Django Lets Ride - Assignment

Prerequisites:

- Knowledge of Django, Python, SQL.
- Knowledge of installing packages using pip.
- Knowledge of APIs.
- A Github account in order to share your code with us.

Submission guidelines

1. **Before Starting the Assignment:**
    a. Create a public Github repository to work on this assignment
2. **After completing the Assignment:**
    a. Share the Loom Video Recording explaining your code and how it works by showing your output in jupyter notebook by explaining each API request and response
        i. Github Link & Loom recording Submission Form*:* https://forms.gle/Uw1DKz6t6g6WwoCv9
        ii. How to record your screen using Loom: https://youtu.be/WSBDNlJCwtg
        iii. We are expecting the backend Apis implementation part only. No need to design UI using any forms/libraries.

Note: Add jupyter notebook in the git repository which you'll be sharing with us

Queries/Issues Form: https://forms.gle/MzhDx6CoDFeXqBpu8

Project Description

There will be 2 persons one is a **Rider** and the other is an **Asset Transportation Requester** (will be referred as **requester** from now on)**.**
A **Rider** is a person who travels from one place to another and is willing to carry some assets(packages/luggages) along with him.
A **Requester** is a person who wants his assets to be carried by someone else from one place to another.
Requesters can create transportation requests and **Riders** can share their rides independently .

