MAJOR REPORT
ON
Text to speech and audio cloning by:

Muskan Sawa	500067886	R164218048
Dhairya Kumar Sharma	500067932	R164218023
Akash Raj	500068004	R164218007

Under the guidance of

DR. Ravi Tomar
Associate Professor
Department of informatics
School of Computer Science

School of Computer Science and Engineering

UNIVERSITY OF PETROLEUM AND ENERGY STUDIES

CANDIDATE’S DECLARATION

We hereby certify that the project work, entitled **Text to speech and voice cloning** in complete fulfilment of the requirements for the award of the Degree of **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE AND ENGINEERING** with specialization in “**Internet of Things and Smart Cities”** and submitted to the School of Computer Science, Department of Systemics, University of Petroleum & Energy Studies, Dehradun, is an authentic record of our work carried out during a period from **AUG-2021** to **DEC-2021** under the supervision of **Dr. Ravi Tomar, Associate Professor, Department of Informatics.**

Muskan Sawa	R164218048
Dhairya Kumar Sharma	R164218023
Akash Raj	R164218007

The matter presented in this project has not been submitted by us for the award of any other degree of this or any other University.

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Dr. Ravi Tomar
Project Guide
UPES, Dehradun

Dr. Neelu Jyoti Ahuja
HoD – Systemics
UPES, Dehradun

ACKNOWLEDGEMENT

We would like to express our gratitude to our mentor, **Dr. Ravi Tomar,** as well as our Activity Coordinator, **Ms. Deepa Joshi,** who gave us the opportunity to do this project. Our project, **Text to speech and voice cloning** would not have been possible without their support and valuable suggestions.

We sincerely thank our respected Head of the Department, **Dr. Neelu Jyoti Ahuja**, for her support in doing our project at the School **of Computer Science**.

We would like to thank all our **friends** for their help and constructive criticism during our project work. We are also grateful to our **parents** who have shown us this world and for every support they have given us.

Finally, we have no words to express our sincere gratitude to our brave **DOCTORS AND NURSES** who are working day and night in this pandemic to save lives, they are the soldiers of humanity and we are proud to say that this project is our small contribution for the same cause.

TABLE OF CONTENTS

Introduction

Allowing people to converse with machines is a long-standing dream of human-computer interaction. The ability of computers to understand natural speech has been revolutionised in the last few years by the application of deep neural networks (e.g., Google Voice Search). However, generating speech with computers — a process usually referred to as speech synthesis or text-to-speech (TTS) — is still largely based on so-called concatenative TTS, where a very large database of short speech fragments is recorded from a single speaker and then recombined to form complete utterances. This makes it difficult to modify the voice (for example switching to a different speaker, or altering the emphasis or emotion of their speech) without recording a whole new database.

This has led to a great demand for parametric TTS, where all the information required to generate the data is stored in the parameters of the model, and the contents and characteristics of the speech can be controlled via the inputs to the model. So far, however, parametric TTS has tended to sound less natural than concatenative. Existing parametric models typically generate audio signals by passing their outputs through signal processing algorithms known as vocoders.

Objectives:

- Text to speech using different Mel models and vocoders
- Web Application connected with a database to display and save outputs properly
- Real time audio cloning

Literature Review:

