Humans learn from experience. The more

we know, the more easily we can predict. By analogy, when we face an unknown situation, the likelihood of success is lower than the known situation. Machines are trained the same. To make an accurate prediction, the machine sees an example. When we give the machine a similar example, it can figure out the outcome. However, like a human, if it's feed a previously unseen example, the machine has difficulties to predict.

5.1.5 Machine learning Algorithms and where they are used?

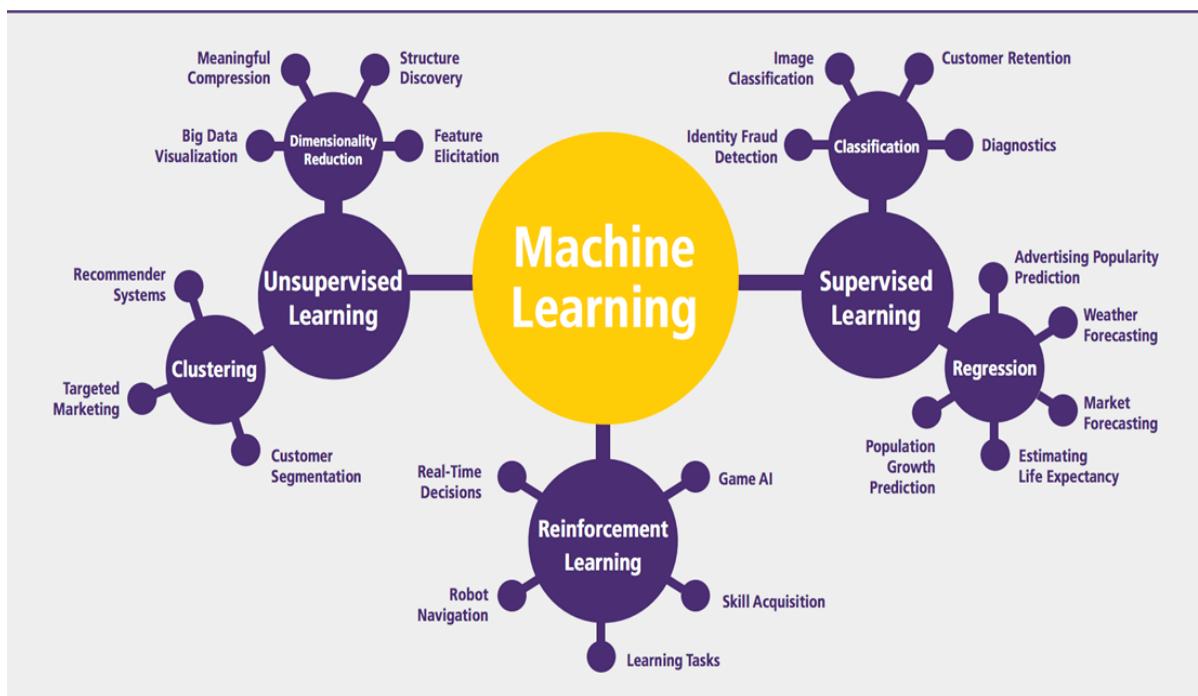


Fig 5.3: Machine Learning Algorithms

5.1.6 Types of Machine Learning:

There are four categories of machine learning algorithms as shown below

- Supervised learning algorithm
- Unsupervised learning algorithm
- Semi-supervised learning algorithm
- Reinforcement learning algorithm

Supervised Learning:

Algorithm Name	Description	Type
Linear regression	Finds a way to correlate each feature to the output to help predict future values.	Regression
Logistic regression	Extension of linear regression that's used for classification tasks. The output variable is binary (e.g., only black or white) rather than continuous (e.g., an infinite list of potential colors)	Classification
Decision tree	Highly interpretable classification or regression model that splits data-feature values into branches at decision nodes (e.g., if a feature is a color, each possible color becomes a new branch) until a final decision output is made	Regression Classification
Naive Bayes	The Bayesian method is a classification method that makes use of the Bayesian theorem. The theorem updates the prior knowledge of an event with the independent probability of each feature that can affect the event.	Regression Classification
Support vector machine	Support Vector Machine, or SVM, is typically used for the classification task. SVM algorithm finds a hyperplane that optimally divides the classes. It is best used with a non-linear solver.	Regression (not very common) Classification
Random forest	The algorithm is built upon a decision tree to improve the accuracy drastically. Random forest generates many times simple decision trees and uses the 'majority vote' method to decide on which label to return. For the classification task, the final prediction will be the one with the most vote; while for the regression task, the average prediction of all the trees is the final prediction.	Regression Classification
AdaBoost	Classification or regression technique that uses a multitude of models to come up with a decision but weighs them based on their accuracy in predicting the outcome	Regression Classification
Gradient-boosting trees	Gradient-boosting trees is a state-of-the-art classification/regression technique. It is focusing on the error committed by the previous trees and tries to correct it.	Regression Classification

Tab 5.1: Algorithms in Supervised Machine Learning

Supervised learning is commonly used in real world applications, such as face and speech recognition, products or movie recommendations, and sales forecasting. Supervised learning can be further classified into two types - Regression and Classification as shown in Tab 5.1. Regression trains on and predicts a continuous-valued response, for example, predicting real estate prices. Classification attempts to find the appropriate class label, such as analyzing positive/negative sentiment, male and female persons, benign and malignant tumors, secure and unsecured loans, etc.

In supervised learning, learning data comes with descriptions, labels, targets or desired outputs and the objective is to find a general rule that maps inputs to outputs. This kind of learning data is called labeled data. The learned rule is then used to label new data with unknown outputs. Supervised learning involves building a machine-learning model that is based on labeled samples.

Supervised learning deals with learning a function from available training data. The learning algorithm analyzes the training data and produces a derived function that can be used for mapping new examples.

Unsupervised Learning:

Unsupervised learning is used to detect anomalies, and outliers, such as fraud or defective equipment, or to group customers with similar behaviors for a sales campaign. It is the opposite of supervised learning. There is no labeled data here. When learning data contains only some indications without any description or labels, it is up to the coder or to the algorithm to find the structure of the underlying data, to discover hidden patterns, or determine how to describe the data. This kind of learning data is called unlabeled data. Suppose that we have a number of data points, and we want to classify them into several groups. We may not exactly know what the criteria of classification would be. So, an unsupervised learning algorithm tries to classify the given dataset into a certain number of groups in an optimum way. Unsupervised learning algorithms are extremely powerful tools for analyzing data and identifying patterns and trends.

