

isper-ai-with-beautifull-web-ui-2

June 25, 2025

0.1 NOTE : TO RUN THE ALL CELLS AUTOMATICALLY USE : CTRL + F9

setup directory

```
[ ]: # Create project directories
!mkdir -p /content/whisper-app
!mkdir -p /content/whisper-app/outputs
!mkdir -p /content/whisper-app/static/css
!mkdir -p /content/whisper-app/static/js
!mkdir -p /content/whisper-app/templates
```

1 Install required packages

```
[ ]: # Install required packages
!pip install -q faster-whisper flask flask_cors pyngrok
```

```
1.1/1.1 MB
24.3 MB/s eta 0:00:00
35.3/35.3 MB
16.7 MB/s eta 0:00:00
38.6/38.6 MB
41.6 MB/s eta 0:00:00
16.4/16.4 MB
81.5 MB/s eta 0:00:00
46.0/46.0 kB
3.3 MB/s eta 0:00:00
86.8/86.8 kB
7.6 MB/s eta 0:00:00
```

1.1 App.py setup

```
[ ]: %%writefile /content/whisper-app/app.py
import os
import torch
import tempfile
```

```

import uuid
import json
import time
import sys
from datetime import timedelta
from flask import Flask, request, jsonify, send_file, render_template
from flask_cors import CORS
from pyngrok import ngrok

# Set up logging
import logging
logging.basicConfig(level=logging.INFO,
                    format='%(asctime)s - %(levelname)s - %(message)s',
                    stream=sys.stdout)
logger = logging.getLogger(__name__)

# Create output directory
output_dir = "outputs"
os.makedirs(output_dir, exist_ok=True)

# Initialize Flask App
app = Flask(__name__, static_folder='static')
CORS(app)

# Initialize Whisper Model in a separate function
def load_whisper_model():
    try:
        logger.info("Starting to load Whisper model...")
        # Import here to avoid slowing down app startup
        from faster_whisper import WhisperModel

        # You can change this to "medium" or "small" for faster loading but
        ↪ less accuracy
        model_size = "large-v3"
        device = "cuda" if torch.cuda.is_available() else "cpu"

        logger.info(f"Using device: {device}")
        logger.info(f>Loading model size: {model_size}")

        # Show progress indicator
        for i in range(5):
            sys.stdout.write(".")
            sys.stdout.flush()
            time.sleep(0.5)

        start_time = time.time()
        model = WhisperModel(

```

```

        model_size,
        device=device,
        compute_type="float16",
        download_root="./whisper_models"
    )

    elapsed_time = time.time() - start_time
    logger.info(f"Model loaded successfully on {device} in {elapsed_time:.
↪2f} seconds!")
    return model
except Exception as e:
    logger.error(f"Error loading Whisper model: {e}")
    raise

# SRT Format Generation Function
def format_srt(segments):
    srt_output = []
    for i, segment in enumerate(segments, start=1):
        start = timedelta(seconds=segment.start)
        end = timedelta(seconds=segment.end)
        text = segment.text
        start_time = f"{start.seconds//3600:02}:{(start.seconds//60)%60:02}:
↪{start.seconds%60:02},{int(start.microseconds/1000):03}"
        end_time = f"{end.seconds//3600:02}:{(end.seconds//60)%60:02}:{end.
↪seconds%60:02},{int(end.microseconds/1000):03}"
        srt_output.append(f"{i}\n{start_time} --> {end_time}\n{text}\n")
    return "\n".join(srt_output)

# Global variable for the model
whisper_model = None

# Process Audio Function
def process_audio(audio_path, task, language):
    global whisper_model

    # Lazy-load the model if it's not already loaded
    if whisper_model is None:
        whisper_model = load_whisper_model()

    logger.info(f"Processing audio: {audio_path}")
    logger.info(f"Task: {task}, Language: {language}")

    segments, info = whisper_model.transcribe(
        audio_path,
        task=task,
        language=language if language != 'auto' else None,
        beam_size=5,

```

```

        vad_filter=True,
        word_timestamps=False
    )

    logger.info(f"Detected language: {info.language} (confidence: {info.
↪language_probability*100:.1f}%)")
    return list(segments), info

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/transcribe', methods=['POST'])
def transcribe_audio():
    if 'file' not in request.files:
        return jsonify({"error": "No file part"}), 400
    file = request.files['file']
    if file.filename == '':
        return jsonify({"error": "No selected file"}), 400
    try:
        language = request.form.get('language', 'auto')
        translate = request.form.get('translate') == 'on'
        output_format = request.form.get('format', 'srt')

        logger.info(f"Received file: {file.filename}")
        logger.info(f"Language: {language}, Translate: {translate}, Format: ↪
↪{output_format}")

        temp_dir = tempfile.mkdtemp()
        audio_path = os.path.join(temp_dir, file.filename)
        file.save(audio_path)

        logger.info(f"File saved to: {audio_path}")

        task = 'translate' if translate else 'transcribe'
        segments, info = process_audio(audio_path, task, language)

        output_filename = f"{uuid.uuid4()}.{output_format}"
        output_path = os.path.join(output_dir, output_filename)

        logger.info(f"Writing output to: {output_path}")

        if output_format == 'srt':
            with open(output_path, 'w', encoding='utf-8') as f:
                f.write(format_srt(segments))
        elif output_format == 'txt':
            with open(output_path, 'w', encoding='utf-8') as f:

