Absolutely! Here's a comprehensive README.md file for your AuraFy project. It covers all the crucial setup and deployment steps we went through, so you have a handy reference and don't forget any details.

Markdown

# AuraFy - Find Your Aura! ✨

AuraFy is a fun, interactive web application that helps users discover their "aura" based on their generational knowledge and perspectives. It generates personalized multiple-choice quizzes using a powerful AI model (Google Gemini) and provides an "aura score" and a fun fact at the end.

## Table of Contents

- [Features](#features)

- [Live Demos](#live-demos)

- [Project Structure](#project-structure)

- [Setup and Deployment](#setup-and-deployment)

- [Prerequisites](#prerequisites)

- [1. Clone the Repository](#1-clone-the-repository)

- [2. Backend Setup (Flask & Render)](#2-backend-setup-flask--render)

- [3. Frontend Setup (HTML/CSS/JS & Netlify)](#3-frontend-setup-htmlcssjs--netlify)

- [Troubleshooting Tips](#troubleshooting-tips)

- [Credits](#credits)

- [License](#license)

## Features

- \*\*Personalized Quizzes:\*\* Questions are tailored to the user's selected generation (Gen Alpha, Gen Z, Millennial, Gen X, Baby Boomer, Silent Generation).

- \*\*Dynamic Content:\*\* AI-powered question generation using Google Gemini.

- \*\*Interactive UI:\*\* Smooth page transitions, option selection, and navigation.

- \*\*Aura Score:\*\* Calculates a score based on quiz responses.

- \*\*Fun Facts:\*\* Provides an interesting fact about the selected generation.

- \*\*Responsive Design:\*\* (Assumed, based on typical frontend practices)

## Live Demos

- \*\*Frontend (Netlify):\*\* [https://aurafy.netlify.app/](https://aurafy.netlify.app/)

- \*\*Backend (Render):\*\* [https://aurafy-backend-denden.onrender.com](https://aurafy-backend-denden.onrender.com) (This is the API endpoint, not a UI)

## Project Structure

Your project is structured into two main parts: `AuraFy\_Frontend` for the web interface and `aura-fy-backend` for the API.

AuraFy/ ├── AuraFy\_Frontend/ │ ├── static/ │ │ ├── css/ # Your CSS files (e.g., home-page-style.css, quiz-backgrounds.css) │ │ ├── images/ # Your image assets │ │ └── js/ # Your JavaScript files (e.g., script.js, quiz-script.js) │ ├── templates/ # Your HTML pages (e.g., index.html, quiz.html, result.html) │ └── netlify.toml # Netlify configuration for deployment ├── aura-fy-backend/ │ ├── app.py # Flask application with API endpoints │ ├── .env.example # Example for environment variables │ ├── requirements.txt # Python dependencies for the backend │ └── Procfile # (Render-specific for Gunicorn) └── README.md # This file

## Setup and Deployment

Follow these steps to set up the project locally and deploy it to Netlify (Frontend) and Render (Backend).

### Prerequisites

Before you start, make sure you have the following:

- \*\*Python 3.8+\*\* installed

- \*\*Node.js & npm\*\* (or yarn) installed (for frontend development if you choose to use local server, not strictly required for Netlify deploy)

- \*\*Git\*\* installed

- A \*\*GitHub account\*\*

- A \*\*Netlify account\*\*

- A \*\*Render account\*\*

- A \*\*Google Cloud Project\*\* with the \*\*Generative Language API\*\* enabled, and an \*\*API Key\*\* generated for the Gemini model.