Algorithm	Description	Type
K-means clustering	Puts data into some groups (k) that each contains data with similar characteristics (as determined by the model, not in advance by humans)	Clustering
Gaussian mixture model	A generalization of k-means clustering that provides more flexibility in the size and shape of groups (clusters)	Clustering
Hierarchical clustering	Splits clusters along a hierarchical tree to form a classification system. Can be used for Cluster loyalty-card customer	Clustering
Recommender system	Help to define the relevant data for making a recommendation.	Clustering
PCA/T-SNE	Mostly used to decrease the dimensionality of the data. The algorithms reduce the number of features to 3 or 4 vectors with the highest variances.	Dimension Reduction

Tab 5.2: Algorithms in Unsupervised Machine Learning

Semi-supervised Learning:

If some learning samples are labeled, but some other are not labeled, then it is semi-supervised learning. It makes use of a large amount of unlabeled data for training and a small amount of labeled data for testing. Semi-supervised learning is applied in

cases where it is expensive to acquire a fully labeled dataset while more practical to label a small subset.

Reinforcement Learning:

Here learning data gives feedback so that the system adjusts to dynamic conditions in order to achieve a certain objective. The system evaluates its performance based on the feedback responses and reacts accordingly.

5.1.7 Challenges and Limitations of Machine learning:

The primary challenge of machine learning is the lack of data or the diversity in the dataset. A machine cannot learn if there is no data available. Besides, a dataset with a lack of diversity gives the machine a hard time.

A machine needs to have heterogeneity to learn meaningful insight. It is rare that an algorithm can extract information when there are no or few variations. It is recommended to have at least 20 observations per group to help the machine learn. This constraint leads to poor evaluation and prediction.

5.1.8 Application of Machine learning:

Augmentation:

Machine learning, which assists humans with their day-to-day tasks, personally or commercially without having complete control of the output. Such machine learning is used in different ways such as Virtual Assistant, Data analysis, software solutions. The primary user is to reduce errors due to human bias.

Automation:

Machine learning, which works entirely autonomously in any field without the need for any human intervention. For example, robots performing the essential process steps in manufacturing plants.

Finance Industry:

Machine learning is growing in popularity in the finance industry. Banks are mainly using ML to find patterns inside the data but also to prevent fraud.

Government organization

The government makes use of ML to manage public safety and utilities. Take the example of China with the massive face recognition. The government uses Artificial intelligence to prevent jaywalker.

Healthcare industry

Healthcare was one of the first industry to use machine learning with image detection.

Marketing

Broad use of AI is done in marketing thanks to abundant access to data. Before the age of mass data, researchers develop advanced mathematical tools like Bayesian analysis to estimate the value of a customer. With the boom of data, marketing department relies on AI to optimize the customer relationship and marketing campaign.

5.1.9 Why is Machine Learning important?

Machine learning is the best tool so far to analyze, understand and identification of a pattern in the data. One of the main ideas behind machine learning is that the computer can be trained to automate tasks that would be exhaustive or impossible for a human being. The clear breach from the traditional analysis is that machine learning can take decisions with minimal human intervention.

5.2 Python:

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. An interpreted language, Python has a design philosophy that emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer lines of code than might be used in languages such as C++ or Java. It provides constructs that enable clear programming on both small and large scales. Python interpreters are available for many operating systems. Python,

the reference implementation of Python, is open-source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation. Python features a dynamic type system and automatic memory management.

It supports multiple programming paradigms, including object oriented, imperative, and functional and has a large and comprehensive standard library.

5.3 python libraries for machine learning:

5.3.1 NumPy:

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The core functionality of NumPy is its "ndarray", for n-dimensional array, data structure. These arrays are stride views on memory. In contrast to Python's built-in list data structure, these arrays are homogeneously typed: all elements of a single array must be of the same type.

5.3.2 Pandas:

Pandas is the most popular python library that is used for data analysis. It provides highly optimized performance with back-end source code is purely written in C or Python. We can analyse data in pandas with Series and Data Frames. Series is one dimensional (1-D) array defined in pandas that can be used to store any data type. Data Frames is two-dimensional (2-D) data structure defined in pandas which consists of rows and columns.

5.3.3 Matplotlib:

Matplotlib is a python library used to create 2D graphs and plots by using python scripts. It has a module named pyplot which makes things easy for plotting by providing feature to control line styles, font properties, formatting axes etc. It supports a very wide variety of graphs and plots namely histogram, bar charts, power spectra, error charts etc. It is used along with NumPy to provide an environment that is an effective open source alternative for MatLab. It can also be used with graphics toolkits like PyQt and wxPython.

5.3.4 Scikit-learn:

Scikit-learn is a machine learning library for Python. It features several regression, classification and clustering algorithms including SVMs, gradient boosting, k-means, random forests and DBSCAN. Scikit is written in Python (most of it) and some of its core algorithms are written in C python for even better performance. Scikit-learn is used to build models and it is not recommended to use it for reading, manipulating and summarizing data as there are better frameworks available for the purpose. It is open source and released under BSD license.

5.3.5 Seaborn:

Seaborn is an open source, BSD-licensed Python library providing high level API for visualizing the data using Python programming language. Data can be visualized by representing it as plots which is easy to understand, explore and grasp. Such data helps in drawing the attention of key elements. To analyse a set of data using Python, we make use of Matplotlib, a widely implemented 2D plotting library. Likewise, Seaborn is a visualization library in Python. It is built on top of Matplotlib.

5.4 Uses of Python:

1. Applications:

Python can be used to develop different applications like web applications, graphic user interface based applications, software development application, scientific and numeric applications, network programming, Games and 3D applications and other business applications. It makes an interactive interface and easy development of applications.

2. Multiple Programming paradigms:

Python is also used because of its providing continuous support to several programming paradigms. As it supports object-oriented programming and structured programming. Python has features, which also support various concepts of functional programming language. It is used for dynamic type system and automatic memory management. Python language features and programming paradigms allow you for developing the small as well as large applications. It can be used for complex software applications.