```

```

        f.write(''.join([segment.text for segment in segments]))
    elif output_format == 'json':
        json_data = [{"start": segment.start, "end": segment.end, "text":
↪segment.text} for segment in segments]
        with open(output_path, 'w', encoding='utf-8') as f:
            json.dump(json_data, f, ensure_ascii=False, indent=2)

    # Clean up
    os.remove(audio_path)
    os.rmdir(temp_dir)

    response = {
        "original": output_filename if task == 'transcribe' else None,
        "translation": output_filename if task == 'translate' else None,
        "language": info.language,
        "detected_language_confidence": info.language_probability * 100
    }

    logger.info("Transcription completed successfully")
    return jsonify(response), 200
except Exception as e:
    logger.error(f"Error in transcribe_audio: {e}", exc_info=True)
    return jsonify({"error": str(e)}), 500

@app.route('/download/<filename>', methods=['GET'])
def download_file(filename):
    try:
        file_path = os.path.join(output_dir, filename)
        logger.info(f"Downloading file: {file_path}")
        return send_file(file_path, as_attachment=True)
    except Exception as e:
        logger.error(f"Error in download_file: {e}")
        return jsonify({"error": str(e)}), 404

@app.route('/health', methods=['GET'])
def health_check():
    return jsonify({"status": "ok"}), 200

# This will be called from the main script
def start_app(ngrok_token=None):
    try:
        # Set ngrok auth token if provided
        if ngrok_token and ngrok_token != "YOUR_NGROK_AUTH_TOKEN":
            try:
                ngrok.set_auth_token(ngrok_token)
                logger.info("Ngrok auth token set successfully")
            except Exception as e:

```

```

        logger.error(f"Failed to set ngrok auth token: {e}")
        # Continue anyway, as it might be using the default token

    # Start ngrok tunnel
    public_url = ngrok.connect(5000)
    logger.info(f" Whisper AI Web UI is running at: {public_url}")
    print(f"\n\n Whisper AI Web UI is running at: {public_url}")
    print("Open this URL in your browser to access the application\n\n")

    # Run the Flask app
    app.run(host='0.0.0.0', port=5000)
except Exception as e:
    logger.error(f"Error starting app: {e}", exc_info=True)
    raise

```

Writing /content/whisper-app/app.py

1.1.1 setup html

```

[ ]: %%writefile /content/whisper-app/templates/index.html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Whisper AI Transcription</title>
    <link rel="stylesheet" href="{ url_for('static', filename='css/styles.
↵css') }}">
    <script src="https://cdn.tailwindcss.com"></script>
</head>
<body>
    <div class="app-container">
        <!-- Theme Toggle Button -->
        <button id="themeToggle" class="theme-toggle">
            <svg id="moonIcon" class="icon" viewBox="0 0 24 24">
                <path d="M21 12.79A9 9 0 1 1 11.21 3 7 7 0 0 0 21 12.79z"></
↵path>
            </svg>
            <svg id="sunIcon" class="icon" viewBox="0 0 24 24">
                <circle cx="12" cy="12" r="5"></circle>
                <line x1="12" y1="1" x2="12" y2="3"></line>
                <line x1="12" y1="21" x2="12" y2="23"></line>
                <line x1="4.22" y1="4.22" x2="5.64" y2="5.64"></line>
                <line x1="18.36" y1="18.36" x2="19.78" y2="19.78"></line>
                <line x1="1" y1="12" x2="3" y2="12"></line>
                <line x1="21" y1="12" x2="23" y2="12"></line>
                <line x1="4.22" y1="19.78" x2="5.64" y2="18.36"></line>

```

```

        <line x1="18.36" y1="5.64" x2="19.78" y2="4.22"></line>
    </svg>
</button>

<div class="container">
    <header>
        <h1>Whisper AI Transcription</h1>
        <p>Convert your audio and video files into accurate
↳ transcriptions and translations using state-of-the-art AI technology.</p>
    </header>

    <div class="main-content">
        <!-- Upload and Settings Card -->
        <div class="card upload-card">
            <div class="card-header">
                <div class="card-title">
                    <svg class="icon" viewBox="0 0 24 24">
                        <path d="M9 18V5l12-2v13"></path>
                        <path d="M9 9l12-2"></path>
                        <circle cx="6" cy="18" r="3"></circle>
                        <circle cx="18" cy="16" r="3"></circle>
                    </svg>
                    <span>Audio Transcription</span>
                </div>
                <p class="card-description">Upload your audio or video
↳ file to get started</p>
            </div>

            <form id="transcriptionForm" enctype="multipart/form-data">
                <div class="form-group">
                    <label for="audioFile">Upload File</label>
                    <div id="uploadArea" class="upload-area">
                        <div class="upload-content">
                            <div id="uploadIcon" class="upload-icon">
                                <svg class="icon" viewBox="0 0 24 24">
                                    <path d="M21 15v4a2 2 0 0 1-2 2H5a2
↳ 2 0 0 1-2-2v-4"></path>
                                    <polyline points="17 8 12 3 7 8"></
↳ polyline>
                                    <line x1="12" y1="3" x2="12"
↳ y2="15"></line>
                                </svg>
                            </div>
                            <div id="fileInfo" class="file-info hidden">
                                <svg class="icon" viewBox="0 0 24 24">
                                    <path d="M22 11.08V12a10 10 0 1 1-5.
↳ 93-9.14"></path>

```

```

        <polyline points="22 4 12 14.01 9_
↪11.01"></polyline>

        </svg>
        <p id="fileName" class="file-name"></p>
        <p id="fileSize" class="file-size"></p>
    </div>
    <div id="uploadText" class="upload-text">
        <p>Drag & drop or click to upload</p>
        <p>Supports audio and video files (MP3,
↪WAV, MP4, etc.)</p>
    </div>
</div>
    <input type="file" id="audioFile" name="file"
↪accept="audio/*,video/*" class="hidden" required>
</div>
</div>

<div class="form-row">
    <div class="form-group">
        <label for="language">Language</label>
        <div class="select-container">
            <select id="language" name="language"
↪class="select">
                <option value="auto">Auto-detect</
↪option>
                <option value="en">English</option>
                <option value="es">Spanish</option>
                <option value="fr">French</option>
                <option value="de">German</option>
                <option value="it">Italian</option>
                <option value="pt">Portuguese</option>
                <option value="ru">Russian</option>
                <option value="zh">Chinese</option>
                <option value="ja">Japanese</option>
                <option value="ko">Korean</option>
            </select>
            <div class="select-arrow">
                <svg class="icon" viewBox="0 0 24 24">
                    <polyline points="6 9 12 15 18 9"></
↪polyline>
                </svg>
            </div>
        </div>
    </div>
</div>

<div class="form-group">

