3. Создание доменной модели

В результате выполнения лабораторной работы должна быть спроектирована модель данных, описывающая выбранную прикладную область..

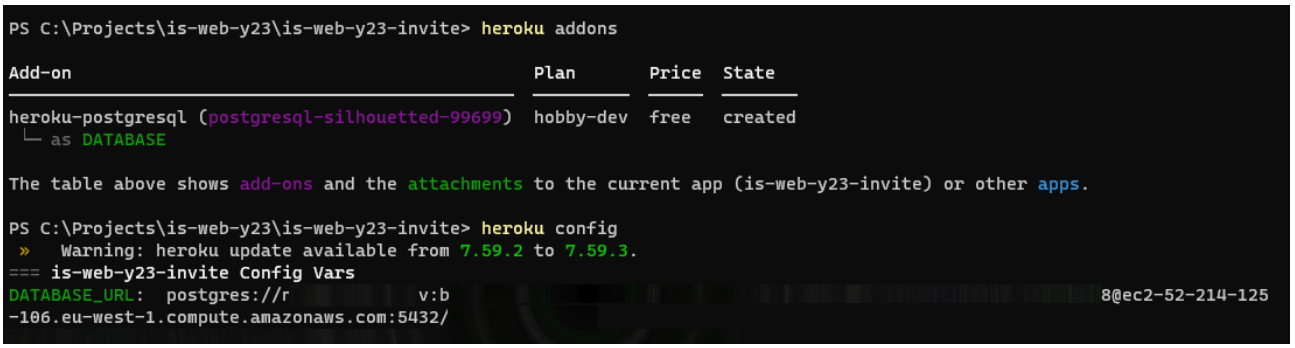
Контрольный срок сдачи работы: 8-ая неделя обучения.

Для выполнения данной лабораторной работы вам необходимо выбрать поставщика данных уровня баз данных. Рекомендуется использовать Heroku Postgres (Database as a Service), но разрешается использовать любую реляционную СУБД.

Для того чтобы подключить Heroku Postgres в качестве СУБД можно воспользоваться Heroku CLI выполнив команду

```
heroku addons:create heroku-postgresql:hobby-dev
```

После выполнения запроса будет создана база данных, а данные для подключения (Connection string) автоматически добавится в качестве переменной среды окружения **DATABASE_URL**



Проверьте что вы можете подключиться к созданной базе данных. Рекомендую использовать инструмент по типу DataGrip. Учтите, что сервера баз данных Heroku требуют SSL подключения, но предоставляют самоподписанный сертификат, не проходящий проверки безопасности, поэтому укажите сообществующую Connection Factory для DataGrip (https://www.jetbrains.com/help/datagrip/how-to-connect-to-heroku-postgres.html)

Подключите коннектор к выбранной СУБД к вашему проекту, если вы выбрали Postgres, то добавьте пакет ‘pg’ как зависимость:

```
npm install pg --save
```

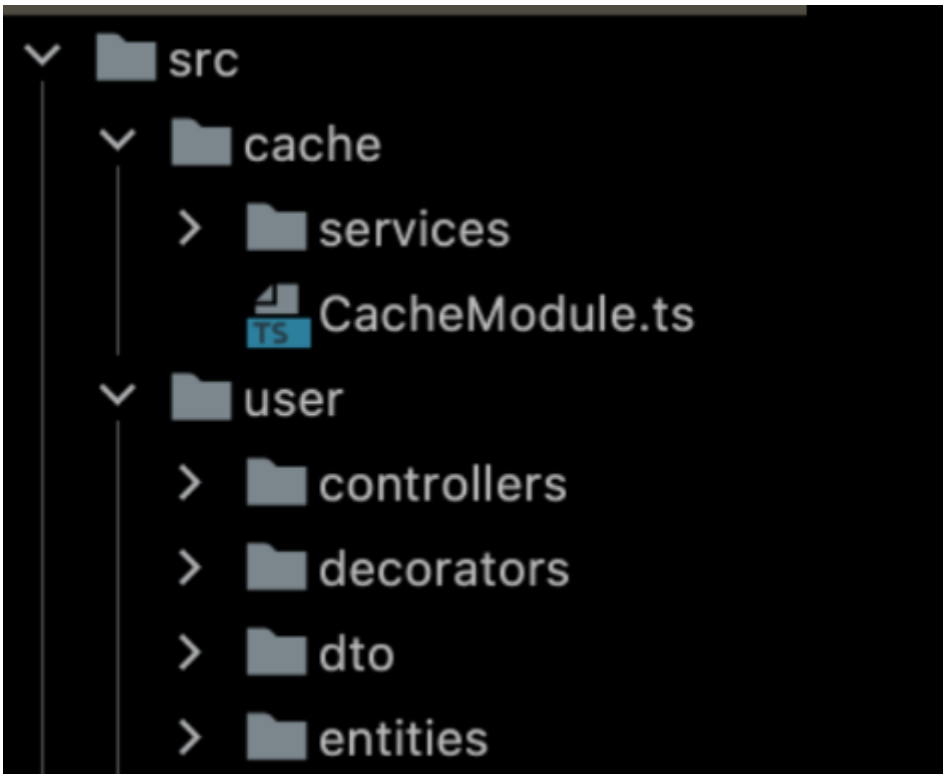
Далее необходимо выбрать одну из поддерживаемых ORM для работы со слоем данных. Рекомендуется либо TypeORM либо Prisma.