### 1. Clone the Repository

First, clone your project from GitHub to your local machine:

```bash

git clone [https://github.com/Rohit121-code/AuraFy.git](https://github.com/Rohit121-code/AuraFy.git)

cd AuraFy

**2. Backend Setup (Flask & Render)**

This section covers setting up your Flask API and deploying it to Render.

**2.1. Local Backend Setup (Optional, for development)**

1. **Navigate to the backend directory:**

Bash

cd aura-fy-backend

1. **Create a Virtual Environment (Recommended):**

Bash

python -m venv venv

1. **Activate the Virtual Environment:**
   * On Windows: .\venv\Scripts\activate
   * On macOS/Linux: source venv/bin/activate
2. **Install Python Dependencies:**

Bash

pip install -r requirements.txt

1. **Create .env file:** Create a file named .env in the aura-fy-backend directory and add your Google Gemini API key:
2. GEMINI\_API\_KEY="YOUR\_GOOGLE\_GEMINI\_API\_KEY"

Replace "YOUR\_GOOGLE\_GEMINI\_API\_KEY" with your actual API key.

1. **Run the Flask App Locally:**

Bash

python app.py

The API should be running on http://127.0.0.1:5000.

**2.2. Deploy to Render**

1. **Log in to Render:** Go to <https://render.com/> and log in.
2. **Create a New Web Service:**
   * Click "New" -> "Web Service".
   * Connect to your GitHub repository where aura-fy-backend resides (e.g., Rohit121-code/AuraFy).
   * **Root Directory:** Set this to aura-fy-backend.
   * **Branch:** master (or main)
   * **Name:** aurafy-backend (or your preferred name)
   * **Region:** Choose a region close to your users.
   * **Runtime:** Python 3
   * **Build Command:** pip install -r requirements.txt
   * **Start Command:** gunicorn app:app
   * **Plan Type:** Choose "Free" for testing, or a paid plan for production.
3. **Add Environment Variable:**
   * Before deploying, go to the "Environment" tab for your new service.
   * Add a new environment variable:
     + **Key:** GEMINI\_API\_KEY
     + **Value:** Your actual Google Gemini API key.
   * Click "Add Environment Variable".
4. **Deploy:** Click "Create Web Service". Render will automatically build and deploy your backend. Monitor the logs for any errors.
5. **Note the Live URL:** Once deployed, Render will provide a public URL for your backend (e.g., https://aurafy-backend-denden.onrender.com). Make a note of this.

**3. Frontend Setup (HTML/CSS/JS & Netlify)**

This section covers configuring your frontend for deployment with Netlify.

**3.1. Verify netlify.toml**

Ensure your AuraFy\_Frontend/netlify.toml file has the following content. This tells Netlify how to build and serve your site.

Ini, TOML

[build]

# IMPORTANT: In Netlify UI, set 'Base directory' to 'AuraFy\_Frontend'.

# This 'publish' then means the root of that 'Base directory'.

publish = "." # The root of your AuraFy\_Frontend directory

command = "echo No build command" # No build command is needed for static HTML/CSS/JS

# Redirects are crucial for your HTML files which are in 'templates'

# This ensures paths like / or /quiz get mapped correctly.

[[redirects]]

from = "/"

to = "/templates/index.html"

status = 200

[[redirects]]

from = "/quiz"

to = "/templates/quiz.html"

status = 200

[[redirects]]

from = "/user-info"

to = "/templates/user-info.html"

status = 200

[[redirects]]

from = "/result"

to = "/templates/result.html"

status = 200

[[redirects]]

from = "/exit"

to = "/templates/exit.html"

status = 200

**3.2. Update Frontend Paths**

You **must** ensure all paths in your HTML and JavaScript files are correct for deployment.

1. **Asset Paths (CSS, JS, Images) in HTML:** In AuraFy\_Frontend/templates/\*.html (e.g., index.html, quiz.html), ensure all links to CSS, JS, and image files use **absolute paths** starting with /static/.

HTML

<link rel="stylesheet" href="/static/css/home-page-style.css">

<script src="/static/js/script.js"></script>

<img src="/static/images/logo.png">

1. **Navigation Paths in HTML & JavaScript:** Ensure navigation between pages uses the **root-relative paths** as defined in netlify.toml.
   * In index.html (for the Exit button):

HTML