```



```

        <label for="format">Output Format</label>
        <div class="select-container">
            <select id="format" name="format"␣
↪class="select">
                <option value="srt">SRT Subtitles</
↪option>
                    <option value="txt">Plain Text</option>
                    <option value="json">JSON</option>
                </select>
                <div class="select-arrow">
                    <svg class="icon" viewBox="0 0 24 24">
                        <polyline points="6 9 12 15 18 9"></
↪polyline>
                            </svg>
                        </div>
                    </div>
                </div>
            </div>

            <div class="form-group switch-group">
                <label class="switch">
                    <input type="checkbox" id="translate"␣
↪name="translate">
                        <span class="slider"></span>
                    </label>
                    <label for="translate"␣
↪class="switch-label">Translate to English</label>
                </div>

                <div class="form-actions">
                    <button type="submit" id="submitBtn" class="btn␣
↪btn-primary">
                        <span id="submitBtnText">
                            <svg class="icon" viewBox="0 0 24 24">
                                <path d="M14 2H6a2 2 0 0 0-2 2v16a2 2 0␣
↪0 0 2 2h12a2 2 0 0 0 2-2V8z"></path>
                                    <polyline points="14 2 14 8 20 8"></
↪polyline>
                                        <line x1="16" y1="13" x2="8" y2="13"></
↪line>
                                            <line x1="16" y1="17" x2="8" y2="17"></
↪line>
                                                <polyline points="10 9 9 9 8 9"></
↪polyline>
                                                    </svg>
                                                    Transcribe

```

```

        </span>
        <span id="processingBtnText" class="hidden">
            <svg class="icon spin" viewBox="0 0 24 24">
                <circle cx="12" cy="12" r="10"
↳stroke-width="4" stroke-dasharray="30 30" stroke-dashoffset="0"></circle>
            </svg>
            Processing...
        </span>
    </button>
</div>

    <div id="progressContainer" class="progress-container
↳hidden">

        <div class="progress-bar">
            <div id="progressBar" class="progress-fill"></
↳div>

        </div>
        <p id="statusText" class="status-text"></p>
    </div>
</form>
</div>

<!-- Results Card -->
<div class="card results-card">
    <div class="card-header">
        <div class="card-title">
            <svg class="icon" viewBox="0 0 24 24">
                <polygon points="12 2 15.09 8.26 22 9.27 17 14.
↳14 18.18 21.02 12 17.77 5.82 21.02 7 14.14 2 9.27 8.91 8.26 12 2"></polygon>
            </svg>
            <span>Results</span>
        </div>
        <p class="card-description">Your transcription results
↳will appear here</p>
    </div>

    <div id="resultsContent" class="results-content">
        <div id="noResults" class="no-results">
            <svg class="icon" viewBox="0 0 24 24">
                <path d="M14 2H6a2 2 0 0 0-2 2v16a2 2 0 0 0 2
↳2h12a2 2 0 0 0 2-2V8z"></path>
                <polyline points="14 2 14 8 20 8"></polyline>
                <line x1="16" y1="13" x2="8" y2="13"></line>
                <line x1="16" y1="17" x2="8" y2="17"></line>
                <polyline points="10 9 9 8 9"></polyline>
            </svg>

```

```

        <p>No transcription results yet</p>
        <p>Upload a file and click Transcribe to get
↪started</p>
    </div>

    <div id="resultsData" class="results-data hidden">
        <div class="result-item">
            <p class="result-label">Detected Language</p>
            <div class="language-info">
                <p id="detectedLanguage"
↪class="detected-language"></p>
                <span id="confidenceBadge"
↪class="confidence-badge"></span>
            </div>
        </div>

        <div class="tabs">
            <div class="tab-list">
                <div id="transcriptionTab" class="tab
↪active">Transcription</div>
                <div id="translationTab"
↪class="tab">Translation</div>
            </div>

            <div id="transcriptionContent"
↪class="tab-content active">
                <div class="result-item">
                    <div>
                        <p class="result-label">Original
↪Transcription</p>
                        <p id="formatBadge"
↪class="format-badge"></p>
                    </div>
                    <a id="downloadOriginal" href="#"
↪class="btn btn-outline">
                        <svg class="icon" viewBox="0 0 24
↪24">
                            <path d="M21 15v4a2 2 0 0 1-2
↪2H5a2 2 0 0 1-2-2v-4"></path>
                            <polyline points="7 10 12 15 17
↪10"></polyline>
                            <line x1="12" y1="15" x2="12"
↪y2="3"></line>
                        </svg>
                        Download
                    </a>

```

```

        </div>
    </div>


    <div id="translationContent"
class="tab-content">
        <div class="result-item">
            <div>
                <p class="result-label">English
Translation</p>
                <p id="translationFormatBadge"
class="format-badge"></p>
            </div>
            <a id="downloadTranslation" href="#"
class="btn btn-outline">
                <svg class="icon" viewBox="0 0 24
24">
                    <path d="M21 15v4a2 2 0 0 1-2
2H5a2 2 0 0 1-2-2v-4"></path>
                    <polyline points="7 10 12 15 17
10"></polyline>
                    <line x1="12" y1="15" x2="12"
y2="3"></line>
                </svg>
                Download
            </a>
        </div>
    </div>
</div>
</div>
</div>

    <div id="resetContainer" class="reset-container hidden">
        <button id="resetBtn" class="btn btn-outline">
            <svg class="icon" viewBox="0 0 24 24">
                <path d="M23 4v6h-6"></path>
                <path d="M1 20v-6h6"></path>
                <path d="M3.51 9a9 9 0 0 1 14.85-3.36L23 10M1
14l4.64 4.36A9 9 0 0 0 20.49 15"></path>
            </svg>
            Start New Transcription
        </button>
    </div>
</div>

<footer>

```

```

        <p>Powered by Whisper AI  Large-v3 Model</p>
    </footer>
</div>
</div>

<script src="{ { url_for('static', filename='js/main.js') } }"></script>
</body>
</html>