В случае, если вы используете TypeORM, вам понадобится распарсить connectionString к базе данных и предоставить её ORM'ке в том виде, как это описано в документации, в виде отдельных полей - Host, username, password и т.д. Для решения этой задачи рекомендую воспользоваться npm пакетом `pg-connection-string`. Ознакомьтесь с разделом Async Configuration и реализуйте сервис инфраструктурного слоя DDD для создания объекта необходимого ORM для подключения.

В случае, если вы используете Prisma, то connection string можно использовать as-is. В качестве инфраструктурного слоя необходимо создать файл schema.prisma и добавить в него указания поставщика и названия ключа переменной окружения - **env(DATABASE_URL)**

После подключения ORM, необходимо описать все модели необходимые для работы вашего проекта. Рекомендуется описывать сущности согласно DDD с использованием общего языка выбранной предметной области.

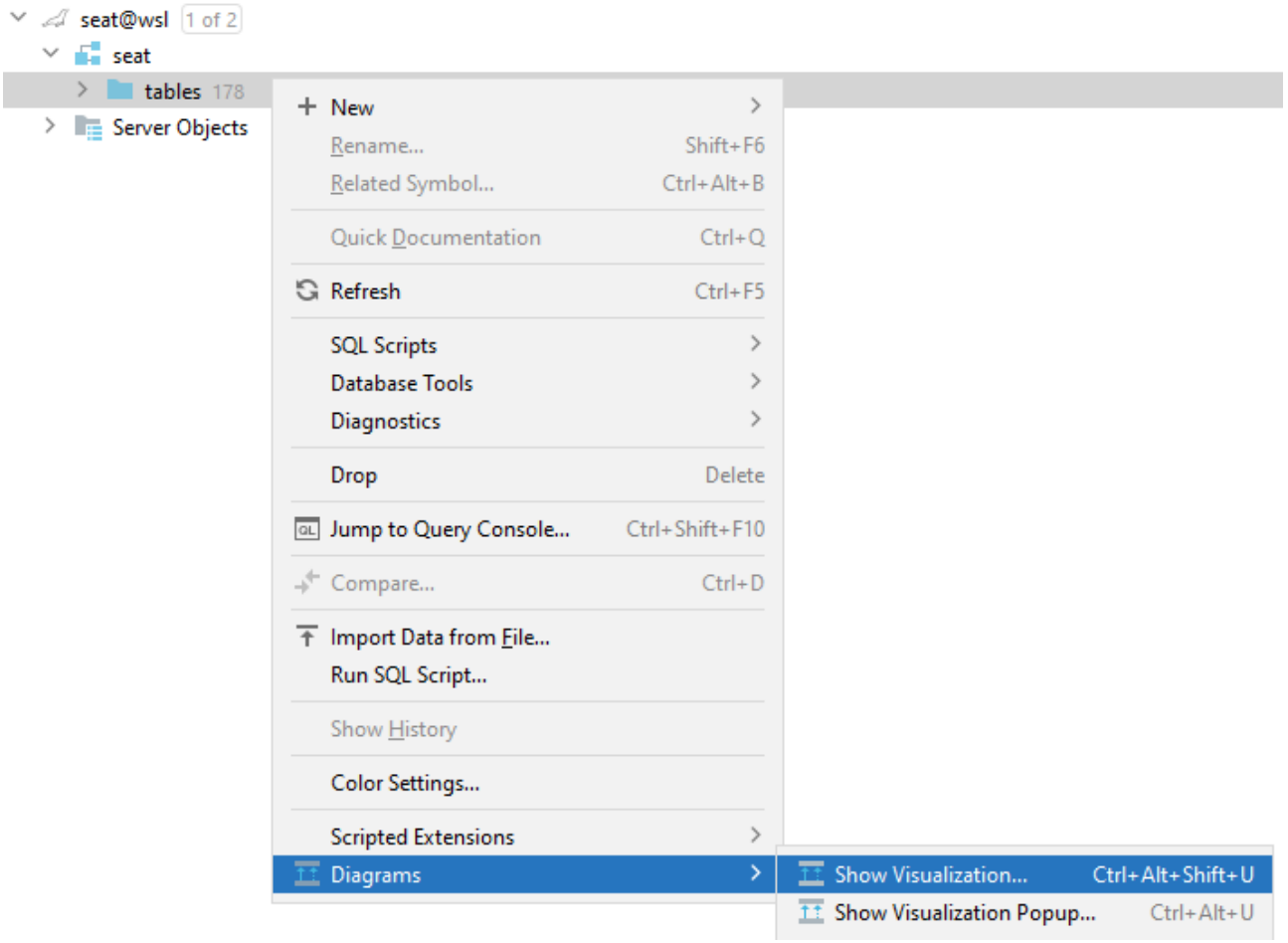
Например, на примере UserModule из лекционного материала, если вы создаёте модуль для работы с пользователем, то выделяйте сущности по смыслу в отдельные директории (src/%moduleName%/entities/%entityName%.ts)

