**Proje debug sırasında webapi rabbit konsolu açar. Database bu sırada oluşturulur.(migration update işlemi webapi startup da içerdeki son hazır migrasyon .cs ine göre çalışır ve yapıyı oluşturur.**

**Database e tablolar oluşturulduktan sonra kullanıcı adı ve şifreyi kendiniz belirlemelisiniz.**

**Yapıdaki örnek gereği sadece zorunlu alan tarih.(02.02.2018) format**

**Açılan webapi sayfasına /help yazarak metodlara ulaşabilirsiniz.**

**LA.BUSINESS:**

**GenericRepository (Abstrack)**

Asenkron olarak Dapper uygulanmıştır ve Sadece Stored Procedure Çağırır.(Post - Get)

**LA.ENTITIES :**

**EntityFrameworks(CodeFirst)**

Entites klasörü altında Database tablo modelleri ve DbContext mevcut.

User ve User Detail tabloları oluşturulurken cascade olacak şekilde bağlandı.

Migrasyon klasörü içerisinde önceden yaratılmış migration sınıfı mevcut.

Add- Migration = yeni database yapısı oluşturur.

Database – Update = son yaratılan database sınıfını yaratır.

Database oluşturulurken kullanılacak Stored Porcedure’ ler de oluşturulur.

Tools klasörü altında Singlaton Pattern uygulanmış bağlantı sınıfı mevcut.

LaBussiness ve Entities projeleri birbirinden bağımsız geliştirilebilir.

**LA.WEBAPI:**

**Asp.Net WebApi Token Based Authentication(Oauth2)**

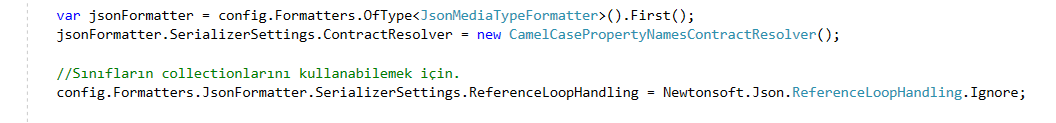
**Context Burada üretilir. Yani müşteriye gittiniz wep apiyi kurdunuz. İlk token isteği ile database oluşturulur. Denemedim ama böyle 😊**

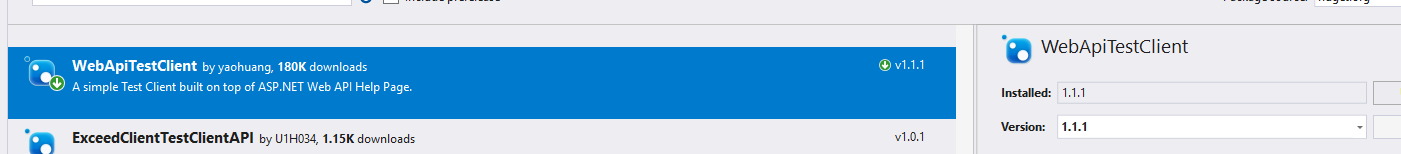
Help sayfası oluşturmak için alltaki Nuget WebApiTestClient paketi kullanıldı ve Custom LaWebApi.Xml oluşturuldu.

Help page xml i güncellemek için rebuild işleminden önce App\_Data ve Bin klasörlerinden xml silinmeli.

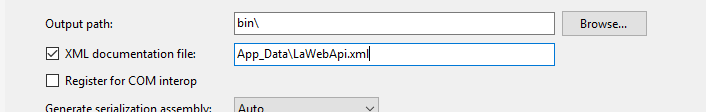
HelpPage Url(Debug sırasında)= <http://localhost:42875/help>

Model ve entities nesnelerinin Json formata cevrilebilmesi ve collection property lerine ulaşabilmek için