- Over time, different techniques have dominated the field. Concatenative & Statistical parametric speech synthesis. However, the audio produced by these systems often sounds muffled and unnatural compared to human speech. Here comes the Tacotron 2 which is introduced as a completely neural TTS framework that consolidates a sequence-to-sequence recurrent network with thoughtfulness regarding predicted Mel spectrograms with an adjusted WaveNet vocoder. The subsequent framework incorporates discourse with Tacotron-level prosody and WaveNet-level sound quality. This framework has been prepared straightforwardly from information without depending on complex component designing, and accomplishes best in class sound quality near that of regular human discourse [1]. Tacotron produces Mel-spectrogram which is passed through a wavenet vocoder MelGAN, a non-autoregressive feed-forward convolutional architecture to perform audio waveform generation in a GAN setup. We come to know that we can successfully train GANs for raw audio generation without additional distillation or perceptual loss functions while still yielding a high-quality text-to-speech synthesis mode [2]. The introduction to use the multi-band MelGAN, a much faster waveform generation model targeting high-quality text-to-speech as compared to melGAN has been clarified here. The improvement associated with mb-MelGAN is because of two reasons-1. increase the receptive field of the generator, 2. Substituting the feature matching loss with the multi-resolution STFT loss to better measure the difference between fake and real speech [3]. Also, keeping in mind that every technology comes with its own advantage as well as disadvantage and in the case of tacotron we have seen some of its limitations i.e. suffering from slow inference speed, and the synthesized speech is usually not robust (i.e., some words are skipped or repeated) and lack of controllability (voice speed or prosody control).So here comes the introduction of Fastspeech which is a novel feed-forward network based on Transformer to generate Mel-spectrogram in parallel for TTS, in this it extricate consideration arrangements from an encoder-decoder based instructor model for phoneme span expectation, which is utilized by a length regulator to extend the source phoneme succession to coordinate with the length of the objective Mel-spectrogram grouping for parallel Mel-spectrogram age. Basically, Fastspeech speeds up Mel-spectrogram generation by 270x and the end-to-end speech synthesis by 38x. [4]. Another main thing about Fastspeech is that it also suffers from some limitations such as 1) the teacher-student refining pipeline is convoluted and tedious, 2) the length extricated from the teacher model isn't sufficiently precise, and the objective Mel-spectrograms refined from teacher model experience the ill effects of data misfortune because of information disentanglement, the two of which limit the voice quality. So, there is need to tackle all this problem and here comes Fastspeech 2 which addresses the issues in FastSpeech and better solves the one-to-many mapping problem in TTS by directly train the model with ground-reality of data instead of teacher’s simplified output and also giving more variation information of speech like pitch, energy and more accurate duration of words. It is shown that FastSpeech 2 achieves a 3x training speed-up over FastSpeech. [5]

SYSTEM REQUIREMENTS:

- Python 3.7+
- Cuda 10.1
- CuDNN 7.6.5
- Tensorflow 2.2/2.3/2.4/2.5/2.6
- Tensorflow Addons >= 0.10.0
- MySQL Workbench
- Pycharm
- Html
- CSS

Libraries imported:

- import Flask, render_template, redirect url_for, request
    - Flask is a lightweight Python web framework that provides useful tools and features that make creating web applications in Python easier
    - render_template is used to generate output from a template file
    - url_for () redirects based on the string representation of a route; you provide the function name of the route you want to redirect to.
    - The request object holds all incoming data from the request, which includes the mimetype, referrer, IP address, raw data, HTTP method, and headers, among other things.
- flask_mysqldb
    - Flask-MySQLdb provides a MySQL connection for Flask.
- import MySQLdb
    - MySQLdb is used to import the required python module. MySQLdb. connect () method takes hostname, username, password, and database schema name to create a database connection. On successfully connecting with the database, it will return a connection object
- import TensorFlow
    - It is an open-source artificial intelligence library, using data flow graphs to build models. It allows developers to create large-scale neural networks with many layers. TensorFlow is mainly used for: Classification, Perception, Understanding, Discovering, Prediction and Creation.
- import yaml
    - YAML is a data serialization language that is often used for writing configuration files. Depending on whom you ask, YAML stands for yet another markup language or YAML ain't markup language (a recursive acronym), which emphasizes that YAML is for data, not documents.
- import numpy
    - NumPy is a Python library used for working with arrays. It also has functions for working in the domain of linear algebra, fourier transform, and matrices.
- import matplotlib.pyplot
    - Matplotlib is a plotting library for creating static, animated, and interactive visualizations in Python. Matplotlib can be used in Python scripts, the Python and IPython shell, web application servers, and various graphical user interface toolkits like Tkinter, etc.