3. Robust Standard Library:

Python has a large and robust standard library to use for developing the applications. It also makes the developers use Python over other languages. The standard library helps in using the different range of modules available for Python. As this module helps you in adding the functionality without writing any more code. To get the information about various modules, documentation on python standard library can be referred. While developing any web application, implementing web services, performing string operations and other usages like interface protocol, the standard library documentation helps.

4. Compatible with Major Platforms and Systems:

Python is mainly compatible with major platforms and systems because of which it is used mainly for developing applications. With help of python interpreters, python code can be run on specific platforms and tools as it supports many operating systems. As python is an interpreted high-level programming language and it allows you to run the code on multiple platforms.

The new and modified code can be executed without recompiling and its impact can be monitored or checked. It means it's not required to recompile the code after every change. This feature helps in saving the development time of the developers.

5. Access of Database:

Uses of Python also helps in accessing the database easily. Python helps in customizing the interfaces of different databases like MySQL, Oracle, Microsoft SQL Server, PostgreSQL, and other databases. It has an object database like Durus and ZODB. It is used for standard database API and freely available for download.

5.5 SAMPLE CODE:

User Side views.py

```
from django.shortcuts import render,HttpResponse
from django.contrib import messages
from .forms import UserRegistrationForm
from .models import
UserRegistrationModel,UsersSpeechDataModel,UserTimeDurationModel
import requests
import time

# Create your views here.
def UserRegisterActions(request):
if request.method == 'POST':
```

```
form = UserRegistrationForm(request.POST)
if form.is_valid():
    print('Data is Valid')
    form.save()
    messages.success(request, 'You have been successfully registered')
    form = UserRegistrationForm()
    return render(request, 'UserRegistrations.html', {'form': form})
else:
    messages.success(request, 'Email or Mobile Already Existed')
    print("Invalid form")
else:
    form = UserRegistrationForm()
    return render(request, 'UserRegistrations.html', {'form': form})
def UserLoginCheck(request):
    if request.method == "POST":
        loginid = request.POST.get('loginname')
        pswd = request.POST.get('pswd')
        print("Login ID = ", loginid, ' Password = ', pswd)
        try:
            check = UserRegistrationModel.objects.get(loginid=loginid, password=pswd)
        def UserHome(request):
            return render(request, 'users/UserHome.html', {})
        def TextToSpeechForm(request):
            return render(request, 'users/UserTextToSpeechForm.html', {})
        def GenerateSpeaksforUser(request):
            input_text = request.POST["text"]
            speech_voice = request.POST["voice"]
            speech_speed = request.POST["speed"]
            url = 'https://api.fpt.ai/hmi/tts/v5'
            header = {
                'api-key': 'BJtsEZz3CCpeRCixZL6EvVpjPmhCewHQ',
                'speed': speech_speed,
                'voice': speech_voice
            }
            start = time.time()
            response = requests.request('POST', url, data=input_text.encode('utf-8'),
                                         headers=header)
            substr = response.text.split('')[3]
            end = time.time()
            secDiffe = end-start
            print("Start Time =", start, " End Time = ", end, " Difference = ", secDiffe)
            voice = speech_voice
            txtLen = len(input_text)
            # success
            context = {"link": substr}
            print("Context is ", context)
            loggeduser = request.session['loggeduser']
            loginid = request.session['loginid']
            email = request.session['email']
            UsersSpeechDataModel.objects.create(loggeduser=loggeduser,
                                                loginid=loginid, email=email, textdata=input_text, link=substr)
```

```
UserTimeDurationModel.objects.create(username=loginid,voicename=voice,timesec=
secDiffe,textlen=txtLen)
#return render(request, 'users/UserTextToSpeechForm.html',context)
return render(request, 'users/PlayAudioFile.html', context)
def UserSpeechData(request):
    loginid = request.session['loginid']
    data = UsersSpeechDataModel.objects.filter(loginid=loginid)
    return render(request, 'users/UserSpeechDataView.html', {'data':data})
def UserTimeDifference(request):
    loginid = request.session['loginid']
    data = UserTimeDurationModel.objects.filter(username=loginid)
    return render(request, 'users/UserTimeDataView.html', {'data':data})
```

user side forms.py

```
from django import forms
from .models import UserRegistrationModel
class UserRegistrationForm(forms.ModelForm):
    name = forms.CharField(widget=forms.TextInput(attrs={'pattern': '[a-zA-Z]+'}),
                           required=True, max_length=100)
    loginid = forms.CharField(widget=forms.TextInput(attrs={'pattern': '[a-zA-Z]+'}), required=True, max_length=100)
    password = forms.CharField(widget=forms.PasswordInput(attrs={'pattern':
        '(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{8,}'},
        'title': 'Must contain at least one number and one uppercase and lowercase
letter, and at least 8 or more characters'}),
                           required=True, max_length=100)
    mobile = forms.CharField(widget=forms.TextInput(attrs={'pattern': '[56789][0-
9]{9}'}), required=True,
                           max_length=100)
    email = forms.CharField(widget=forms.TextInput(attrs={'pattern': '[a-z0-9._%+-]@[a-z0-9.-]+\.[a-z]{2,}$'}),
                           required=True, max_length=100)
    locality = forms.CharField(widget=forms.TextInput(), required=True,
                               max_length=100)
    address = forms.CharField(widget=forms.Textarea(attrs={'rows': 4, 'cols':
22}), required=True, max_length=250)
    city = forms.CharField(widget=forms.TextInput(
        attrs={'autocomplete': 'off', 'pattern': '[A-Za-z ]+', 'title': 'Enter
Characters Only '}), required=True,
                           max_length=100)
    state = forms.CharField(widget=forms.TextInput(
        attrs={'autocomplete': 'off', 'pattern': '[A-Za-z ]+', 'title': 'Enter
Characters Only '}), required=True,
                           max_length=100)
    status = forms.CharField(widget=forms.HiddenInput(), initial='waiting',
                           max_length=100)

class Meta():
    model = UserRegistrationModel
    fields = '__all__'
```

