TODO App -

Statement for Candidates:

Below is the problem statement for your project.

1. Your task is to create an **API** that supports project requirements.
2. You have 3 hours to develop the API.
3. You are free to use google to find solutions, however, copying an existing project is not allowed.
4. At the end of the 3 hour time period, you will be given get 15 minutes to do your submission on Google Form & to push your project in a GitHub repository, submission form: https://forms.gle/PqfNAakDWgz146BJ8
5. Your API submission will be evaluated by our team and a result will be published at 11:30 AM on the same day.
6. No need to create the session, authenticate using user_id
7. No need to apply JWT or any token for authentication, simply use user_id to authenticate

Recorded Voice**:** Good morning, Agent X! IMF is on a mission to sabotage a terrorist activity most likely involving the *Syndicate* at the heart of Silicon Valley.

Mr. Hunt needs your help on this front. We believe that the roots of the terrorist organization have penetrated every level of all the major tech giants and no one else can be trusted.

Your mission, should you choose to accept it is to design a web-app to efficiently manage tasks. A TODO list application that would help Mr. Hunt and other agents.

Requirements:

1. Agent account registration:
   Provide the ability to create a new agent. This agent_id and password will be used to login to the panel.

   ```
   [POST] /app/agent

   Request Data: {
       'agent_id': str,
       'password': str
   }

   Response Data: {
       'status': 'account created',
       'status_code': 200
   }
   ```

   Mandatory Requirement: All the agent passwords stored in the database have to be encrypted.
   No one should be able to read the stored items without a secret key. The key can be stored in the code itself. You know why it's necessary!

2. Agent account login:
   Provide the ability to log into the panel using agent_id & password.

```
[POST] /app/agent/auth

Request Data: {
    'agent_id': int,
    'password': str
}

For success,
Response Data: {
    'status': 'success',
    'agent_id': int,
    'status_code': 200
}

For failure,
Response Data: {
    'status': 'failure',
    'status_code': 401
}
```

3. Task/TODO list:
   After authenticating the agent, the agent can see a list of stored TODO items

   ```
   [GET] /app/sites/list/?agent={agentId}

   Request Data: None
   Response Data: [list of TODO items with their details]
   ```

   a. A list of TODO items returned should only belong to the 'agent_id' provided in the request. We don't want anyone else interfering in Mr. Hunt's business.

   b. Sort TODO items on the basis of their Due-Date. You don't want an agent to miss a deadline where lives are at stake.

4. Save a new TODO item:
   Provide the ability for the agent to add new TODO items using a form.
   Inputs: agent_id, TODO Title, TODO Description, Category, Due Date

```
[POST] /app/sites?agent={agentId}

Request Data: {
    'title': str,
    'description': str,
    'category': str,
    'due_date': date
}

Response Data: {
    'status': 'success',
    'status_code': 200
}
```

Tech Stack:

1. Any web-server of your choice (Python Flask / Django, NodeJS Express / Koa, Java, etc)
2. Database: MySQL (Compulsory)

Good luck, Agent X!

The message will auto-destruct in 5 seconds.