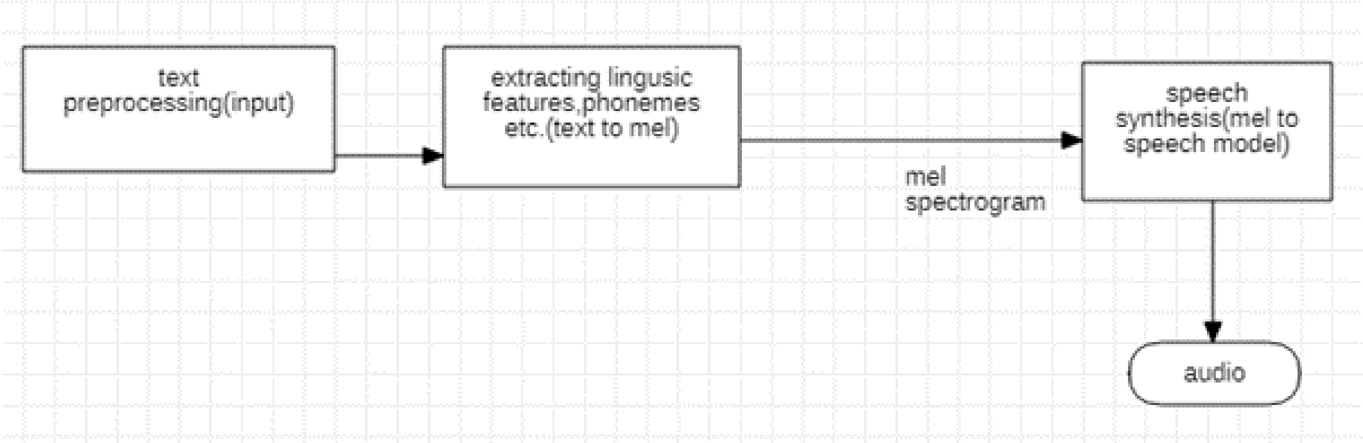
- import soundfile
    - SoundFile is an audio library based on libsndfile. We used “sf.write() function to save the audio files in our program”
- tensorflow_tts.inference import TFAutoModel
    - TFAutoModel is used to load Pretrained models. it is a generic tokenizer class that will be instantiated as one of the tokenizer classes of the library when created with the AutoTokenizer
- tensorflow_tts.inference import AutoConfig
    - AutoConfig is a tool that simplifies and standardizes configuration management tasks in an Oracle E-Business Suite environment
- tensorflow_tts.inference import AutoProcessor
    - It is use to convert text to sequence as an input for the Text to mel models
- synthesizer.inference import Synthesizer
    - This component is the core model of Text-to-Speech Synthesis. It takes in the sequence of phonemes as inputs and generates a spectrogram of the corresponding text input. Phonemes are distinct units of a sound of words. Each word is decomposed into these phonemes and sequence input to the model is formed.
- encoder import inference as encoder
    - The speaker encoder network's job is to take audio of a given speaker as input, and encode the characteristics of their voice into a low dimensional vector embedding.
- vocoder import inference as vocoder
    - It takes the Mel spectrograms generated by the synthesis network as input and autoregressively generate the time-domain audio waveforms as output.
- pathlib import Path
    - The Pathlib module in Python deals with path related tasks, such as constructing new paths from names of files and from other paths, checking for various properties of paths and creating files and folders at specific paths.
- import librosa
    - Librosa is a Python package for music and audio analysis. Librosa is basically used when we work with audio data like in music generation(using LSTMs), Automatic Speech Recognition. It provides the building blocks necessary to create the music information retrieval systems.

METHODOLOGY AND WORKINGS:

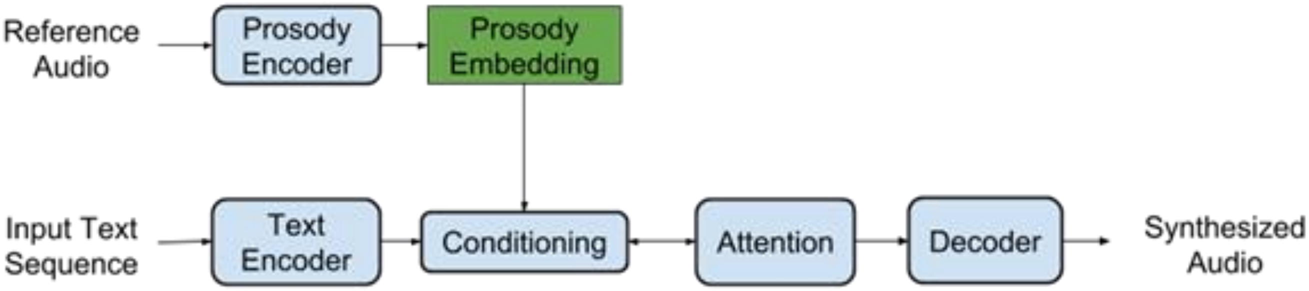
- ## Text to Speech Web Application:
  - The Web Application has 4 inputs and a submit button on the html form and the respective inputs are: 1) The input text (which is to be converted to speech). 2) The Text to Mel model. 3) The Vocoder model. 4) File name (by which the audio file and spectrogram image is saved)
  - The html form has method = “post” so once the input is verified, the main program connects with the database and checks for any ambiguity.
  - If no ambiguity is found, it then generates audio file and spectrogram image based on the provided input and saves them in “static/output/audio” and “static/output/graph” respectively
  - Once the image file and audio file are generated without any issues, it then records the input text and file name along with the Text to mel model and vocoder in the database and refreshes the web page
  - Once the page refreshes, it connects with the database and verifies all the input data and then uploads the data in table format by checking the data in the database and output folders.