Admin side views.py

```

from django.shortcuts import render,HttpResponse
from django.contrib import messages
from users.models import
UserRegistrationModel,UsersSpeechDataModel,UserTimeDurationModel
def AdminLoginCheck(request):
if request.method == 'POST':
usrid = request.POST.get('loginid')
pswd = request.POST.get('pswd')
print("User ID is = ", usrid)
if usrid == 'admin' and pswd == 'admin':
return render(request, 'admins/AdminHome.html')
elif usrid == 'Admin' and pswd == 'Admin':
return render(request, 'admins/AdminHome.html')
else:
messages.success(request, 'Please Check Your Login Details')
return render(request, 'AdminLogin.html', {})
def AdminHome(request):
return render(request, 'admins/AdminHome.html')
def ViewRegisteredUsers(request):
data = UserRegistrationModel.objects.all()
return render(request, 'admins/RegisteredUsers.html', {'data': data})
def AdminViewSpeechData(request):
data = UsersSpeechDataModel.objects.all()
return render(request, 'admins/AdminViewSpeech.html', {'data':data})

def AdminViewTimeDifference(request):
data = UserTimeDurationModel.objects.all()
return render(request, 'admins/AdminViewExeTime.html', {'data':data})

```

urls.py

```

from django.contrib import admin
from django.urls import path
from TextToSpeech import views as mainView
from users import views as usr
from admins import views as admins

urlpatterns = [
path('admin/', admin.site.urls),
path("", mainView.index, name='index'),
path("workingProcess/", mainView.workingProcess, name='workingProcess'),
path("index/", mainView.index, name="index"),
path("logout/", mainView.logout, name="logout"),
path("UserLogin/", mainView.UserLogin, name="UserLogin"),
path("AdminLogin/", mainView.AdminLogin, name="AdminLogin"),
path("UserRegister/", mainView.UserRegister, name="UserRegister"),

### User Side Views
path("UserRegisterActions/", usr.UserRegisterActions,
name="UserRegisterActions"),
path("UserLoginCheck/", usr.UserLoginCheck, name="UserLoginCheck"),
path("UserHome/", usr.UserHome, name="UserHome"),

```

```

path("TextToSpeechForm/", usr.TextToSpeechForm, name='TextToSpeechForm'),
path("GenerateSpeaksforUser/", usr.GenerateSpeaksforUser,
name="GenerateSpeaksforUser"),
path("UserSpeechData/", usr.UserSpeechData, name='UserSpeechData'),
path("UserTimeDifference/", usr.UserTimeDifference,
name="UserTimeDifference"),

#### Admin Side Views
path("AdminLoginCheck/", admins.AdminLoginCheck, name="AdminLoginCheck"),
path("AdminHome/", admins.AdminHome, name="AdminHome"),
path("ViewRegisteredUsers/", admins.ViewRegisteredUsers,
name="ViewRegisteredUsers"),
path("AdminActivaUsers/", admins.AdminActivaUsers, name="AdminActivaUsers"),
path("AdminViewSpeechData/", admins.AdminViewSpeechData,
name="AdminViewSpeechData"),
path("AdminViewTimeDifference/", admins.AdminViewTimeDifference,
name="AdminViewTimeDifference"),

```

]

Base.html

```

{%load static%}
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Reveal Bootstrap Template</title>
<meta content="width=device-width, initial-scale=1.0" name="viewport">
<meta content="" name="keywords">
<meta content="" name="description">

<!-- Favicons -->
<link href="{%static 'img/favicon.png'%}" rel="icon">
<link href="{%static 'img/apple-touch-icon.png'%}" rel="apple-touch-icon">

<!-- Google Fonts -->
<link
href="https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,400i,700,
700i|Raleway:300,400,500,700,800|Montserrat:300,400,700" rel="stylesheet">

<!-- Bootstrap CSS File -->
<link href="{%static 'lib/bootstrap/css/bootstrap.min.css'%}"
rel="stylesheet">

<!-- Libraries CSS Files -->
<link href="{%static 'lib/font-awesome/css/font-awesome.min.css'%}"
rel="stylesheet">
<link href="{%static 'lib/animate/animate.min.css'%}" rel="stylesheet">

```

```
<link href="{%static 'lib/ionicons/css/ionicons.min.css'%}" rel="stylesheet">
<link href="{%static 'lib/owlcarousel/assets/owl.carousel.min.css'%}"
rel="stylesheet">
<link href="{%static 'lib/magnific-popup/magnific-popup.css'%}"
rel="stylesheet">
<link href="{%static 'lib/ionicons/css/ionicons.min.css'%}" rel="stylesheet">

<!-- Main Stylesheet File -->
<link href="{%static 'css/style.css'%}" rel="stylesheet">

</head>

<body id="body">
```

CHAPTER - 6

TESTING

6.1 ABOUT TESTING:

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

Field testing will be performed manually and functional tests will be written in detail.

6.2 TYPES OF TESTS:

Unit Testing:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration Testing:

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully.