```

Writing /content/whisper-app/templates/index.html

1.1.2 setup css

```

[ ]: %%writefile /content/whisper-app/static/css/styles.css
@import url('https://fonts.googleapis.com/css2?family=Inter:wght@300;400;500;
↳600;700&display=swap');

:root {
    /* Light mode colors */
    --light-bg-primary: #f0f9ff;
    --light-bg-secondary: #e0f2fe;
    --light-text-primary: #0f172a;
    --light-text-secondary: #334155;
    --light-accent-primary: #2563eb;
    --light-accent-secondary: #3b82f6;
    --light-border: #bfdbfe;
    --light-card-bg: rgba(255, 255, 255, 0.9);
    --light-hover: #eff6ff;
    --light-input-bg: #ffffff;

    /* Dark mode colors */
    --dark-bg-primary: #0c0a09;
    --dark-bg-secondary: #1c1917;
    --dark-text-primary: #f8fafc;
    --dark-text-secondary: #cbd5e1;
    --dark-accent-primary: #eab308;
    --dark-accent-secondary: #facc15;
    --dark-border: #78350f;
    --dark-card-bg: rgba(0, 0, 0, 0.8);
    --dark-hover: rgba(245, 158, 11, 0.1);
    --dark-input-bg: #1c1917;

    /* Common variables */
    --radius: 1rem;
    --shadow-sm: 0 1px 2px 0 rgba(0, 0, 0, 0.05);
    --shadow: 0 4px 6px -1px rgba(0, 0, 0, 0.1), 0 2px 4px -1px rgba(0, 0, 0, 0.
↳06);

```

```

    --shadow-lg: 0 10px 15px -3px rgba(0, 0, 0, 0.1), 0 4px 6px -2px rgba(0, 0, 0, 0.05);
    --transition: all 0.3s ease;
}

* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    font-family: 'Inter', sans-serif;
}

body {
    transition: background-color 0.5s ease, color 0.5s ease;
    background: linear-gradient(to bottom, var(--light-bg-primary),
    ↪var(--light-bg-secondary));
    color: var(--light-text-primary);
    min-height: 100vh;
}

body.dark {
    background: linear-gradient(to bottom, var(--dark-bg-primary),
    ↪var(--dark-bg-secondary));
    color: var(--dark-text-primary);
}

.app-container {
    position: relative;
    min-height: 100vh;
    padding: 1rem;
}

.container {
    max-width: 1200px;
    margin: 0 auto;
    padding: 2rem 1rem;
}

/* Header Styles */
header {
    text-align: center;
    margin-bottom: 3rem;
}

header h1 {
    font-size: 2.5rem;
    font-weight: 700;
}

```

```

    margin-bottom: 0.5rem;
    background: linear-gradient(to right, var(--light-accent-primary),
↪var(--light-accent-secondary));
    -webkit-background-clip: text;
    -webkit-text-fill-color: transparent;
    background-clip: text;
}

body.dark header h1 {
    background: linear-gradient(to right, var(--dark-accent-primary),
↪var(--dark-accent-secondary));
    -webkit-background-clip: text;
    -webkit-text-fill-color: transparent;
    background-clip: text;
}

header p {
    font-size: 1.1rem;
    color: var(--light-text-secondary);
    max-width: 600px;
    margin: 0 auto;
}

body.dark header p {
    color: var(--dark-text-secondary);
}

/* Main Content Styles */
.main-content {
    display: grid;
    grid-template-columns: 1fr;
    gap: 2rem;
}

@media (min-width: 1024px) {
    .main-content {
        grid-template-columns: 3fr 2fr;
    }
}

/* Card Styles */
.card {
    background-color: var(--light-card-bg);
    border-radius: var(--radius);
    border: 1px solid var(--light-border);
    box-shadow: var(--shadow);
    padding: 1.5rem;
}

```

```

    transition: var(--transition);
}

body.dark .card {
    background-color: var(--dark-card-bg);
    border-color: var(--dark-border);
}

.card-header {
    margin-bottom: 1.5rem;
}

.card-title {
    display: flex;
    align-items: center;
    gap: 0.5rem;
    font-size: 1.25rem;
    font-weight: 600;
    color: var(--light-accent-primary);
    margin-bottom: 0.25rem;
}

body.dark .card-title {
    color: var(--dark-accent-primary);
}

.card-description {
    font-size: 0.875rem;
    color: var(--light-text-secondary);
}

body.dark .card-description {
    color: var(--dark-text-secondary);
}

/* Form Styles */
.form-group {
    margin-bottom: 1.5rem;
}

.form-row {
    display: grid;
    grid-template-columns: 1fr;
    gap: 1rem;
    margin-bottom: 1.5rem;
}

```



```

@media (min-width: 768px) {
  .form-row {
    grid-template-columns: 1fr 1fr;
  }
}

label {
  display: block;
  font-size: 0.875rem;
  font-weight: 500;
  margin-bottom: 0.5rem;
  color: var(--light-accent-primary);
}

body.dark label {
  color: var(--dark-accent-primary);
}

.upload-area {
  border: 2px dashed var(--light-border);
  border-radius: var(--radius);
  padding: 2rem 1rem;
  cursor: pointer;
  transition: var(--transition);
  text-align: center;
}

.upload-area:hover {
  border-color: var(--light-accent-primary);
  background-color: var(--light-hover);
}

body.dark .upload-area {
  border-color: var(--dark-border);
}

body.dark .upload-area:hover {
  border-color: var(--dark-accent-primary);
  background-color: var(--dark-hover);
}

.upload-content {
  display: flex;
  flex-direction: column;
  align-items: center;
  gap: 1rem;
}

```

```

.upload-icon {
  animation: bounce 2s infinite;
}

.file-info {
  display: flex;
  flex-direction: column;
  align-items: center;
}

.file-name {
  font-size: 0.875rem;
  font-weight: 500;
  color: var(--light-accent-primary);
}

body.dark .file-name {
  color: var(--dark-accent-primary);
}

.file-size {
  font-size: 0.75rem;
  color: var(--light-text-secondary);
}

body.dark .file-size {
  color: var(--dark-text-secondary);
}

.upload-text p:first-child {
  font-size: 0.875rem;
  font-weight: 500;
  color: var(--light-accent-primary);
  margin-bottom: 0.25rem;
}

body.dark .upload-text p:first-child {
  color: var(--dark-accent-primary);
}

.upload-text p:last-child {
  font-size: 0.75rem;
  color: var(--light-text-secondary);
}

body.dark .upload-text p:last-child {