- ## Real Time Audio Cloning:
  - pretrained models for an encoder, synthesizer and a vocoder are loaded from the default folder in saved_models
  - A 40 second audio file in .wav format is used as input for the program.
  - A text input is then provided to the program which is to be converted to the required audio. Its path is saved using inpath variable and pathlib library.
  - This audio file is then passed into the encoder to generate low dimensional vector embeddings of the audio
  - After obtaining the sequence of phonemes from the encoder, the synthesizer is then used to generate a spectrogram of the corresponding text input based on the embedding provided by the encoder.
  - Once the spectrogram of the input text is obtained, the vocoder is used to generate waveform of the audio based on the input spectrogram
  - The waveform generated by the vocoder is then used to create an audio in .wav format.

Flow Chart

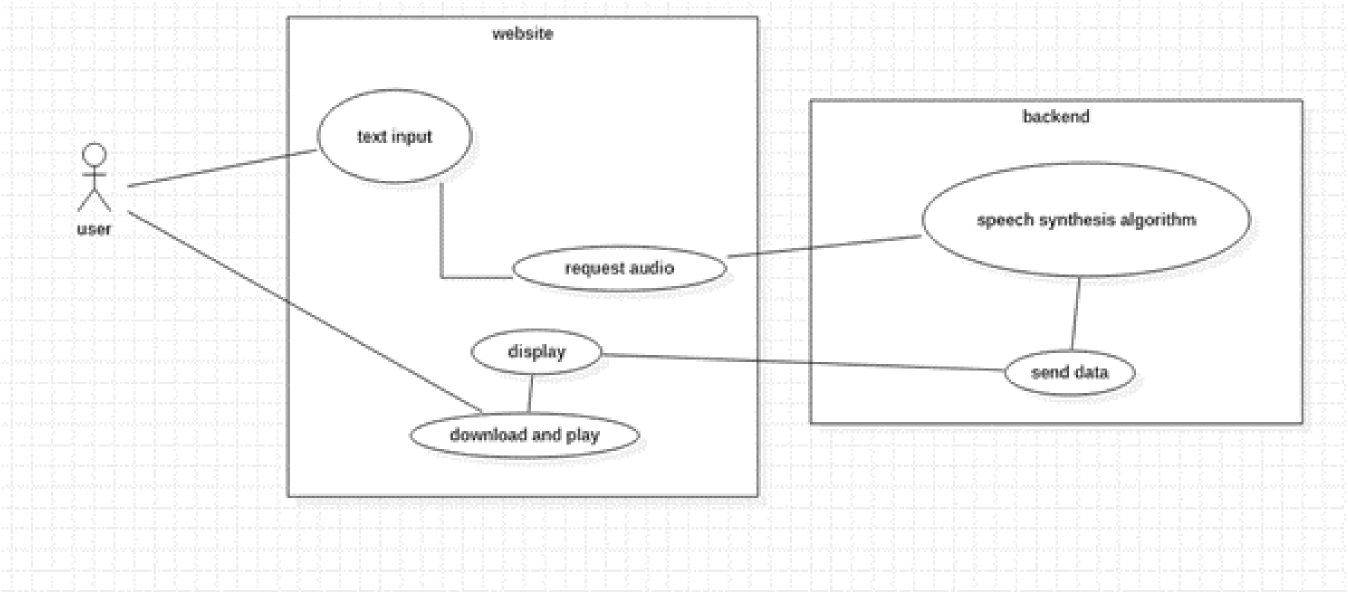

text preprocessing(input)
extracting lingusic features,phonemes etc.(text to mel)
mel spectrogram
speech synthesis(mel to speech model)
audio