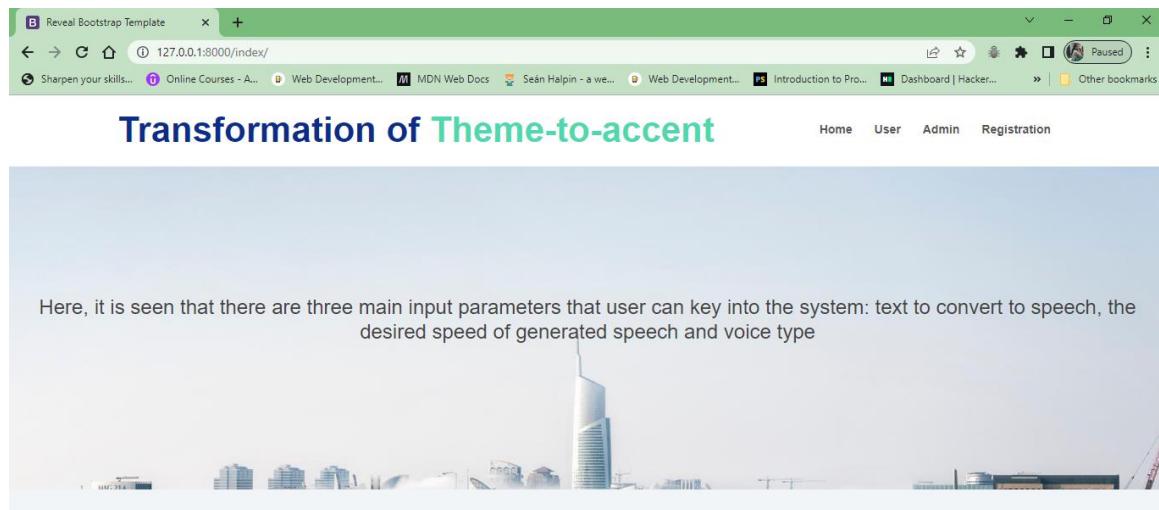
6.3 TEST CASES

Test Case Id	Test Case Name	Test Case description	Test Steps		Test case status
			Expected	Actual	
01	Registration	Registration successfully completed	Registration successful	Registration successful	pass
02	Login	Enter valid username and password	Login Successful	Login Successful	pass
03	Invalid Login	Enter invalid username and password	Login Unsuccessful	Login Unsuccessful	Pass
03	Admin Login	Enter valid username and password	Login successful	Login successful	pass
04	Admin can activate the register users	Admin activates user account successfully	Activated Successfully	Activated successfully	pass
05	Start Data preprocessing	Post input data	Post input data	Post input data	pass
06	Play the audio	play the audio file	Listen to the result	Listen to the result	pass
07	View All speech Data	View all mp3 files	View the data files	View the data files	pass

Tab 6.1 Test cases

CHAPTER - 7

SAMPLE SCREENS



Here, it is seen that there are three main input parameters that user can key into the system: text to convert to speech, the desired speed of generated speech and voice type