```

```

        color: var(--dark-text-secondary);
    }

    .select-container {
        position: relative;
    }

    .select {
        appearance: none;
        width: 100%;
        padding: 0.75rem 1rem;
        font-size: 0.875rem;
        border-radius: var(--radius);
        border: 1px solid var(--light-border);
        background-color: var(--light-input-bg);
        color: var(--light-text-primary);
        transition: var(--transition);
    }

    .select:hover, .select:focus {
        border-color: var(--light-accent-primary);
        outline: none;
    }

    body.dark .select {
        border-color: var(--dark-border);
        background-color: var(--dark-input-bg);
        color: var(--dark-text-primary);
    }

    body.dark .select:hover, body.dark .select:focus {
        border-color: var(--dark-accent-primary);
    }

    .select-arrow {
        position: absolute;
        right: 1rem;
        top: 50%;
        transform: translateY(-50%);
        pointer-events: none;
        color: var(--light-accent-primary);
    }

    body.dark .select-arrow {
        color: var(--dark-accent-primary);
    }

```

```

.switch-group {
  display: flex;
  align-items: center;
  gap: 0.75rem;
}

.switch {
  position: relative;
  display: inline-block;
  width: 3rem;
  height: 1.5rem;
}

.switch input {
  opacity: 0;
  width: 0;
  height: 0;
}

.slider {
  position: absolute;
  cursor: pointer;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  background-color: var(--light-bg-secondary);
  transition: var(--transition);
  border-radius: 1.5rem;
}

.slider:before {
  position: absolute;
  content: "";
  height: 1rem;
  width: 1rem;
  left: 0.25rem;
  bottom: 0.25rem;
  background-color: white;
  transition: var(--transition);
  border-radius: 50%;
}

input:checked + .slider {
  background-color: var(--light-accent-primary);
}

```

```

input:checked + .slider:before {
  transform: translateX(1.5rem);
}

body.dark .slider {
  background-color: var(--dark-bg-secondary);
}

body.dark input:checked + .slider {
  background-color: var(--dark-accent-primary);
}

.switch-label {
  font-size: 0.875rem;
  font-weight: 500;
}

.btn {
  display: inline-flex;
  align-items: center;
  justify-content: center;
  gap: 0.5rem;
  padding: 0.75rem 1.5rem;
  font-size: 0.875rem;
  font-weight: 500;
  border-radius: var(--radius);
  cursor: pointer;
  transition: var(--transition);
  border: none;
}

.btn:hover {
  transform: translateY(-2px);
}

.btn:active {
  transform: translateY(1px);
}

.btn-primary {
  background: linear-gradient(to right, var(--light-accent-primary),
↪var(--light-accent-secondary));
  color: white;
}

.btn-primary:hover {

```

```

        background: linear-gradient(to right, var(--light-accent-secondary),
↪var(--light-accent-primary));
    }

body.dark .btn-primary {
    background: linear-gradient(to right, var(--dark-accent-primary),
↪var(--dark-accent-secondary));
    color: black;
}

body.dark .btn-primary:hover {
    background: linear-gradient(to right, var(--dark-accent-secondary),
↪var(--dark-accent-primary));
}

.btn-outline {
    background: transparent;
    border: 1px solid var(--light-border);
    color: var(--light-accent-primary);
}

.btn-outline:hover {
    background-color: var(--light-hover);
    border-color: var(--light-accent-primary);
}

body.dark .btn-outline {
    border-color: var(--dark-border);
    color: var(--dark-accent-primary);
}

body.dark .btn-outline:hover {
    background-color: var(--dark-hover);
    border-color: var(--dark-accent-primary);
}

.form-actions {
    margin-top: 2rem;
}

.form-actions .btn {
    width: 100%;
    padding: 1rem;
}

.progress-container {
    margin-top: 1.5rem;
}

```

```

}

.progress-bar {
  height: 0.75rem;
  background-color: var(--light-bg-secondary);
  border-radius: 1rem;
  overflow: hidden;
}

.progress-fill {
  height: 100%;
  background-color: var(--light-accent-primary);
  width: 0%;
  transition: width 0.5s ease;
}

body.dark .progress-bar {
  background-color: var(--dark-bg-secondary);
}

body.dark .progress-fill {
  background-color: var(--dark-accent-primary);
}

.status-text {
  font-size: 0.75rem;
  text-align: center;
  margin-top: 0.5rem;
  color: var(--light-text-secondary);
}

body.dark .status-text {
  color: var(--dark-text-secondary);
}

/* Results Styles */
.results-content {
  min-height: 300px;
  display: flex;
  flex-direction: column;
}

.no-results {
  flex: 1;
  display: flex;
  flex-direction: column;
  align-items: center;
}

```

```

    justify-content: center;
    color: var(--light-text-secondary);
    opacity: 0.7;
}

body.dark .no-results {
    color: var(--dark-text-secondary);
}

.no-results .icon {
    width: 4rem;
    height: 4rem;
    margin-bottom: 1rem;
    animation: pulse 2s infinite;
}

.no-results p:first-of-type {
    font-size: 0.875rem;
    margin-bottom: 0.25rem;
}

.no-results p:last-of-type {
    font-size: 0.75rem;
}

.results-data {
    display: flex;
    flex-direction: column;
    gap: 1rem;
}

.result-item {
    background-color: var(--light-hover);
    border: 1px solid var(--light-border);
    border-radius: var(--radius);
    padding: 1rem;
    display: flex;
    justify-content: space-between;
    align-items: center;
}

body.dark .result-item {
    background-color: var(--dark-hover);
    border-color: var(--dark-border);
}

.result-label {

