Alembic autogenerate feature :-

11:10:20

- We can make Alembic intelligent enough to figure out what exactly is missing, from our Postgres database, and what needs to be created.

The way Alembic does it, is by looking at the Sqlalchemy model classes.
Alembic will import all of our sqlalchemy models, and based on the columns, it will figure out what our postgres database is supposed to look like.

- We need not delete all the tables and make them from scratch, as Alembic can find the missing columns in table, by comparing it with model class.

To make Alembic figure out the changes automatically, and make necessary changes by looking at the model classes, the command is :-

```
alembic --autogenerate -m "message"
```

- This function will create a new revision (with stated message), and in that, it will automatically put

the code for database changes, in the upgrade() & downgrade() function.

For eg:-
In our case, we manually made the "posts" & "users" table completely, and only votes table creation is left, according to models. Thus, to create a revision that will create the "votes" table as difference, by seeing the models, we execute following :-

[alembic revision --autogenerate -m "auto-votes"
A revision will be created, with the name "auto-votes", that will contain codes to make the exactly same "votes" table.

⇒ Then, we upgrade to this revision as :-
[alembic upgrade head

⇒ At any state of the database, if we have a change in code, we can perform autogeneration.

⇒ Using this, if we make any changes to model class, we can simply implement that, without the need to delete the entire table.

11:13:50

⇒ As we have alembic configured, we will remove the following line from the main.py file :-

[models.Base.metadata.create_all(bind=engine)

This line of code tells the script to run all the create table statements, when the API is started using uvicorn