Fig 7.1 Home Page

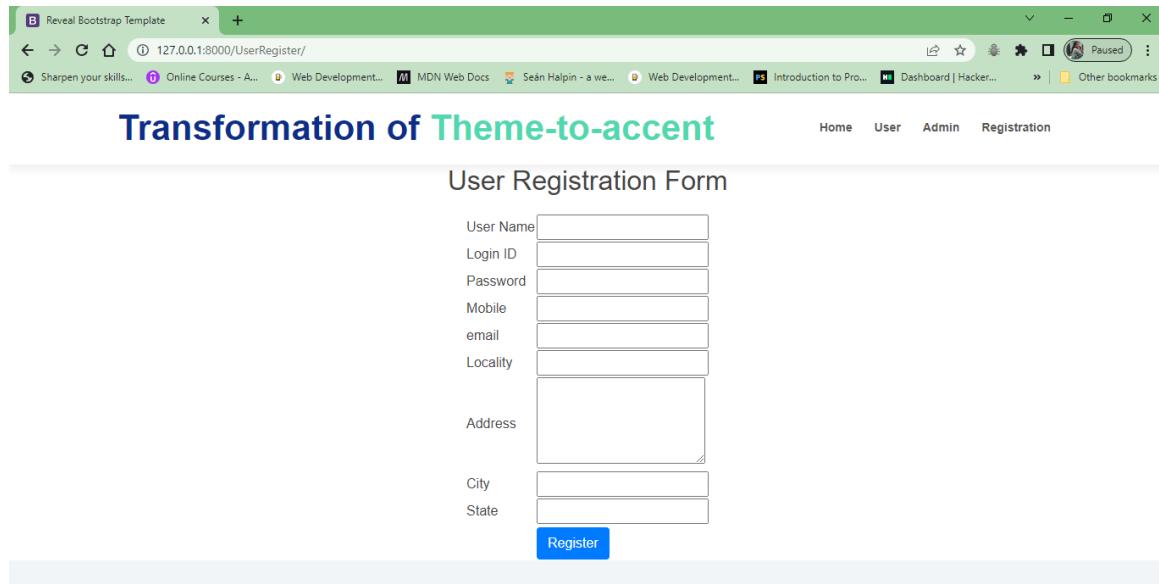


Fig 7.2 User Register Form

THEME TO ACCENT TRANSFORMATION APPLICATION

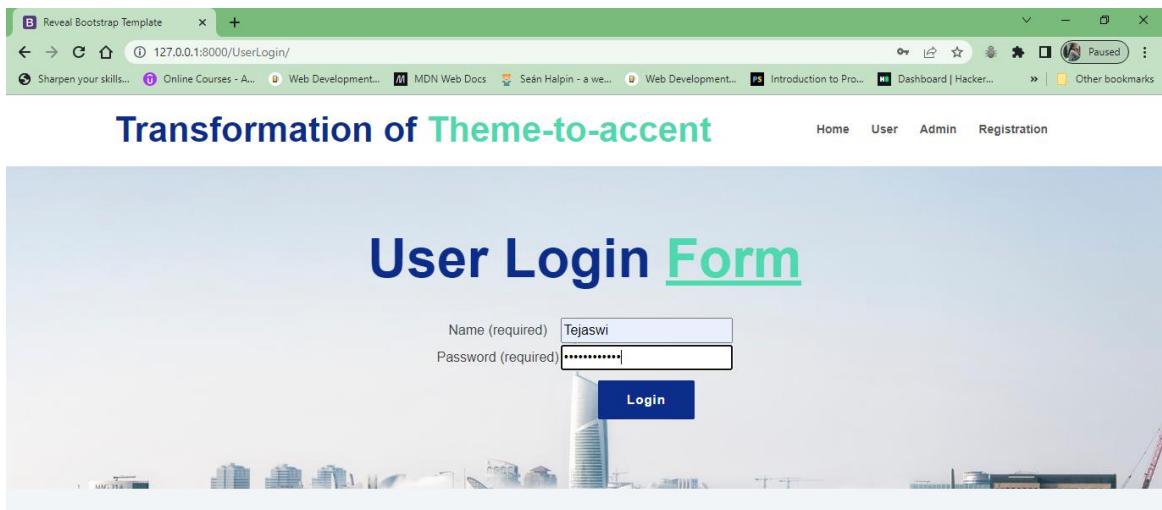


Fig 7.3 User Login Form

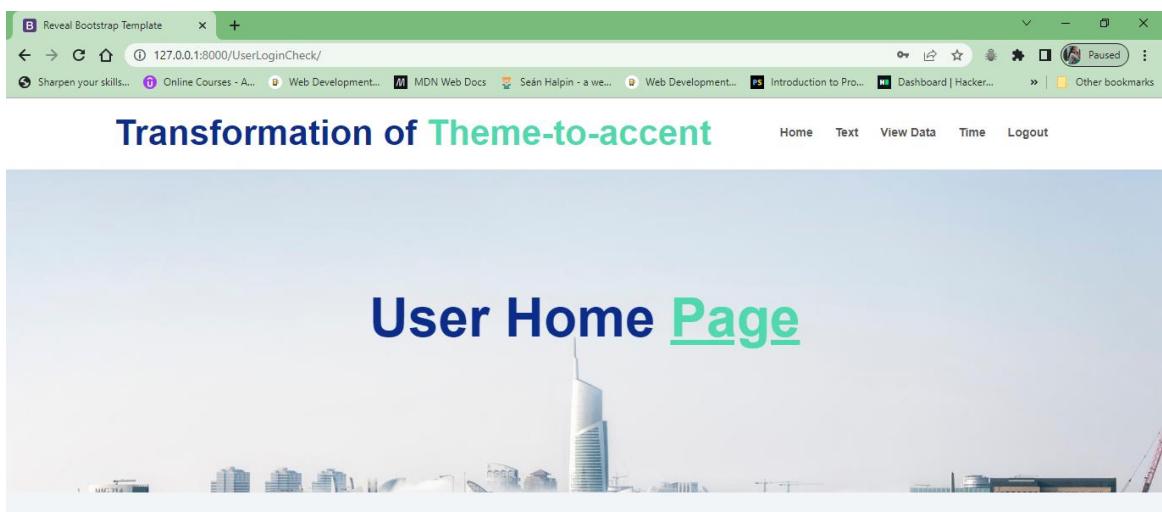


Fig 7.4 User Home Page

THEME TO ACCENT TRANSFORMATION APPLICATION

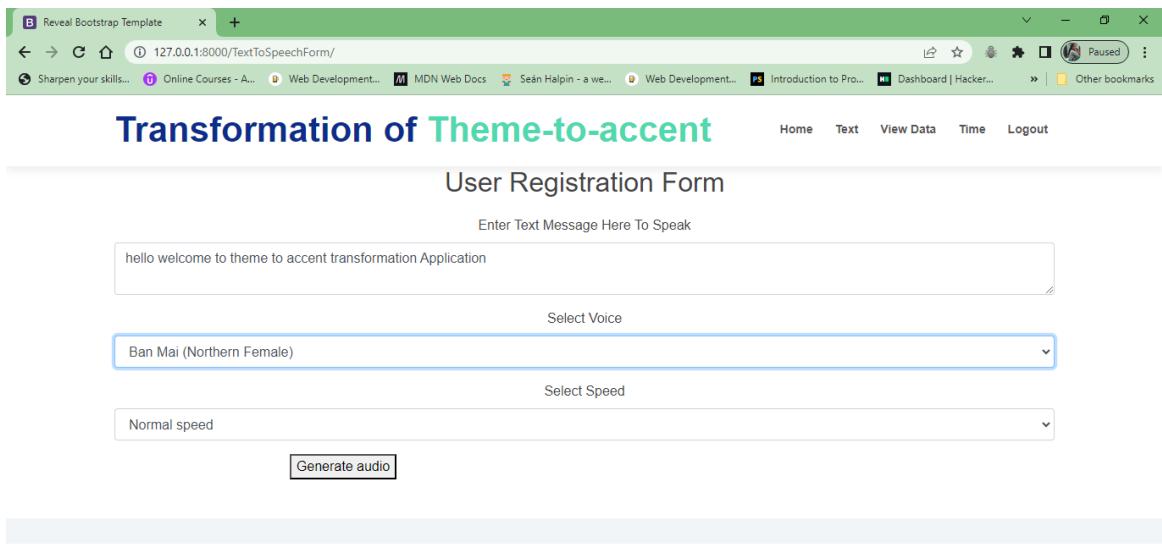


Fig 7.5 Text Input Form

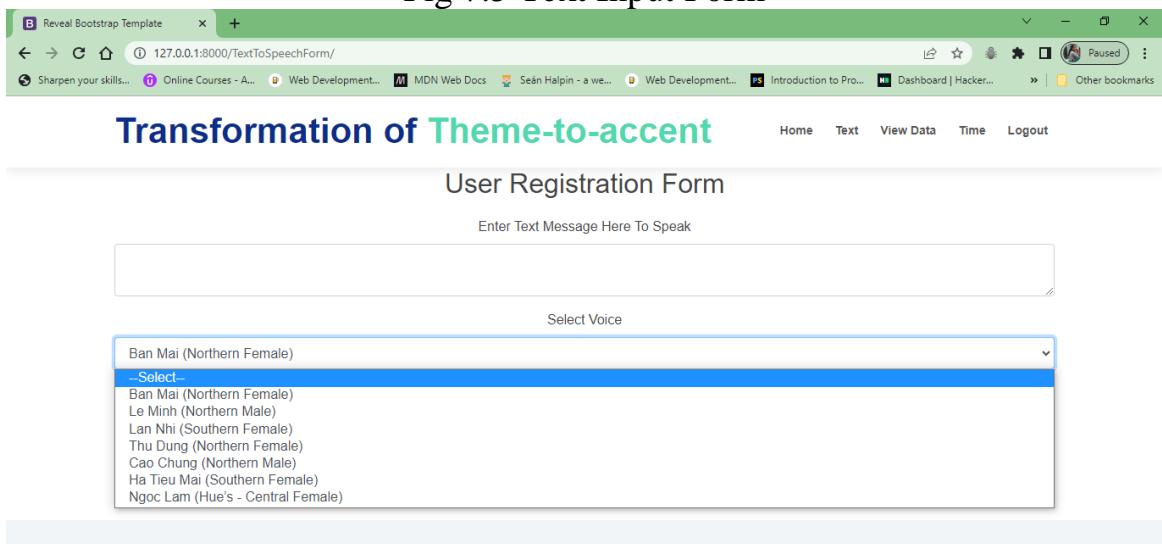


Fig 7.6 Our Voices

THEME TO ACCENT TRANSFORMATION APPLICATION

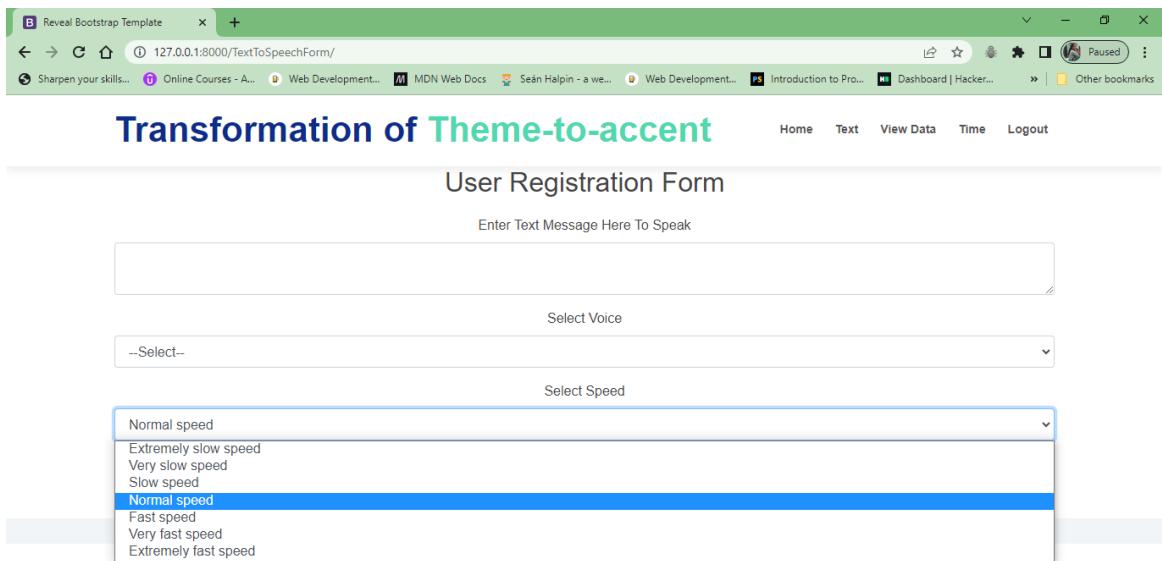


Fig 7.7 Speed Level

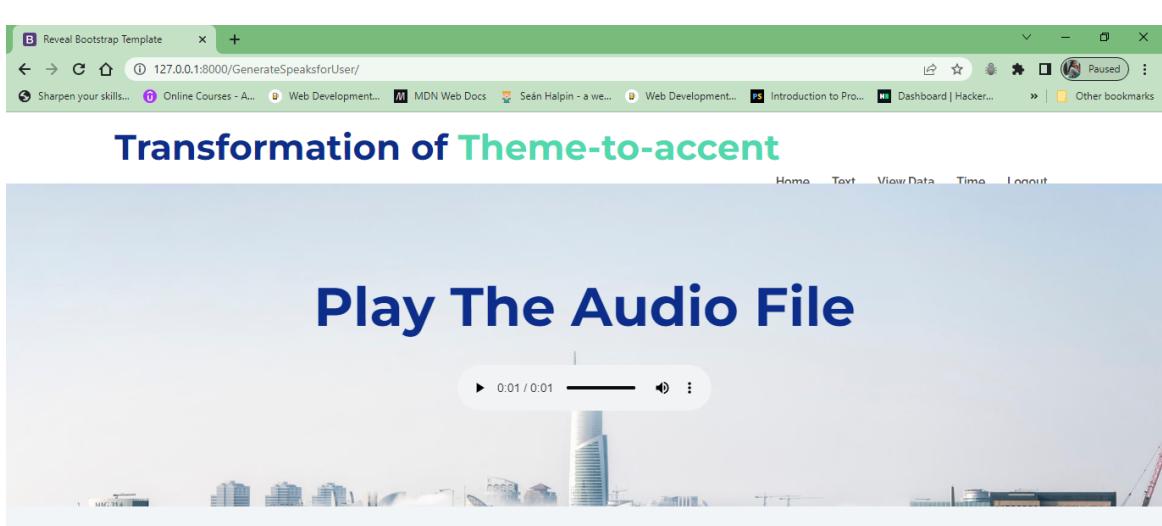


Fig 7.8 TTA Result

THEME TO ACCENT TRANSFORMATION APPLICATION

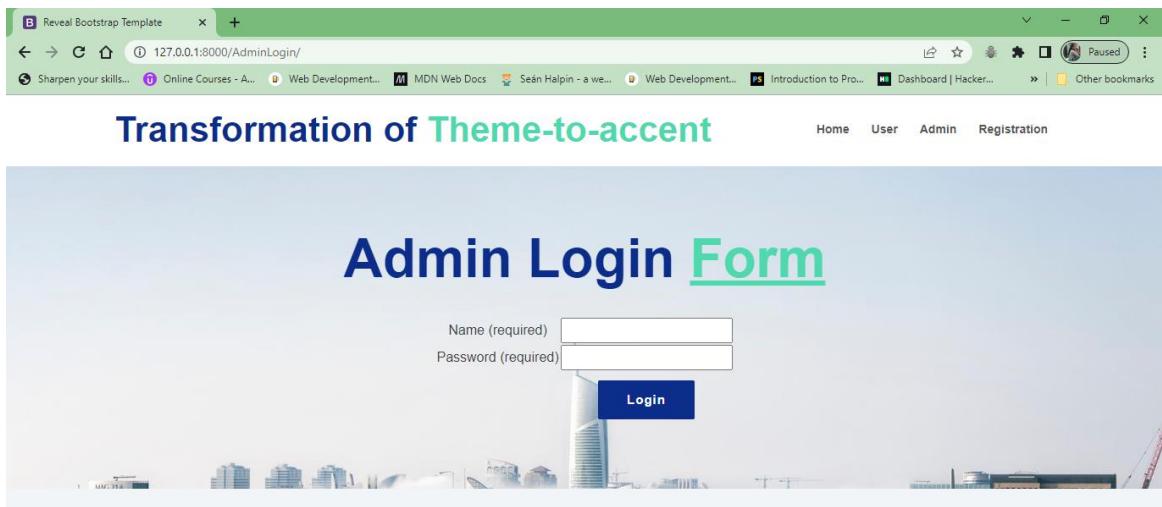


Fig 7.9 Admin Login Form

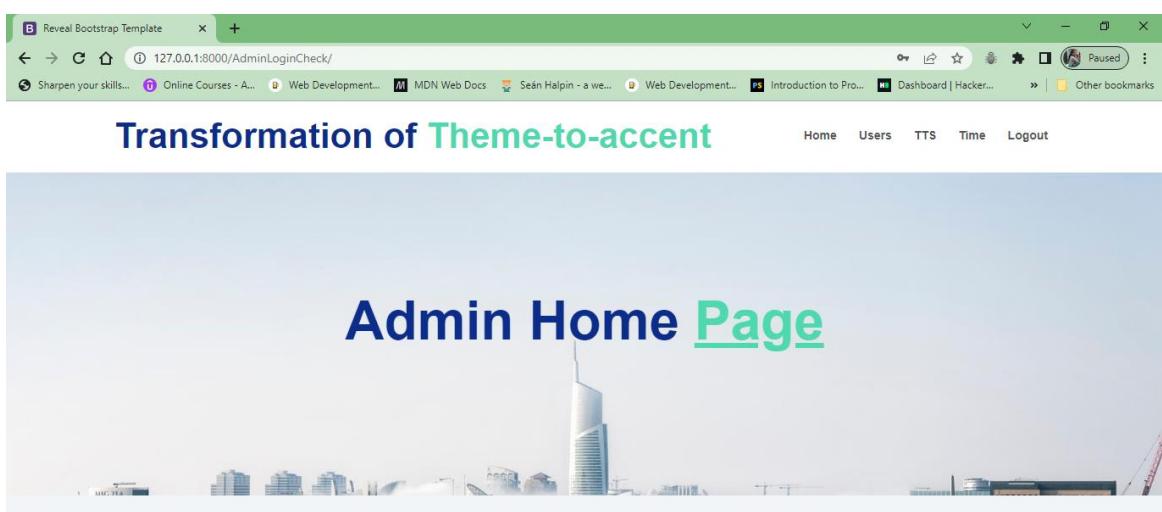


Fig 7.10 Admin Home Page

THEME TO ACCENT TRANSFORMATION APPLICATION

The screenshot shows a web browser window with a title bar "Reveal Bootstrap Template". The address bar displays "127.0.0.1:8000/ViewRegisteredUsers/". The page content is titled "Transformation of Theme-to-accent" and includes a sub-header "View User Registration Activate them". Below this is a table with the following data:

S.No	Name	Login ID	Mobile	Email	Locality	Status	Activate
1	alex	alex	9849098490	lx160cm@gmail.com	Hyderabad	activated	Activated
2	Sagar	sagar	9705858125	sagarmarri121@gmail.com	Godavarikhani	activated	Activated
3	Meghana	meghana	9849056568	meghanaarumalla@gmail.com	Vijayawada	waiting	Activate
4	Harish	harish	9849256683	gangishettyharish@gmail.com	Markapuram	activated	Activated
5	Tejaswi	Tejaswi	9390468496	21h41f0042tejaswi@gmail.com	Ambajipeta	activated	Activated