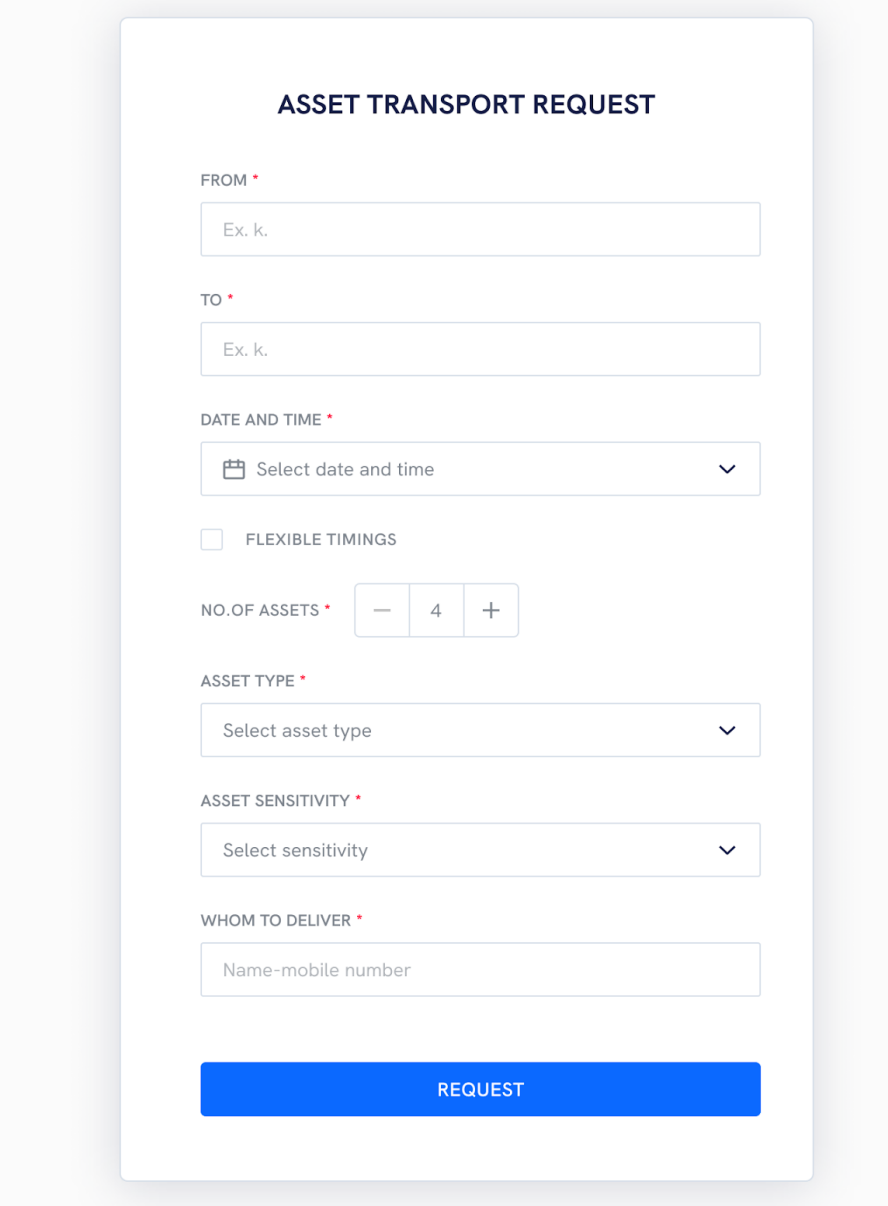
Features

1. A **Rider** can share his travel info with details like **from** and **to locations**, the **number of assets** he can take with him etc.
2. **Requesters** can request to carry their assets, with details like **from** and **to locations**, the **type of assets, number of assets** that need to be carried.

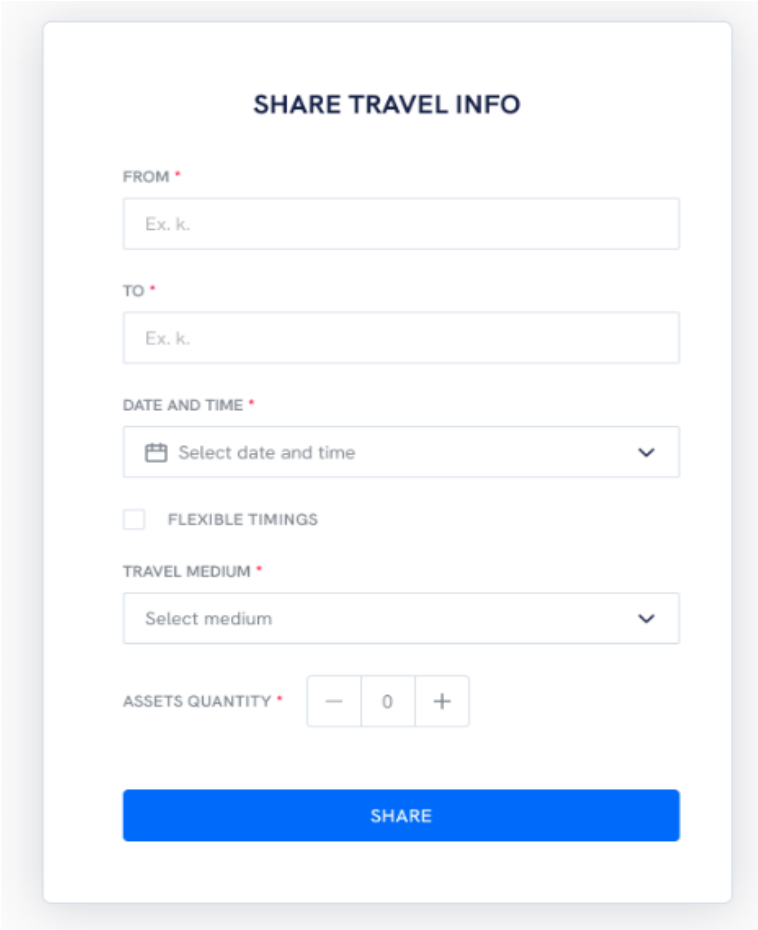
3. A **Requester** should be able to see all the asset transportation requests requested by him.
4. A **Requester** should be able to see all the matching travel info shared by **Riders** based on his asset transportation requests locations.
5. A **Requester** can **apply** to carry his assets by a **Rider.**

