Cryptocurrency Backend Assignment

Introduction: Design and implement an application that provides information about cryptocurrency. The goal is to create a functional backend system that retrieves and updates cryptocurrency data.

External Components:

1. **AirTable:** https://airtable.com/developers/web/api/introduction
2. **CoinGecko API:** https://www.coingecko.com/en/api. An API that provides cryptocurrency-related data, including coin details and prices.

Functionality Oriented Main Components:

1. **Cache:** An in-memory data structure used to store and retrieve frequently accessed data.
2. **Rate Limiting:** Ensuring that requests to CoinGecko API are within the allowed rate limits to prevent excessive usage.

Functionalities:

1. **Background Jobs:**
    1. **Coin Details Update:** i. Every 10 minutes, retrieve the list of top 20 coins from the **/coins/list** endpoint of the CoinGecko API. ii. Update the coin details for these top 20 coins in the AirTable database.
    2. **Current Coin Price Update:** i. Every 1 minute, retrieve the current coin prices using the **/simple/price** endpoint of the CoinGecko API. ii. Update the current coin prices for each coin ID in both the cache and the AirTable database.
2. **Exposed APIs:**
    1. **GET /coins:** i. This API endpoint should retrieve coins and their information from the AirTable database. ii. The response should include details like coin name, symbol, market cap, and more.
    2. **GET /coins/price/:coinId:i.**
        1. This API endpoint should accept a parameter coinId representing the ID of the coin.
        2. Retrieve the current price of the specified coin using the cache.
        3. If the price is not available in the cache, fetch it from the AirTable.

Instructions:

1. Set up an AirTable account and create a table to store cryptocurrency information, including coin details and prices.
2. Understand the CoinGecko API documentation to make requests to the specified endpoints.
3. Implement a simple rate limiting mechanism to ensure that requests to the CoinGecko API are within the allowed limits.
4. Create a cache data structure in your backend application to store frequently accessed data, such as current coin prices.
5. Implement the background jobs to periodically update coin details and current prices according to the specified intervals.
6. Test your application thoroughly to ensure that the background jobs, cache updates, and API endpoints are working as expected.
7. Provide clear documentation on how to set up and run your application, including any necessary environment variables, dependencies, and instructions.
8. Remember to keep your code clean, well-structured, and properly commented.

Submission Guidelines

1. Create a .zip file of all application related files and send an email.
2. Ensure that a README file is present in the base folder describing steps for any local steps required.
3. Create a demo video and add the link to the video in GitHub readme.