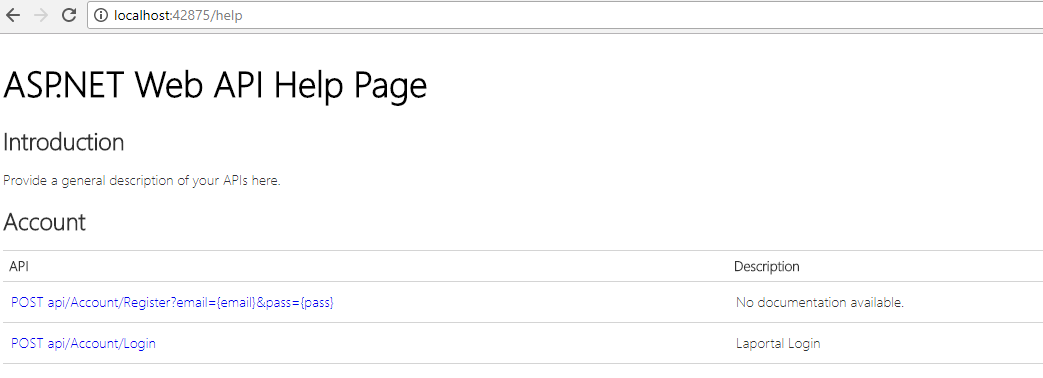




Help page için xml



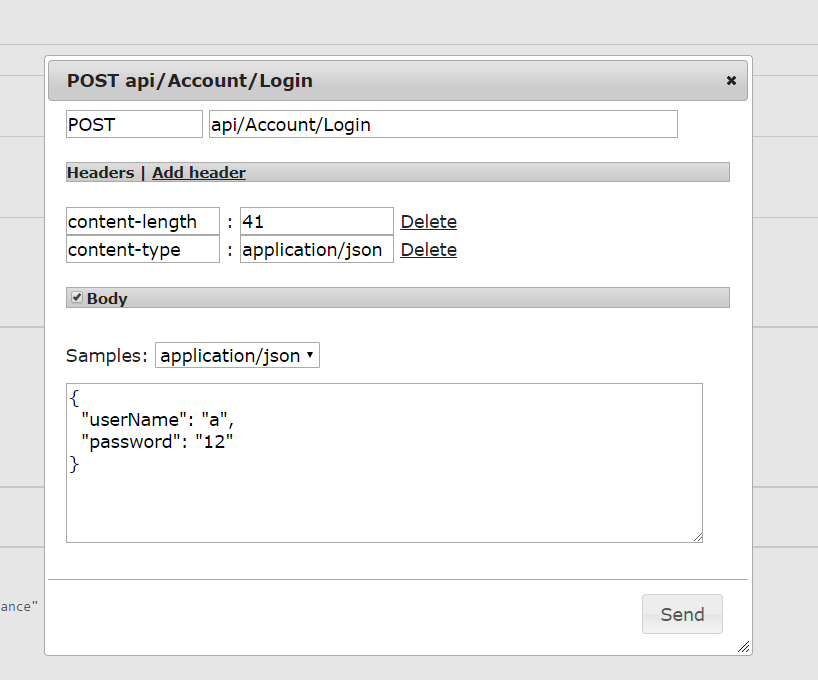
Help Page



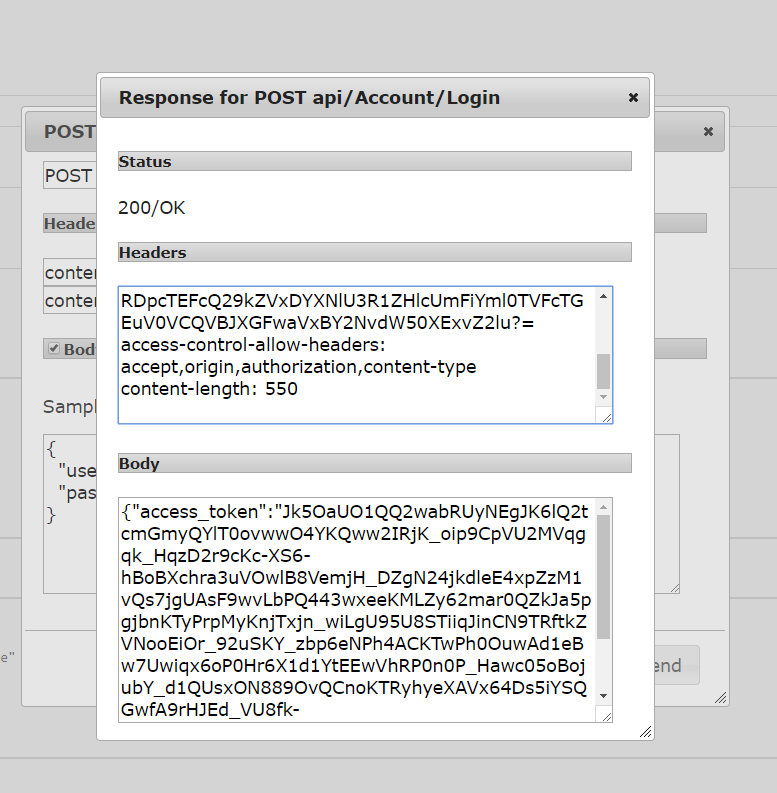
**Login İşlemleri ve Bearer Token Üretimi**

Help page ana sayfasından test login linkine tıkanır ve login giriş ekranı alt sağdaki popup butonu ile açılır.

