To run prisma schema : npx prisma migrate dev --name init

* Vuetify vue3 -frontend
* Nestjs – backend side
* Posgre database oluştur, query oluştur ve nest ile bağla, update işlemi yap
* Prisma ile user module oluştur.

Rectangle

Description automatically generated with low confidence

Nuxtjs oluştur vuetify ile birlikte .

Backend;

* Service, controller bunlar neler araştır.
* Nest generate controller user
* Her işlemde test işlemini gerçekleştir.

MessageQueue araştır.

* Rol ataması

Frontend için;

Frontend tarafını backend ile bağla.

* User, admin ve superadmin olsun.

Bir kullanıcı adı password sayfası oluştur, bir admin panelin olsun.

İmports: İmports[],

Controllers:

Client tarafından gelen requestleri handling(işlemek) yapar ve kullanıcıya response döndürür. Routing mekanizması gelen isteklerin hangi controllerda işleneceğini hesaplar ve isteği o controllera yönlendirir. Basic controller oluşturmak için class ve decorators kullanılır. (Decorators: sınıfları gerekli verilerle ilişkilendirip istekleri gerekli controllerslarla bağlanmasını sağlar)

HTTP isteklerini karşılar ve bunları işler.

Bu controllerlar aldıkları verileri oluşturulan servisler, factory sınıflar yada helperlar sayesinde işleyebilirler.

GET, POST

Params: Parametre sayıları belli olmayan methodlar anlamına gelir.

DTO (Data Transfer Object): DTO, verilerin ağ üzerinden nasıl gönderileceğini tanımlayan bir nesnedir. TypeScript arayüzlerini kullanarak DTO şemasını belirleyebiliriz.

Providers:

Nest'teki sağlayıcılar, Nest'in yerleşik dependecny injection kullanarak denetleyicilere ve diğer sağlayıcılara enjekte edilebilecek services, helpers ve daha fazlasını oluşturmak için kullanılır. @Injectable() dekoratör, bir sağlayıcı sınıfı oluşturmak için kullanılır.

Bu sayede oluşturduğunuz nesneler birbirleri ile ilişkisel halde bulunabiliyorlar ve böylelikle Nest’in runtime’ına basit bir şekilde aktarılabiliyorlar.

Dependency Injection : Bir class’ın diğer classlara bağlı olmasından kurtarılma durumudur. Proje büyüdükçe bu durum sorun olarak karşımıza çıkar. Ayrıca dependency injection unitTest işlemini kolaylaştırır.

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Service : Servisler veriyi tutmak ve üzerinde işlem yapmamızı sağlar. Örneğin bir studentService() içerisinde create() ve findAll() methodları ile öğrenci oluşturmayı ve bütün öğrencileri çağırmayı sağlar. Örneğin client tarafından gönderilen bir istek ( arkadaş listemi görüntülemek istiyorum) bu arkadaş listesindeki datayı serviceler tutar.

Graphical user interface, text, application, email

Description automatically generated

Data migration

İngilizceden çevrilmiştir-Veri taşıma, verileri seçme, hazırlama, çıkartma ve dönüştürme ve bir bilgisayar depolama sisteminden diğerine kalıcı olarak aktarma işlemidir. Ayrıca, taşınan verilerin tamlık için doğrulanması ve eski veri depolamasının hizmetten çıkarılması, tüm veri taşıma sürecinin bir parçası olarak kabul edilir.

TODO :

* UserCreateInput’tan dönen type’ı doğrular eğer doğruysa data oluşturuldu mesajı, doğru değilse error response döndür.
* Ve eğer sonuç başarılı ise deleteMethod fonksiyonu ile datayı sil.
* Prisma.service’yi nasıl global olarak import ederiz?
* Global modulese bak. <https://progressivecoder.com/how-to-use-global-modules-in-nestjs/>
* Expectin içerisindeki kısmı bir değişkene ata daha sonra burda dönen veri tipinin mackResolve içerisinde dönen veri tipiyle uyup uymadığını check et.
* test kısmı için console log bak
* test için iki sorgu oluştur biri user created edilip edilmediği, diğeri ise eğer edilmişse silinip silinmediği kontrolü yapılacak.
* Enum kısmı kod içerisinde rolleri çağırmak için. Senin bir tane daha tablon(roller) olucak.
* One to many roller için ayrı bir tablo oluştur. Bir kullanıcın birden fazla rolü olabilir.
* İd için prisma otomatin primary key atıyor mu?

await userService.deleteUser({id: createdUser.id});