Fig(a) IPC for Text to speech synthesis



Fig(b) Flow chart about the workings of encoder, synthesizer and decoder

Use Case Diagram

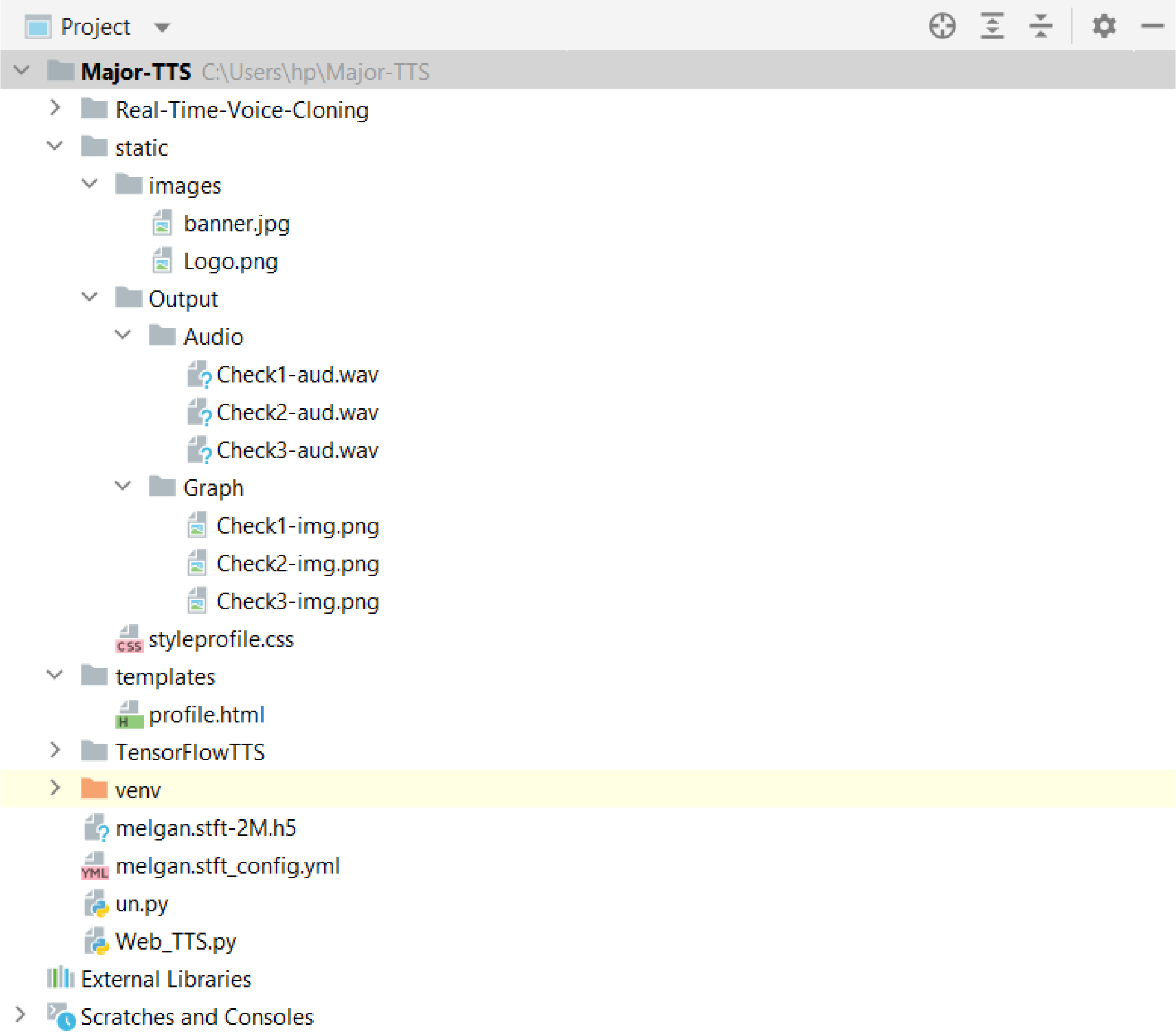


Fig(c) Use can diagram depicting the internal workings of the web application

CODE:

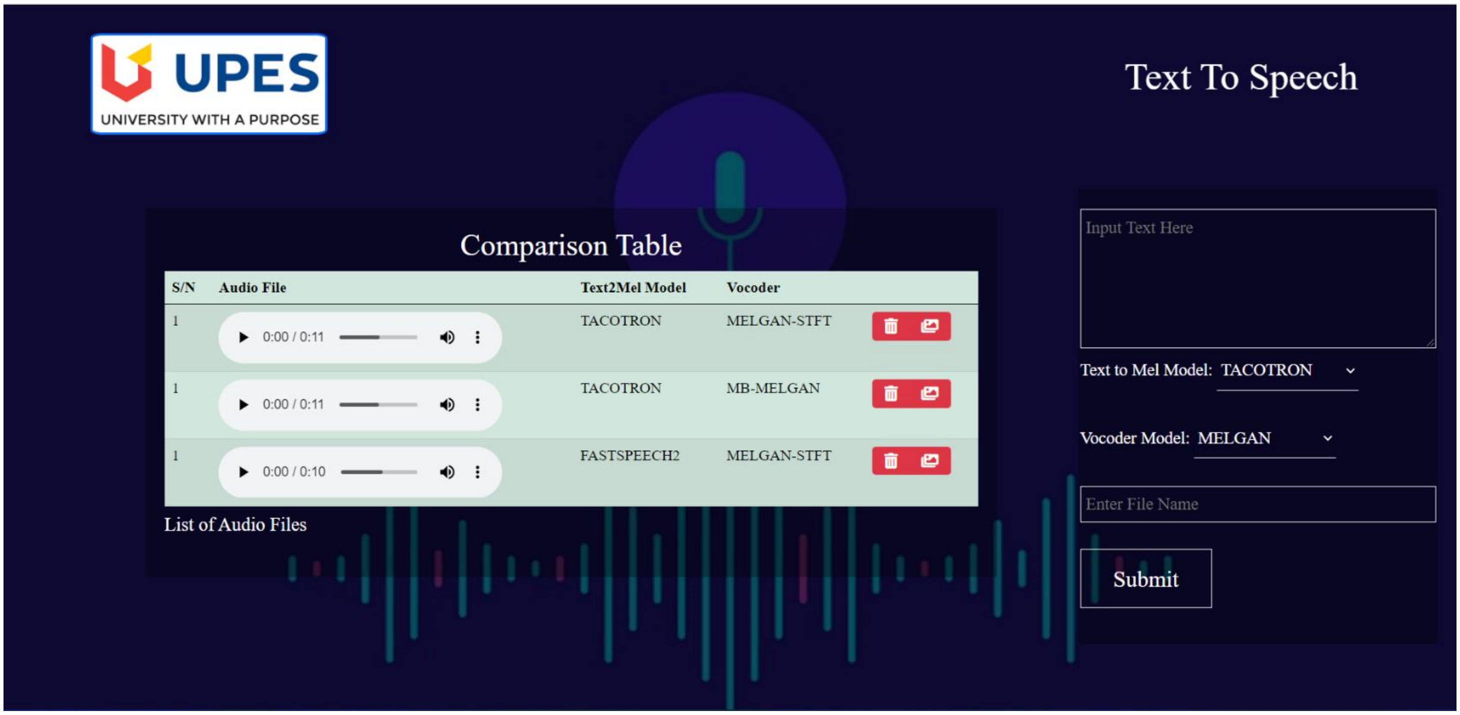
https://github.com/BlueDice37/Major-TTS

Project:



```
Major-TTS C:\Users\hp\Major-TTS
  Real-Time-Voice-Cloning
  static
    images
      banner.jpg
      Logo.png
    Output
      Audio
        Check1-aud.wav
        Check2-aud.wav
        Check3-aud.wav
      Graph
        Check1-img.png
        Check2-img.png
        Check3-img.png
    styleprofile.css
  templates
    profile.html
  TensorFlowTTS
  venv
  melgan.stft-2M.h5
  melgan.stft_config.yml
  un.py
  Web_TTS.py
External Libraries
Scratches and Consoles
```

Web Application:



Text To Speech

Comparison Table

S/N	Audio File	Text2Mel Model	Vocoder	
1	0:00 / 0:11	TACOTRON	MELGAN-STFT	
1	0:00 / 0:11	TACOTRON	MB-MELGAN	
1	0:00 / 0:10	FASTSPEECH2	MELGAN-STFT	

List of Audio Files

Text to Mel Model: TACOTRON

Vocoder Model: MELGAN

References:

[1] J. Shen and R. Pang, "Natural tts synthesis by conditioning wavenet on mel spectrogram predictions," in (ICASSP):(pp. 4779-4783)IEEE, Canada, 2018

[2] K. Kumar, "MelGAN: Generative Adversarial Networks for," in arXiv preprint:1910.06711, Mila, 2019.

[3] G. Yang, "Multi-band MelGAN: Faster Waveform Generation for High-Quality Text-to-Speech," in IEEE: Spoken Language Technology Workshop (SLT), China, 2020.

[4] Y. Ren, "FastSpeech: Fast, Robust and Controllable," in arXiv preprint:1905.09263, China, 2019.

[5] Y. Ren, "FastSpeech 2: Fast and High-Quality End-to-End Text to Speech," in arXiv preprint:2006.04558, China, 2020.

Report verified by:

Ravi Tomar **(Project Guide)**	**Dr. Neelu Jyoti Ahuja** **(Program Head)**