```



```

    font-size: 0.875rem;
    font-weight: 500;
    color: var(--light-accent-primary);
    margin-bottom: 0.25rem;
}

body.dark .result-label {
    color: var(--dark-accent-primary);
}

.format-badge {
    font-size: 0.75rem;
    color: var(--light-text-secondary);
}

body.dark .format-badge {
    color: var(--dark-text-secondary);
}

.language-info {
    display: flex;
    align-items: center;
    justify-content: space-between;
    width: 100%;
}

.detected-language {
    font-size: 1.125rem;
    font-weight: 600;
    color: var(--light-text-primary);
}

body.dark .detected-language {
    color: var(--dark-text-primary);
}

.confidence-badge {
    font-size: 0.75rem;
    padding: 0.25rem 0.75rem;
    border-radius: 9999px;
    background-color: var(--light-bg-secondary);
    color: var(--light-accent-primary);
}

body.dark .confidence-badge {
    background-color: var(--dark-bg-secondary);
    color: var(--dark-accent-primary);
}

```

```

}

.tabs {
  margin-top: 1rem;
}

.tab-list {
  display: flex;
  background-color: var(--light-bg-secondary);
  border-radius: var(--radius);
  padding: 0.25rem;
  margin-bottom: 1rem;
}

body.dark .tab-list {
  background-color: var(--dark-bg-secondary);
}

.tab {
  flex: 1;
  text-align: center;
  padding: 0.5rem;
  cursor: pointer;
  border-radius: calc(var(--radius) - 0.25rem);
  transition: var(--transition);
  font-size: 0.875rem;
}

.tab.active {
  background-color: var(--light-input-bg);
  color: var(--light-accent-primary);
  font-weight: 500;
}

body.dark .tab.active {
  background-color: var(--dark-input-bg);
  color: var(--dark-accent-primary);
}

.tab-content {
  display: none;
}

.tab-content.active {
  display: block;
}

```

```

.reset-container {
  margin-top: 1.5rem;
}

.reset-container .btn {
  width: 100%;
  padding: 1rem;
}

/* Theme Toggle */
.theme-toggle {
  position: absolute;
  top: 1rem;
  right: 1rem;
  width: 3rem;
  height: 3rem;
  border-radius: 50%;
  background-color: var(--light-input-bg);
  border: 1px solid var(--light-border);
  color: var(--light-accent-primary);
  display: flex;
  align-items: center;
  justify-content: center;
  cursor: pointer;
  transition: var(--transition);
  z-index: 10;
}

.theme-toggle:hover {
  background-color: var(--light-hover);
}

body.dark .theme-toggle {
  background-color: var(--dark-input-bg);
  border-color: var(--dark-border);
  color: var(--dark-accent-primary);
}

body.dark .theme-toggle:hover {
  background-color: var(--dark-hover);
}

#moonIcon {
  display: block;
}

#sunIcon {

```

```

        display: none;
    }

    body.dark #moonIcon {
        display: none;
    }

    body.dark #sunIcon {
        display: block;
    }

    /* Footer */
    footer {
        text-align: center;
        margin-top: 3rem;
        font-size: 0.875rem;
        color: var(--light-text-secondary);
    }

    body.dark footer {
        color: var(--dark-text-secondary);
    }

    /* Icons */
    .icon {
        width: 1.25rem;
        height: 1.25rem;
        stroke: currentColor;
        stroke-width: 2;
        stroke-linecap: round;
        stroke-linejoin: round;
        fill: none;
    }

    /* Animations */
    @keyframes bounce {
        0%, 100% { transform: translateY(0); }
        50% { transform: translateY(-10px); }
    }

    @keyframes pulse {
        0%, 100% { opacity: 0.7; }
        50% { opacity: 0.4; }
    }

    @keyframes spin {
        from { transform: rotate(0deg); }

```

```

    to { transform: rotate(360deg); }
}

.spin {
  animation: spin 1.5s linear infinite;
}

/* Utilities */
.hidden {
  display: none !important;
}

/* Responsive adjustments */
@media (max-width: 768px) {
  header h1 {
    font-size: 2rem;
  }

  header p {
    font-size: 1rem;
  }

  .card {
    padding: 1.25rem;
  }
}

@media (max-width: 480px) {
  header h1 {
    font-size: 1.75rem;
  }

  .upload-area {
    padding: 1.5rem 1rem;
  }

  .btn {
    padding: 0.625rem 1.25rem;
  }
}

```

Writing `/content/whisper-app/static/css/styles.css`

1.1.3 setup javascript

```
[ ]: %%%writefile /content/whisper-app/static/js/main.js
document.addEventListener('DOMContentLoaded', function() {
    // Get the current URL from the server
    const currentUrl = window.location.origin;

    // DOM Elements
    const themeToggle = document.getElementById('themeToggle');
    const uploadArea = document.getElementById('uploadArea');
    const audioFileInput = document.getElementById('audioFile');
    const uploadIcon = document.getElementById('uploadIcon');
    const fileInfo = document.getElementById('fileInfo');
    const uploadText = document.getElementById('uploadText');
    const fileName = document.getElementById('fileName');
    const fileSize = document.getElementById('fileSize');
    const transcriptionForm = document.getElementById('transcriptionForm');
    const submitBtn = document.getElementById('submitBtn');
    const submitBtnText = document.getElementById('submitBtnText');
    const processingBtnText = document.getElementById('processingBtnText');
    const progressContainer = document.getElementById('progressContainer');
    const progressBar = document.getElementById('progressBar');
    const statusText = document.getElementById('statusText');
    const noResults = document.getElementById('noResults');
    const resultsData = document.getElementById('resultsData');
    const detectedLanguage = document.getElementById('detectedLanguage');
    const confidenceBadge = document.getElementById('confidenceBadge');
    const formatBadge = document.getElementById('formatBadge');
    const translationFormatBadge = document.
    ↪getElementById('translationFormatBadge');
    const downloadOriginal = document.getElementById('downloadOriginal');
    const downloadTranslation = document.getElementById('downloadTranslation');
    const transcriptionTab = document.getElementById('transcriptionTab');
    const translationTab = document.getElementById('translationTab');
    const transcriptionContent = document.
    ↪getElementById('transcriptionContent');
    const translationContent = document.getElementById('translationContent');
    const resetContainer = document.getElementById('resetContainer');
    const resetBtn = document.getElementById('resetBtn');

    // Theme Toggle
    function initTheme() {
        const savedTheme = localStorage.getItem('theme');
        if (savedTheme === 'dark') {
            document.body.classList.add('dark');
        }
    }
}
```

```

themeToggle.addEventListener('click', function() {
    document.body.classList.toggle('dark');
    const isDark = document.body.classList.contains('dark');
    localStorage.setItem('theme', isDark ? 'dark' : 'light');
});

// Initialize theme
initTheme();

// File Upload Handling
uploadArea.addEventListener('click', function() {
    audioFileInput.click();
});

uploadArea.addEventListener('dragover', function(e) {
    e.preventDefault();
    uploadArea.classList.add('dragover');
});

uploadArea.addEventListener('dragleave', function() {
    uploadArea.classList.remove('dragover');
});

uploadArea.addEventListener('drop', function(e) {
    e.preventDefault();
    uploadArea.classList.remove('dragover');

    if (e.dataTransfer.files.length) {
        audioFileInput.files = e.dataTransfer.files;
        handleFileSelect(e.dataTransfer.files[0]);
    }
});

audioFileInput.addEventListener('change', function() {
    if (this.files.length) {
        handleFileSelect(this.files[0]);
    }
});

function handleFileSelect(file) {
    uploadIcon.classList.add('hidden');
    fileInfo.classList.remove('hidden');
    uploadText.classList.add('hidden');

    fileName.textContent = file.name;
    fileSize.textContent = formatFileSize(file.size);
}

