PRACTICAL TASK

General Instructions:

- API response should be in Json format with appropriate status code.
- Use the status code 200 series for successful data insert or successful partial data insert.
- User the status code 300 series for wrong data insert, incorrect data insert or validation errors.
- Example 200 and 300 status codes:

```
{
   "code": 200,
   "msg": "city name inserted "
}
{
   "code": 300,
   "msg": "city name contains numeric characters"
}
```

Primary Task Starts

Step 1: POST API with City Route

- Post API should have only 1 (one) parameter city.
- Insert city name in the city table (model).
- City name should be unique.
- City name should allow only Alphabets in upper/ small case.
- City name should not allow numeric/ special characters.
- Once city name is inserted, provide the response via status codes.

Step 2: GET API with City List Route

- Provide the List of Cities available from Database in Json object.

Step 3: Consume 3rd Party API

- Run this API https://api.binance.com/api/v1/time
- Console the Output in terminal

Step 4: POST API with User Route

- Post API should have 2 (three) mandatory parameters name and city.
- Post API should have 2 (two) optional parameters mobile and media url.
- Post API should have 1(one) invisible (server side) parameter ID.
- ID parameter should take the value of Step 3, Output servertime.
- Name parameter should allow only Alphabets in upper/ small case.
- Name parameter should not allow numeric/ special characters.
- City parameter should accept only city names available in database.
- Mobile parameter should allow only numeric characters.
- Media url parameter should allow string with https:// only.
- Once the user details have been inserted, provide the response via status codes along with inserted user details and ID.

Note: ID must not be passed via API. It should be taken by Node.js script whenever the data is inserted.

Step 5: GET API with User List Route

- Provide the List of Users available from Database in Json object along with ID.

Step 6: PATCH/ POST API with User Modify Route

- PATCH/ POST API should allow modification of user data based on unique variable of ID.
- PATCH/ POST API should allow modification of name, city, mobile and media url parameters.
- Once the user details have been modified, provide the response via status codes along with modified user details.

Note: ID should not be updated in any case.

Primary Task Ends

Bonus (Optional) Task Starts

Bonus Step 1: EJs Display

- Display the user data from API using EJs
- Allow user data modification via API using EJs

Bonus (Optional) Task Ends

===== xxx ===== xxx =====