UI Screens for above features

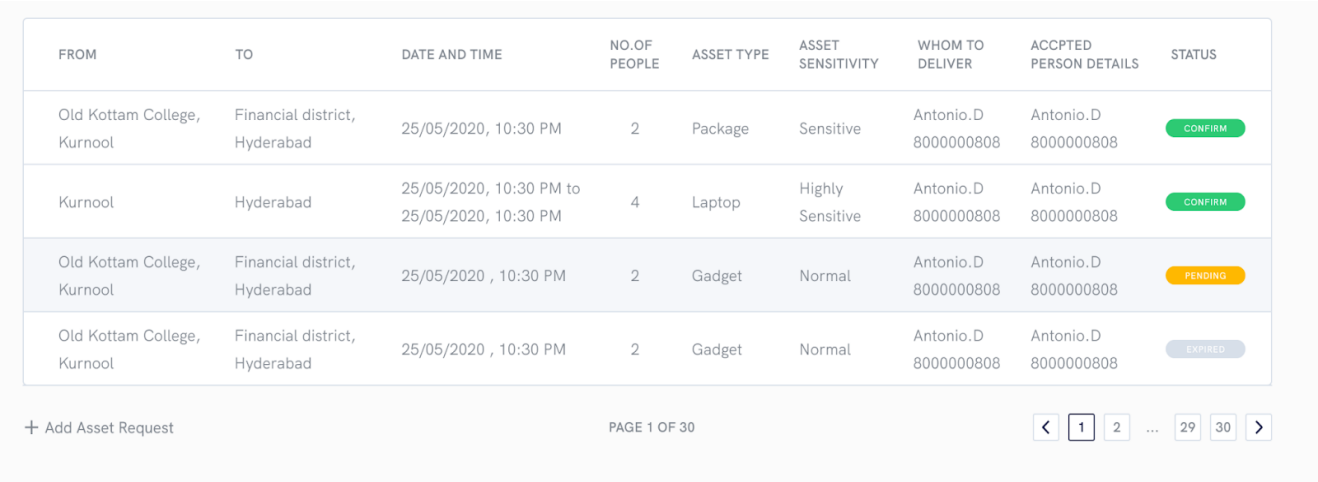
Asset Transportation Request (For Requester)



Share Travel Info (For Rider)



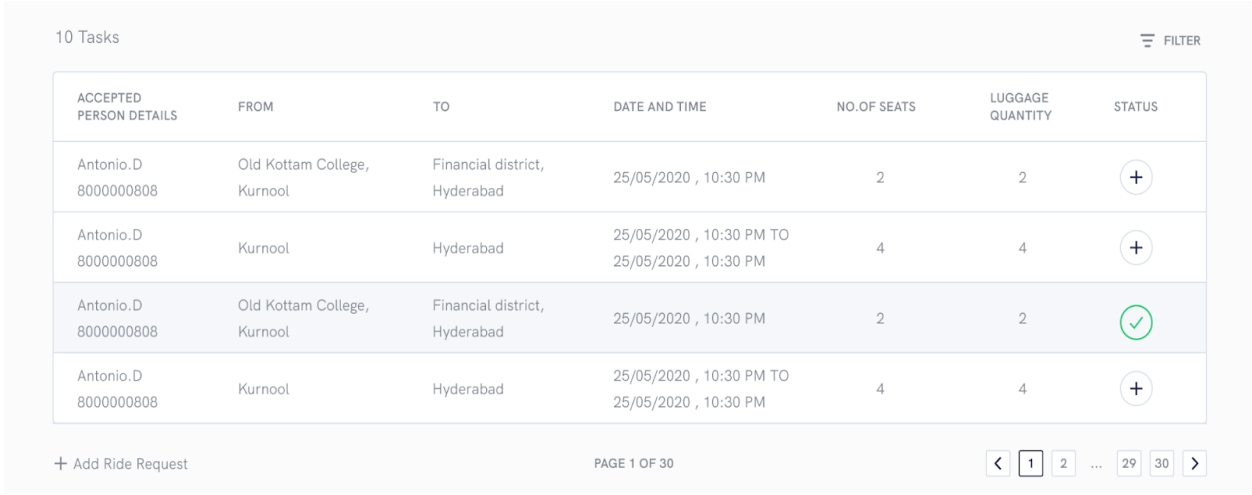
My Asset Transportation Requests (For Requester)