```

```

}

function formatFileSize(bytes) {
  if (bytes < 1024) return bytes + ' bytes';
  else if (bytes < 1048576) return (bytes / 1024).toFixed(1) + ' KB';
  else return (bytes / 1048576).toFixed(1) + ' MB';
}

// Form Submission
transcriptionForm.addEventListener('submit', function(e) {
  e.preventDefault();

  if (!audioFileInput.files.length) {
    alert('Please select a file to transcribe');
    return;
  }

  // Show processing state
  submitBtnText.classList.add('hidden');
  processingBtnText.classList.remove('hidden');
  submitBtn.disabled = true;

  // Show progress container
  progressContainer.classList.remove('hidden');
  progressBar.style.width = '0%';
  statusText.textContent = 'Uploading file...';

  // Create FormData
  const formData = new FormData(transcriptionForm);

  // Simulate progress (since we can't get real progress from the server)
  let progress = 0;
  const progressInterval = setInterval(() => {
    if (progress < 90) {
      progress += Math.random() * 10;
      progressBar.style.width = progress + '%';

      if (progress < 30) {
        statusText.textContent = 'Uploading file...';
      } else if (progress < 60) {
        statusText.textContent = 'Processing audio...';
      } else {
        statusText.textContent = 'Generating transcription...';
      }
    }
  }, 1000);

```



```

// Send request to server
fetch(`${currentUrl}/transcribe`, {
  method: 'POST',
  body: formData
})
.then(response => {
  if (!response.ok) {
    throw new Error('Server error: ' + response.status);
  }
  return response.json();
})
.then(data => {
  // Clear progress interval
  clearInterval(progressInterval);

  // Complete progress bar
  progressBar.style.width = '100%';
  statusText.textContent = 'Transcription complete!';

  // Update results
  updateResults(data);

  // Reset form state after a delay
  setTimeout(() => {
    progressContainer.classList.add('hidden');
    submitBtnText.classList.remove('hidden');
    processingBtnText.classList.add('hidden');
    submitBtn.disabled = false;
  }, 1000);
})
.catch(error => {
  // Clear progress interval
  clearInterval(progressInterval);

  // Show error
  progressBar.style.width = '100%';
  progressBar.style.backgroundColor = '#ef4444';
  statusText.textContent = 'Error: ' + error.message;

  // Reset form state after a delay
  setTimeout(() => {
    progressContainer.classList.add('hidden');
    submitBtnText.classList.remove('hidden');
    processingBtnText.classList.add('hidden');
    submitBtn.disabled = false;
  }, 3000);
});

```

```

        console.error('Error:', error);
    });
});

// Update Results
function updateResults(data) {
    // Hide no results message
    noResults.classList.add('hidden');

    // Show results data
    resultsData.classList.remove('hidden');

    // Show reset container
    resetContainer.classList.remove('hidden');

    // Update language info
    const languageName = getLanguageName(data.language);
    detectedLanguage.textContent = languageName;

    // Update confidence badge
    const confidence = data.detected_language_confidence.toFixed(1);
    confidenceBadge.textContent = confidence + '% confidence';

    // Get format from form
    const format = document.getElementById('format').value;

    // Update format badges
    formatBadge.textContent = format.toUpperCase();
    translationFormatBadge.textContent = format.toUpperCase();

    // Update download links
    if (data.original) {
        downloadOriginal.href = `${currentUrl}/download/${data.original}`;
        downloadOriginal.classList.remove('hidden');
    } else {
        downloadOriginal.classList.add('hidden');
    }

    if (data.translation) {
        downloadTranslation.href = `${currentUrl}/download/${data.
↪translation}`;
        downloadTranslation.classList.remove('hidden');
        translationTab.classList.remove('hidden');
    } else {
        downloadTranslation.classList.add('hidden');
        translationTab.classList.add('hidden');
    }
}

```

```

        // Make sure transcription tab is active if translation is not
        ↪available
        transcriptionTab.classList.add('active');
        translationTab.classList.remove('active');
        transcriptionContent.classList.add('active');
        translationContent.classList.remove('active');
    }
}

// Tab Switching
transcriptionTab.addEventListener('click', function() {
    transcriptionTab.classList.add('active');
    translationTab.classList.remove('active');
    transcriptionContent.classList.add('active');
    translationContent.classList.remove('active');
});

translationTab.addEventListener('click', function() {
    translationTab.classList.add('active');
    transcriptionTab.classList.remove('active');
    translationContent.classList.add('active');
    transcriptionContent.classList.remove('active');
});

// Reset Button
resetBtn.addEventListener('click', function() {
    // Reset file input
    audioFileInput.value = '';

    // Reset file info display
    uploadIcon.classList.remove('hidden');
    fileInfo.classList.add('hidden');
    uploadText.classList.remove('hidden');

    // Hide results
    noResults.classList.remove('hidden');
    resultsData.classList.add('hidden');
    resetContainer.classList.add('hidden');
});

// Helper function to get language name from code
function getLanguageName(code) {
    const languages = {
        'en': 'English',
        'es': 'Spanish',
        'fr': 'French',
        'de': 'German',
    };