<button class="exit-button" onclick="fadeOutAndNavigate('/exit')">Exit</button>

* + In AuraFy\_Frontend/static/js/script.js (for the Start button):

JavaScript

startButton.addEventListener('click', () => {

fadeOutAndNavigate('/user-info');

});

* + In AuraFy\_Frontend/static/js/quiz-script.js (for Home and Result navigation):

JavaScript

// For Home button

homeButton.addEventListener('click', () => {

fadeOutAndNavigate('/');

});

// For Quiz completion

fadeOutAndNavigate('/result');

1. **API Endpoint URL in quiz-script.js:** This is crucial. In AuraFy\_Frontend/static/js/quiz-script.js, ensure your fetch call targets the full URL of your deployed Render backend and the correct endpoint (/generate\_quiz\_questions).

JavaScript

// In AuraFy\_Frontend/static/js/quiz-script.js

const fetchQuestionsFromAI = async () => {

try {

// Use your Render backend URL here!

const response = await fetch('[https://aurafy-backend-denden.onrender.com/generate\_quiz\_questions](https://aurafy-backend-denden.onrender.com/generate\_quiz\_questions)', {

method: 'POST',

headers: {

'Content-Type': 'application/json',

},

body: JSON.stringify({ generation: userGeneration }),

});

// ...

} catch (error) {

// ...

}

};

**3.3. Deploy to Netlify**

1. **Commit and Push Changes:** After making all the necessary path and netlify.toml adjustments, commit and push your entire project to your GitHub repository:

Bash

git add .

git commit -m "Configure for Netlify & Render deployment, fix all paths"

git push origin master # or main

1. **Log in to Netlify:** Go to <https://app.netlify.com/> and log in.
2. **Import a New Site from Git:**
   * Click "Add new site" -> "Import an existing project".
   * Connect to your GitHub repository (e.g., Rohit121-code/AuraFy).
   * **Owner:** Select your GitHub user/organization.
   * **Repository:** Select AuraFy.
   * **Branch to deploy:** master (or main)
   * **Base directory:** AuraFy\_Frontend (This is critical!)
   * **Build command:** echo No build command
   * **Publish directory:** .
   * Click "Deploy site".
3. **Monitor Deployment:** Netlify will build and deploy your site. You can monitor the progress in the deploy logs.
4. **Access Your Site:** Once deployed, Netlify will provide a live URL (e.g., https://aurafy.netlify.app/).

**Troubleshooting Tips**

* **"404 Not Found" for assets (CSS, JS, Images) on Netlify:**
  + Ensure netlify.toml is in AuraFy\_Frontend/.
  + Check Base directory in Netlify settings is AuraFy\_Frontend.
  + Verify asset paths in your HTML/JS are absolute (e.g., /static/css/style.css).
* **"404 Not Found" for navigation (e.g., /quiz not loading):**
  + Double-check the [[redirects]] in your netlify.toml.
  + Ensure navigation calls in script.js and index.html use the correct root-relative paths (e.g., /quiz).
* **CORS Errors (e.g., OPTIONS / HTTP/1.1" 404 in backend logs, or "Access to fetch from origin..." in browser console):**
  + **Most common:** The origins in CORS(app, resources={r"/\*": {"origins": ["https://aurafy.netlify.app"]}}) in your app.py must **exactly** match the protocol (https://) and domain of your Netlify frontend.
  + Make sure your backend service on Render is fully redeployed after any app.py changes.
* **Backend Errors (e.g., 500 status from API, "Failed to generate questions" message):**
  + **Check Render logs:** Go to your Render dashboard for aurafy-backend and check the "Logs" tab for detailed Python tracebacks.
  + **API Key:** Verify GEMINI\_API\_KEY is correctly set as an environment variable in Render for your backend service.
  + **API Endpoint URL:** Confirm the fetch URL in your quiz-script.js is the exact, correct Render API URL (e.g., https://aurafy-backend-denden.onrender.com/generate\_quiz\_questions).

**Credits**

* **Frontend:** Built with HTML, CSS, JavaScript.
* **Backend:** Developed with Flask and Python.
* **AI Model:** Powered by Google Gemini (Generative Language API).
* **Deployment:** Hosted on Netlify (Frontend) and Render (Backend).

**License**

(Add your desired license information here, e.g., MIT License)