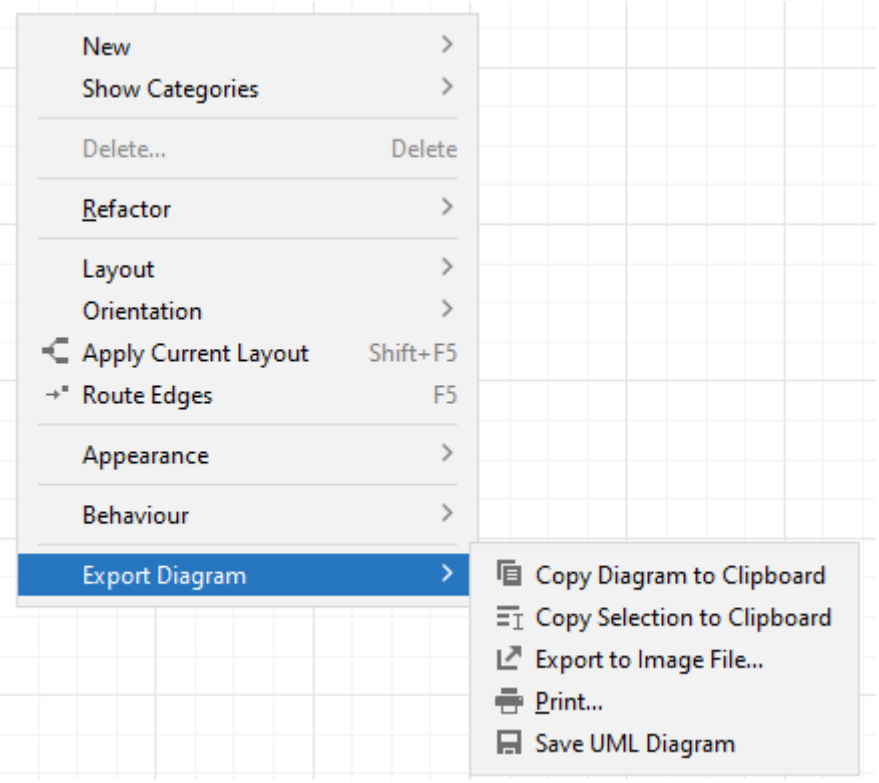


Для того чтобы ваши сущности были обработаны TypeOrm’ом необходимо добавить декторатор **@Entity** согласно документации. Если в конфигурации ORM вы добавили поле synchronize со значением true то согласно правилу описаному в поле `"entities": ["src/**/entites/*.entity{.ts,.js}"]` все ваши сущности с этим декоратором будут автоматически мигрированы в базу данных.