Json Formatında veri gönderilir. Kullanıcı bilgileri alttaki resimde verildi.



Status Rasponse Code : 200/OK



Rasponse içeriği :

access\_token":"T2XAYC-kYxR21XxHi81X7U3fzJsvJKu43KAF9uCCkwaDZGaAK04uxxtIEHRNQqAHJWZiBDBS0LoJNfMPRxRoMNGZuQAGY8KUUgKay6RqPki4M-H1CXgDWV\_RYiFmCDbpDvmCD8Gn\_VhCMdEmKTTeVOg7dJwYpOs7zbsWmhs6wKODTrLY23VDUcDWaw\_gSaZlD0YhUSLgfGwdntKwxNxTzzOGICG4NbZ--gvEIylyfsQFTYO7F\_6am\_Bfy28tnSRO9R8xN5Ntj6NblV2tIPQqWg-vJQF-iWBdwRqEVUScQrUYH1hhsZbRjCKebiENJaVWv5p44sXLj\_UDJOXJmjm9VGsFbR0ud0IXZna6X41Vs3s",

"token\_type":"bearer",

"expires\_in":86399,

"userName":"a",

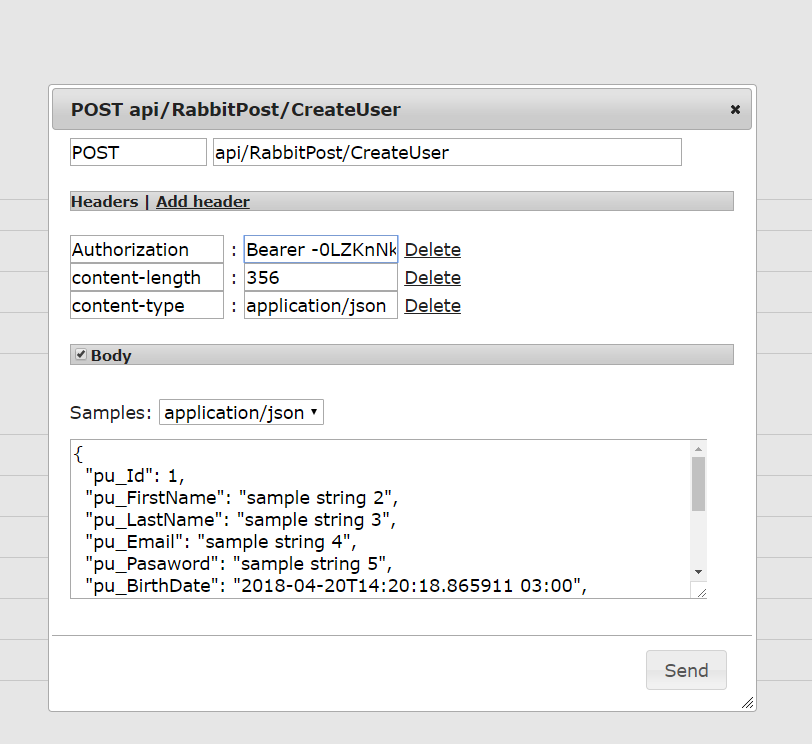
"userId":"1",

"userRole":"",

".issued":"Thu, 19 Apr 2018 19:13:29 GMT",

".expires":"Fri, 20 Apr 2018 19:13:29 GMT"

Token edinildikten sonra Diğer metod da kullanımı alttaki resimdeki gibidir.



RabbitPostController sınıfındaki CreateUser metoduna user Json olarak post edilir.

CreateUser metodundaki RabbitMq ya post işlemi generik sınıf yazılarak json formatında

Gönderim yapıldı.

**LA.RABBITMQLISTENER:**

**RabbitMq Consumer**

Rabbit e gönderilen istek burada yakalanır ve mongo db deki log tablosuna QUE olarak kaydeder.

Sonra sql insert işlemini yapar

Status CPM ‘e update edilir.

Hata oluşursa status ERR olur.

Zaman yetmediği için User insert işlemi yapılamadı. Mongo tarafı test edilemedi. Üzgünüm :☹