#### ****Motivation****

Here’s a **Project Specification Document** for your Rubik's Cube Learning website based on the given technical requirements.

### ****CleanCube - Project Specification Document****

#### ****Rubik's Cube Learning Website****

### ****1. Project Overview****

The **Rubik's Cube Learning Website** aims to teach users how to solve the Rubik's Cube interactively. Users can navigate through different solution stages, view corresponding algorithms, and practice solving the cube via an embedded simulator. The Website will feature the ability to manage solution stages, and integrate with OpenAI-API for enhanced functionality.

### ****2. Target Audience****

1. Beginners learning to solve the Rubik's Cube.
2. Intermediate solvers looking to refine their skills.
3. Cube enthusiasts interested in a user-friendly, interactive platform.

**Top Goal – One should be able to solve a cube within one day using the website, with no prior knowledge.**

### ****3. Features & Functionality****

#### ****3.1. Core Functionalities****

* **Interactive Rubik's Cube Simulator:** Embed an iframe to allow users to practice cube-solving virtually.
* **Learning Stages:** Step-by-step instructions for solving the cube, categorized into stages (e.g., Cross, Corners).
* **Algorithm Display:** Show specific algorithms for each stage with detailed explanations.
* **Advanced Cubes:** Suggesting other cubes for intermediate users, allowing for continuous learning experience.
* **Store:** an online shop for several cubes.

#### ****3.2. Responsive Design****

* Ensure the Website is fully functional on both desktop and mobile devices.
* Implement a responsive layout using CSS Media Queries and Flexbox/Grid.

#### ****3.3. External API Integration****

* **Feedback Feature:** Use an external API – OpenAI.
* **Tutorial API:** Optionally, fetch additional cube-solving tutorials or images from public API.

### ****4. Technical Requirements****

#### ****4.1. Technology Stack****

* **HTML, CSS, JavaScript:** Core technologies for structure, styling, and interactivity.
* **JSON File (Optional):** Use a static JSON file for pre-loaded stage data if needed.
* **Iframe Integration:** Embed the external Rubik's Cube simulator.

#### ****4.3. API Requirements****

* **Endpoint for Feedback Submission:** Example endpoint for sending user feedback using

### ****5. User Interface Design****

#### ****5.1. Layout (Desktop / Mobile)****

* **Left / Middle Section:** List of solution stages (dynamic, populated from localStorage).
* **Middle / Top Section:** Embedded iframe for the Rubik's Cube simulator.
* **Right / Bottom Section:** Details of the selected stage, including the algorithm.

#### ****5.2. Interaction Flow****

1. User is guided to align the 3D infective cube to 3 specific points, allowing for an AI process to analyze its state, and come up with a specific solution to it.
2. Full algorithm is generated and nested in the left section.
3. Corresponding algorithm of the current stage appears in the right column.
4. User interacts with the Rubik's Cube simulator in the middle column, in order to solve the cube.

#### ****5.3. Responsive Design****

* Mobile-first approach with a vertically stacked layout.
* Desktop layout with three columns (flexible width).

### ****6. Development Workflow****

#### ****6.1. Milestones****

1. **Design Phase:**
   * Create wireframes for desktop and mobile layouts.
2. **Development Phase:**
   * Build the HTML, CSS, and JavaScript structure.
   * Build a Node server and deploy it in Render.
3. **Testing Phase:**
   * Test responsiveness on various devices manually.
4. **Deployment:**
   * Host the Website using Netlify.

#### ****6.2. Tools****

* **Code Editor:** Visual Studio Code.
* **Version Control:** Git/GitHub for code management.
* **Testing:** Browser Developer Tools for responsive and functionality testing.

### ****7. Data Management Plan****

#### ****7.1. Data Schema****

{

"stages": [

{

"id": 1,

"name": "Cross",

"algorithm": "F R U R' U' F'"

},

{

"id": 2,

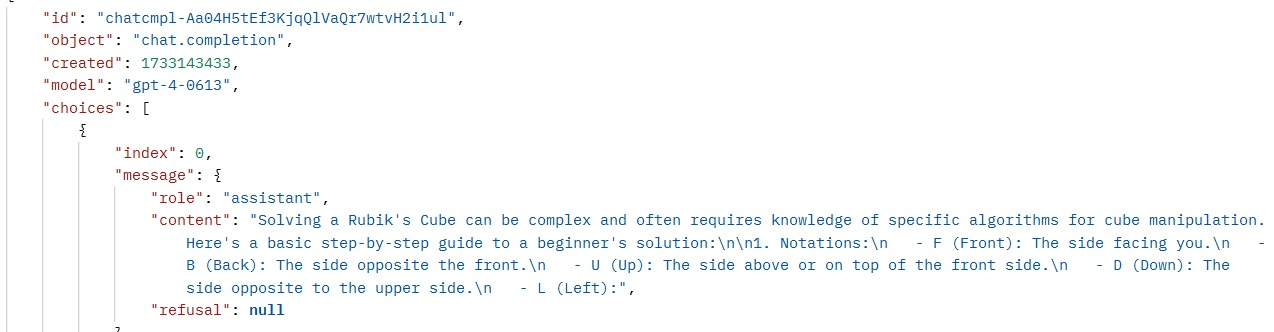
"name": "Corners",

"algorithm": "U R U' L' U R' U' L"

}

]

}



### ****8. Future Enhancements****

1. **Back-End Integration:** Allow for cube-state cache for continuing from last solving stage.
2. **Gamification:** Introduce a scoring system for completing cube-solving stages.

### ****9. Project Timeline****

| ****Phase**** | ****Duration**** | ****Tasks**** |
| --- | --- | --- |
| Design Phase | 1 day | Wireframing and layout design. |
| Development Phase | 2 days | Implementing functionality, API, UI. |
| Testing & Deployment | 0.5 day | Final testing and deployment. |

### ****10. Conclusion****

This specification serves as a roadmap for building the Rubik's Cube Learning Website. It focuses on providing an interactive and responsive user experience while meeting the outlined technical requirements.

**Next Steps:** Review and finalize specifications before beginning the design phase.