```

```
'it': 'Italian',  
'pt': 'Portuguese',  
'ru': 'Russian',  
'zh': 'Chinese',  
'ja': 'Japanese',  
'ko': 'Korean',  
'ar': 'Arabic',  
'hi': 'Hindi',  
'bn': 'Bengali',  
'ur': 'Urdu',  
'te': 'Telugu',  
'ta': 'Tamil',  
'mr': 'Marathi',  
'gu': 'Gujarati',  
'kn': 'Kannada',  
'ml': 'Malayalam',  
'pa': 'Punjabi',  
'or': 'Odia',  
'as': 'Assamese',  
'nl': 'Dutch',  
'tr': 'Turkish',  
'pl': 'Polish',  
'uk': 'Ukrainian',  
'cs': 'Czech',  
'sv': 'Swedish',  
'fi': 'Finnish',  
'no': 'Norwegian',  
'da': 'Danish',  
'hu': 'Hungarian',  
'ro': 'Romanian',  
'bg': 'Bulgarian',  
'el': 'Greek',  
'he': 'Hebrew',  
'th': 'Thai',  
'vi': 'Vietnamese',  
'id': 'Indonesian',  
'ms': 'Malay',  
'fa': 'Persian',  
'sw': 'Swahili',  
'af': 'Afrikaans',  
'sq': 'Albanian',  
'am': 'Amharic',  
'hy': 'Armenian',  
'az': 'Azerbaijani',  
'eu': 'Basque',  
'be': 'Belarusian',  
'bs': 'Bosnian',
```

```

        'ca': 'Catalan',
        'hr': 'Croatian',
        'et': 'Estonian',
        'tl': 'Filipino',
        'gl': 'Galician',
        'ka': 'Georgian',
        'ha': 'Hausa',
        'is': 'Icelandic',
        'ig': 'Igbo',
        'ga': 'Irish',
        'jv': 'Javanese',
        'kk': 'Kazakh',
        'km': 'Khmer',
        'ky': 'Kyrgyz',
        'lo': 'Lao',
        'lv': 'Latvian',
        'lt': 'Lithuanian',
        'lb': 'Luxembourgish',
        'mk': 'Macedonian',
        'mg': 'Malagasy',
        'mt': 'Maltese',
        'mi': 'Maori',
        'mn': 'Mongolian',
        'my': 'Myanmar (Burmese)',
        'ne': 'Nepali',
        'ps': 'Pashto',
        'sr': 'Serbian',
        'st': 'Sesotho',
        'si': 'Sinhala',
        'sk': 'Slovak',
        'sl': 'Slovenian',
        'so': 'Somali',
        'su': 'Sundanese',
        'tg': 'Tajik',
        'tt': 'Tatar',
        'tk': 'Turkmen',
        'uz': 'Uzbek',
        'cy': 'Welsh',
        'xh': 'Xhosa',
        'yi': 'Yiddish',
        'yo': 'Yoruba',
        'zu': 'Zulu'
    };

    return languages[code] || code;
}
});

```

Writing /content/whisper-app/static/js/main.js

2 setup run.py / Also setup ngrok token here

```
[ ]: %%writefile /content/whisper-app/run.py
import logging
import sys
from app import start_app, load_whisper_model

if __name__ == "__main__":
    try:
        print("Initializing Whisper AI Transcription App...")
        print("This may take a few minutes to download and load the model...")

        # Pre-load the model before starting the app
        # This ensures the model is loaded before the first request
        print("Pre-loading Whisper model...")
        model = load_whisper_model()
        print("Model pre-loaded successfully!")

        # Start the Flask app with the ngrok token
        print("Starting web server...")

    except KeyboardInterrupt:
        print("\nShutting down gracefully...")
    except Exception as e:
        print(f"Error: {e}")
        logging.error("Fatal error", exc_info=True)
        sys.exit(1)
```

Writing /content/whisper-app/run.py

2.1 RUN THE WEBAPP FINALLY

```
[ ]: # Change to the app directory
%cd /content/whisper-app

# Run the application with your ngrok token
!python run.py
```

/content/whisper-app

Initializing Whisper AI Transcription App...

This may take a few minutes to download and load the model...

```
Pre-loading Whisper model...
2025-05-30 12:01:33,122 - INFO - Starting to load Whisper model...
2025-05-30 12:01:36,844 - INFO - Using device: cuda
2025-05-30 12:01:36,845 - INFO - Loading model size: large-v3
...Xet Storage is enabled for this repo, but the 'hf_xet' package is not
installed. Falling back to regular HTTP download. For better performance,
install the package with: `pip install huggingface_hub[hf_xet]` or `pip install
hf_xet`
2025-05-30 12:01:39,632 - WARNING - Xet Storage is enabled for this repo, but
the 'hf_xet' package is not installed. Falling back to regular HTTP download.
For better performance, install the package with: `pip install
huggingface_hub[hf_xet]` or `pip install hf_xet`
preprocessor_config.json: 100% 340/340 [00:00<00:00, 2.74MB/s]
vocabulary.json: 0% 0.00/1.07M [00:00<?, ?B/s]
config.json: 100% 2.39k/2.39k [00:00<00:00, 20.7MB/s]

vocabulary.json: 100% 1.07M/1.07M [00:00<00:00, 30.7MB/s]
model.bin: 0% 0.00/3.09G [00:00<?, ?B/s]
tokenizer.json: 100% 2.48M/2.48M [00:00<00:00, 12.5MB/s]
model.bin: 6% 199M/3.09G [00:01<00:17, 162MB/s]
```