**Polling App Readme File**

**Description**

A full-stack polling web application with the following features:

* **Frontend:** React.js-based user interface.
* **Backend:** Node.js-based API server.
* **Database:** MongoDB for data storage.

All services are containerized using Docker and orchestrated with Docker Compose.

**Features**

* Poll creation and voting.
* User-friendly interface built with React.
* RESTful API to handle backend logic.
* MongoDB integration for persistent data storage.
* Scalable and deployable using Docker Compose.

**Prerequisites**

* [**Docker**](https://www.docker.com/)
* [**Docker Compose**](https://docs.docker.com/compose/)

**Project Structure**

**Polling/ //**Main Directory

**backend/ //Sub Directory**

Dockerfile //File

package.json //File

server.js //File

**├── frontend/ //**Sub Directory

Dockerfile //File

package.json //File

**public/ //**Sub Directory

index.html //File

**src/ //**Sub Directory

index.js //File

App.js //File

**├── mongo/ //**Dirrectory

Dockerfile //File

**├── docker-compose.yml //**Compose File

**└── README.md //**Readme File

**Installation and Setup**

1. Clone the repository:
2. git clone https://github.com/your-username/polling.git
3. cd polling
4. Build and run the application using Docker Compose:
5. docker-compose up --build
6. Access the application:
   * Frontend: [http://localhost:80](http://localhost/)
   * Backend: [http://localhost:5000](http://localhost:5000/)
   * MongoDB: Accessible on localhost:27017 (default port)

**Application Workflow**

1. **Frontend:**
   * Users interact with the React-based interface to create and vote on polls.
   * The frontend communicates with the backend API to fetch and send data.
2. **Backend:**
   * The Node.js server handles RESTful API endpoints to manage polls.
   * It performs CRUD operations and communicates with the MongoDB database.
3. **Database (MongoDB):**
   * Stores all poll data, including questions, options, and votes.
   * Ensures data persistence across application restarts using Docker volumes.
4. **Docker Compose:**
   * Orchestrates the frontend, backend, and database services.
   * Provides a seamless deployment environment with minimal configuration.

**Docker Overview**

**Frontend**

* Base Image: node:14-alpine
* Build Command: npm run build
* Port: 80

**Backend**

* Base Image: node:14-alpine
* Start Command: node server.js
* Port: 5000

**Database (MongoDB)**

* Image: mongo:6
* Data Storage: Volume mounted to mongo/data
* Port: 27017

**Known Issues**

* Ensure the src/ folder contains all necessary files, including index.js, index.css, and reportWebVitals.js.
* The public/ folder should include index.html.

**License**

This project is licensed under the MIT License.

**Contributing**

Feel free to open issues or create pull requests. Contributions are welcome!

**Author**

**Umair Abbas**

[**GitHub Profile**](https://github.com/your-username)