**Synthetic speech generation system**

This guide will take you through building a synthetic speech generation system, leveraging GitHub for collaboration, and deploying the final product live. We'll break the project into **three phases**, each lasting about a month. The project will include tools like **WaveNet** and **Tacotron 2**, along with GitHub best practices.

**Phase 1: Research, Setup, and Voice Model Integration (Week 1-4)**

**1. Project Setup & Repository Creation**

* **Create a GitHub Repository**: Name it something like ai-voiceover-generator.
* **Initial Branch Setup**: Set up main, development, and feature branches.
* **GitHub Skills**:
  + Write a basic README.md outlining the project’s purpose.
  + Create a .gitignore file to exclude unnecessary files like large audio datasets or temporary files.
  + Enable Issues and Pull Requests to manage features and bug tracking.

**Resources**:

* [GitHub for Beginners](https://docs.github.com/en/get-started)
* [GitHub Actions](https://docs.github.com/en/actions)

**2. Model Research and Selection**

* **Generative Speech Models**: Focus on **Tacotron 2** and **WaveNet**.
  + **Tacotron 2** for converting text to mel-spectrograms.
  + **WaveNet** for generating speech waveforms from spectrograms.
* Review pretrained models to save time on training from scratch.

**Resources**:

* [Tacotron 2 Research Paper](https://arxiv.org/abs/1712.05884)
* [WaveNet on GitHub](https://github.com/ibab/tensorflow-wavenet)

**3. Text Input and Preprocessing**

* Implement a basic Python script to accept user input text.
* Use **NLP preprocessing** to clean and format the text for TTS (Text-to-Speech).
* GitHub Action: Set up a pipeline to automatically lint your code (using flake8) and run tests (using pytest) after every push to the development branch.

**Resources**:

* NLP Preprocessing Guide

**4. Voice Customization Parameters**

* Research and define customization parameters for different **voice styles** (male, female, formal, casual) and **emotions** (happy, sad, neutral).
* Design a basic **UI mockup** using **Streamlit** or **Flask** where users can select voice type and emotion.

**Phase 2: Model Integration and Customization (Week 5-8)**

**1. Integrating Tacotron 2 for Text to Spectrogram Conversion**

* Implement **Tacotron 2** to convert input text to mel-spectrograms.
* Ensure the output spectrograms are of high quality and can later be used by the vocoder model for speech synthesis.

**GitHub Task**:

* Work on a branch like feature-tacotron2-integration.
* Regularly create pull requests to development for code review and testing.

**Resources**:

* [Tacotron 2 GitHub Implementation](https://github.com/Rayhane-mamah/Tacotron-2)

**2. WaveNet Integration for Speech Generation**

* Implement **WaveNet** to synthesize the final speech waveform from the mel-spectrogram.
* Ensure smooth transitions in speech and natural-sounding voices.

**Resources**:

* WaveNet Overview
* [WaveNet in TensorFlow](https://github.com/ibab/tensorflow-wavenet)

**3. Voice Style and Emotion Customization**

* Fine-tune Tacotron 2 or WaveNet to enable adjustments in tone, accent, and emotion.
* Allow users to switch between different voice styles and apply emotional tones.
* **GitHub Skills**: Create separate branches for each major feature (e.g., feature-voice-style, feature-emotion-customization).

**Resources**:

* [Deep Voice by Baidu](https://arxiv.org/pdf/1702.07825.pdf) for reference on emotional tones.
* Transfer Learning in TTS

**4. Audio Post-Processing**

* Use libraries like librosa or pydub to enhance the generated speech quality.
* Post-process the audio to ensure clarity, reduce noise, and improve voice quality.

**GitHub Actions**:

* Use continuous integration (CI) to automatically test audio post-processing features.
* Collaborate on feature-postprocessing branch, testing the sound quality.

**Resources**:

* [Librosa for Audio Processing](https://librosa.org/)
* [Pydub Documentation](https://pydub.com/)

**Phase 3: Real-Time Testing, Deployment & Polishing (Week 9-12)**

**1. Real-Time Text-to-Speech Testing**

* Test the full pipeline: text input ➝ Tacotron 2 (mel-spectrogram) ➝ WaveNet (speech generation).
* Optimize for performance, ensuring minimal lag between user input and speech generation.
* Implement stress tests to handle larger text inputs or real-time requests.

**GitHub Tasks**:

* Collaboratively test on a branch like feature-real-time-testing.
* Conduct A/B testing to collect user feedback and optimize voice styles and emotions.

**2. Deploy the Application**

* Use **Streamlit Cloud**, **AWS**, or **Heroku** for deployment.
* Deploy the model using a **Docker container** to ensure consistency across environments.
* **GitHub CI/CD Pipeline**: Set up automatic deployment using **GitHub Actions**. Upon merging code to the main branch, the app should be automatically deployed to your chosen platform.

**Resources**:

* Streamlit Deployment
* Docker for Beginners
* Heroku Python App Deployment

**3. Polishing the UI & Final Tests**

* Polish the user interface with better controls for voice style, emotion, and text input.
* Conduct final tests to ensure the quality of generated speech.
* Add error handling for unsupported characters or long text inputs.
* GitHub Actions: Set up notifications for code failures and user feedback integration.

**Resources**:

* Streamlit UI Design
* Improving Speech Synthesis

**4. Documentation & Final Deployment**

* Write final documentation on how to use the tool, contribute to the codebase, and troubleshoot issues.
* Deploy the final version and open it for public or controlled user access.
* Promote the tool by sharing it on relevant platforms (e.g., GitHub, Streamlit community).

**GitHub Task**:

* Finalize all documentation and close remaining issues before merging everything into the main branch.

**Key Tools & Resources**

* **Generative Models**:
  + [Tacotron 2](https://github.com/Rayhane-mamah/Tacotron-2)
  + [WaveNet](https://github.com/ibab/tensorflow-wavenet)
* **Audio Processing**:
  + [Librosa](https://librosa.org/)
  + [Pydub](https://pydub.com/)
* **Web App & Deployment**:
  + Streamlit
  + Heroku Deployment Guide
* **GitHub Skills**:
  + [GitHub Basics](https://docs.github.com/en/get-started)
  + [GitHub Actions for CI/CD](https://docs.github.com/en/actions)