Senior Golang Engineer Assessment

Problem Description:

You are tasked with creating a Go application that manages a simple employee database or in-memory store. Additionally, you need to implement a RESTful API with pagination for listing employee records.

Requirements:

1. Employee Struct:

- Define a struct named `Employee` with the following fields:

- `ID` (int): Unique identifier for the employee.

- `Name` (string): Name of the employee.

- `Position` (string): Position/title of the employee.

- `Salary` (float64): Salary of the employee.

2. CRUD Operations:

- Implement functions/methods to perform CRUD operations on the employee database or in-memory store:

- `CreateEmployee`: Adds a new employee to the database or store.

- `GetEmployeeByID`: Retrieves an employee from the database or store by ID.

- `UpdateEmployee`: Updates the details of an existing employee.

- `DeleteEmployee`: Deletes an employee from the database or store by ID.

3. Concurrency:

- Ensure that the application is safe for concurrent use by using appropriate synchronization mechanisms.

4. Testing:

- Write unit tests to cover the CRUD operations and ensure the correctness of the implementation.

5. RESTful API with Pagination:

- Implement a RESTful API for listing employee records with pagination.

- The API should provide endpoints for listing employees with support for pagination.

- Each page should contain a configurable number of records.

- Implement proper error handling and response formatting for the API endpoints.

-Evaluation Criteria:

- Accuracy and completeness of implementation of all requirements.

- Proper usage of Go concurrency primitives for safe concurrent access.

- Adequate test coverage and correctness of unit tests.

- Efficiency and performance of CRUD operations.

- Correct implementation of the RESTful API with pagination.

- Clarity, readability, and organization of code.

Submission:

Candidates are required to complete the quiz by coding the solution in Go and providing the code in a GitHub repository or any other preferred version control system. Additionally, candidates should include:

- A video demonstration showcasing the functionality of the application, including CRUD operations and API testing using tools like Postman. **(Good to have)**