**Projede Bulunan Entityler**

**Animal**

- id:Long

- name:String

- species:String

- breed:String

- gender:String

- colour:String

- dateOfBirth:LocalDate

**Customer**

- id:Long

- name:String

- phone:String

- mail: String

- address:String

- city:String

**Vaccine**

- id: Long

- name: String

- code: String

- protectionStartDate: LocalDate

- protectionFinishDate: LocalDate

**Doctor**

- id:Long

- name:String

- phone:String

- mail: String

- address:String

- city:String

**AvailableDate**

- id:Long

- availableDate:LocalDate

**Appointment**

- id:Long

- appointmentDate:LocalDateTime

**API Temel Özellikleri**

**Hayvanların ve Sahiplerinin (customer) Yönetimi**

* Hayvanları kaydetme, bilgilerini güncelleme, görüntüleme ve silme
* Hayvan sahiplerini kaydetme, bilgilerini güncelleme, görüntüleme ve silme
* Hayvan sahipleri isme göre filtrelenecek şekilde end point oluşturmak.
* Hayvanlar isme göre filtrelenecek şekilde end point oluşturmak.
* Hayvan sahibinin sistemde kayıtlı tüm hayvanlarını görüntülemek için API end point'ini oluşturmak. Hayvan sahibine göre hayvanlara filtreleme yapmalısın.

**Uygulanan Aşıların Yönetimi**

* Hayvanlara uygulanan aşıları kaydetme, bilgilerini güncelleme, görüntüleme ve silme
* Eğer hastaya ait aynı tip aşının (adı ve kodu aynı olan aşı) aşı koruyuculuk bitiş tarihi daha gelmemiş ise sisteme yeni aşı girilememelidir. Aşı kodlarından ve aşı bitiş tarihlerinden bunu kontrol edebilirsin.
* Hayvan id’sine göre belirli bir hayvana ait tüm aşı kayıtlarını listelemek için gerekli API end point'ini oluşturmak.
* Kullanıcının aşı koruyuculuk bitiş tarihi yaklaşan hayvanları listeleyebilmesi için kullanıcının gireceği başlangıç ve bitiş tarihlerine göre aşı koruyuculuk tarihi bu aralıkta olan hayvanların listesini geri döndüren API end  point'ini oluşturmak.

**Randevu Yönetimi**

* Hayvanların aşı ve muayene randevularının oluşturulması, bilgilerinin güncellenmesi, görüntülenmesi ve silinmesi
* Randevular sisteme tarih ve saat içerecek şekilde kaydedilmelidir. Bunun için LocalDateTime kullanılmalıdır.
* Hayvanların her türlü muayenesi için doktorlardan uygun tarihlerde ve saatlerde randevu oluşturulmalıdır. Her doktor için sadece saat başı randevu oluşturulabilir. Bir muayenenin sabit olarak bir saat süreceğini kabul edin.
* Randevu kaydı oluştururken doktorun girilen tarihte müsait günü olup olmadığı eğer ki müsait günü varsa randevu kayıtlarında girilen saatte başka bir randevusu olup olmadığı kontrol edilmelidir. Her iki durum şartı sağlanırsa randevu oluşturulmalıdır. Şart sağlanmaz ise **“throw new RuntimeException("Doktor bu tarihte çalışmamaktadır!/Girilen saatte başka bir randevu mevcuttur.");”** gibi hata mesajı fırlatılmalıdır.
* Randevular kullanıcı tarafından girilen tarih aralığına ve doktora göre filtrelenmelidir. Buna ait API end point’i oluşturulmalıdır. (randevu için kliniği arayan müşterilerin doktor ve gün taleplerinde uygunluk olup olmadığını sorgulamak için kullanılacaktır.) Jpa’nın findBy between kullanımına bakabilirsin.
* Randevular kullanıcı tarafından girilen tarih aralığına ve hayvana göre filtrelenmelidir. Buna ait API end point’i oluşturulmalıdır. Jpa’nın findBy between kullanımına bakabilirsin.

**Veteriner Doktor Yönetimi**

* Veteriner doktorların kaydedilmesi, bilgilerinin güncellenmesi, görüntülenmesi ve silinmesi

**Doktorların Müsait Günlerinin Yönetimi**

* Doktorların müsait günlerini ekleme, bilgilerini güncelleme, görüntüleme ve silme
* Doktorun çalıştığı günler sisteme LocalDate olarak kaydedilecektir. Sadece tarih bilgisi olacaktır. Saat, dakika, saniye bilgisi olmayacaktır.