FROM	TO	DATE AND TIME	NO.OF PEOPLE	ASSET TYPE	ASSET SENSITIVITY	WHOM TO DELIVER	ACCPTED PERSON DETAILS	STATUS
Old Kottam College, Kurnool	Financial district, Hyderabad	25/05/2020, 10:30 PM	2	Package	Sensitive	Antonio.D 8000000808	Antonio.D 8000000808	CONFIRM
Kurnool	Hyderabad	25/05/2020, 10:30 PM to 25/05/2020, 10:30 PM	4	Laptop	Highly Sensitive	Antonio.D 8000000808	Antonio.D 8000000808	CONFIRM
Old Kottam College, Kurnool	Financial district, Hyderabad	25/05/2020 , 10:30 PM	2	Gadget	Normal	Antonio.D 8000000808	Antonio.D 8000000808	PENDING
Old Kottam College, Kurnool	Financial district, Hyderabad	25/05/2020 , 10:30 PM	2	Gadget	Normal	Antonio.D 8000000808	Antonio.D 8000000808	EXPIRED

Note: In the above screen treat **No OF PEOPLE** column as **NO OF ASSETS**

Matched Asset Transportation Requests (For Requester)



ACCEPTED PERSON DETAILS	FROM	TO	DATE AND TIME	NO.OF SEATS	LUGGAGE QUANTITY	STATUS
Antonio.D 8000000808	Old Kottam College, Kurnool	Financial district, Hyderabad	25/05/2020 , 10:30 PM	2	2	+
Antonio.D 8000000808	Kurnool	Hyderabad	25/05/2020 , 10:30 PM TO 25/05/2020 , 10:30 PM	4	4	+
Antonio.D 8000000808	Old Kottam College, Kurnool	Financial district, Hyderabad	25/05/2020 , 10:30 PM	2	2	✓
Antonio.D 8000000808	Kurnool	Hyderabad	25/05/2020 , 10:30 PM TO 25/05/2020 , 10:30 PM	4	4	+

Note: In the above screen ignore **NO. OF SEATS** column

APIs

1. Write an API to create an asset transport request by using the reference screen mentioned below and return valid response with respective status code (**For Requester**)
    a. Requester should be able to request an asset transport
    b. Valid **asset types** are LAPTOP, TRAVEL_BAG, PACKAGE
    c. Valid **sensitivities** are HIGHLY_SENSITIVE, SENSITIVE, NORMAL

    Ref: **Asset Transportation Request**

2. Write an API to share the travel info of the Rider by using the reference screen mentioned below and return valid response (**For Rider**)
    a. Valid **travel mediums** are BUS, CAR, TRAIN

    Ref: **Share Travel Info**

3. Write an API to get asset transport requests created by the Requester following the reference screen and get the response details as shown in the screen.**(For Requester)** This API should also support
    1. **sorting** on datetime
    2. **filtering** on status and asset type.

    3. Api should support pagination

    If no filter is applied, all requests should be shown by default.

    Note:
    1. Here **NO.OF PEOPLE** in the reference screen means **NO.OF ASSETS**
    2. The initial status of the request should be **Pending**.
    3. When the request's end datetime is completed then the status should be **Expired.**
    4. You can ignore the accepted person details column
    5. Possible statuses are **Expired** and **Pending**. Ignore the **"Confirm"** status shown in the reference image.

    Ref: **My Asset Transportation Requests**

4. Write an API to get the matching rides by **Riders** based on the requests applied by the **Requester**. The API should support pagination.

    Example of matched requests criteria.

    If a requester creates a transportation request from hyderabad to bangalore on a specific date, the matched rides should be of the same locations and date. Multiple rides can be matched to a single request.

    The above criteria applies for all the transportation requests of a **Requester**

    Note:
    1. Here **NO.OF PEOPLE** in the reference screen means **NO.OF ASSETS**
    2. The initial status of the matched Asset Transportation request should be **NOT_APPLIED** & gets changed to **APPLIED** once the requester applies for it

    Ref: **Matched Asset Transportation Rides**

5. As a Requester, I should be able to **apply** for the travel info shared by a **Rider.** Once applied the status should be shown as applied in matched rides api**.**

Common Code Guidelines:

1. Tables which you are creating should be optimal.
2. DB Queries & logic should be optimal.
3. Follow clean code guidelines.

All The Best!