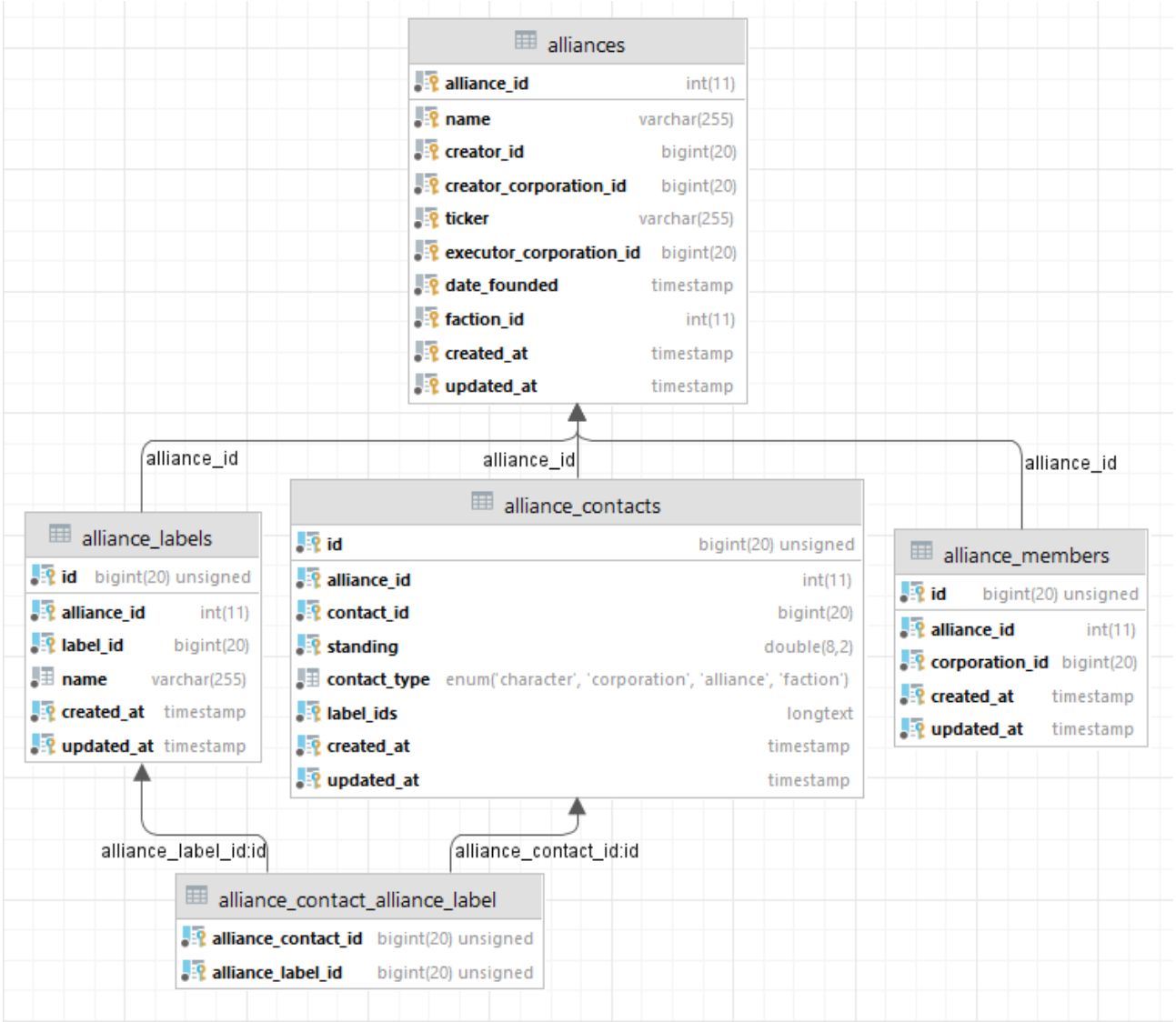
Для Prisma ORM схема объявляется отдельно согласно документации.

После создания всех моделей необходимо сгенирировать визуальное представление, см. пример ниже:





Сгенерируйте визуальное представление схемы вашей данных в виде ERD (диаграммы). Пример диаграммы в DataGrip представлен ниже:



Полученную схему в виде ERD диаграммы загрузите в корень вашего репозитория и опишите значение сущностей (текстовое описание на русском языке) в файле readme.md