Fig 7.11 Activate Users

CHAPTER – 8

CONCLUSION

This work has presented a development framework for processing text to generate different male and female voices for Vietnamese. Based on the results obtained, it is seen that as input text length increases, its end-to-end conversion time to obtain the converted speech also increases accordingly. It is important to note that TTS application can help people in various working and entertainment environment, especially with those having difficulties in communication. In future, we will further improve the developed application to provide more attractive graphical user interface, and improve its voice's quality.

CHAPTER - 9

FUTURE ENHANCEMENTS

In the future, if speech recognition techniques reach an adequate level: synthesized speech may also be used in language interpreters or several other communication systems, such as videophones, videoconferencing or talking mobile phones. If it is possible to recognize speech, transcribe it into ASCII string and then resynthesize it back to speech a large amount of transmission capacity may be saved. With talking mobile phones, it is possible to increase the usability considerably for example with visually impaired users or in situations where it is difficult or even dangerous to try to reach the visual information. It is obvious that it is less dangerous to listen than to read the output from mobile phone for example when driving a car. During the last few decades the communication aids have been developed from talking calculators to modern three-dimensional audiovisual applications. The application field for speech synthesis is becoming wider all the time which brings more funds into research and development areas.

CHAPTER - 10

BIBLIOGRAPHY

Books Referred :

- Object Oriented Software Engineering: using UML, patterns and java, brooch
- The UML user guide – Grady Brooch
- Fundamentals of Software Engineering – Rajib Mali(PHI), Second Edition
- Think Python-Alen Downey, Green Tea Press.
- Python Programming- W.Chun, Pearson.
- Machine Learning in Action-Peter Harrington

Websites Referred:

- https://www.tutorialspoint.com/machine_learning_with_python/index.htm
- <https://pythonprogramming.net/basic-gui-pyqt-tutorial/>
- <https://www.geeksforgeeks.org/decision-tree-introduction-example/>

REFERENCES:

- [1] B. Liu et al., “Content-Oriented User Modeling for Personalized Response Ranking in Chatbots,” IEEE/ACM Trans. Audio Speech Lang. Process., 2018, doi: 10.1109/TASLP.2017.2763243.
- [2] H. Cuayáhuitl et al., “Ensemble-based deep reinforcement learning for chatbots,” Neurocomputing, 2019, doi: 10.1016/j.neucom.2019.08.007.
- [3] S. Arsovski, H. Osipyan, M. I. Oladele, and A. D. Cheok, “Automatic knowledge extraction of any Chatbot from conversation,” Expert Syst. Appl., 2019.
- [4] D. C. Tran, H. S. Ha, and A. Khalyasmaa, “A Question Detection Algorithm for Text Analysis,” in 2020 5th International Conference on Intelligent Information Technology (ICIIT), 2020, pp. 1–6.
- [5] F. Eyben et al., “Unsupervised clustering of emotion and voice styles for expressive TTS,” in ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, 2012, doi: 10.1109/ICASSP.2012.6288797.
- [6] W. Ping et al., “Deep Voice 3: 2000-Speaker Neural Text-to-Speech,” in Proc. ICLR, 2018.