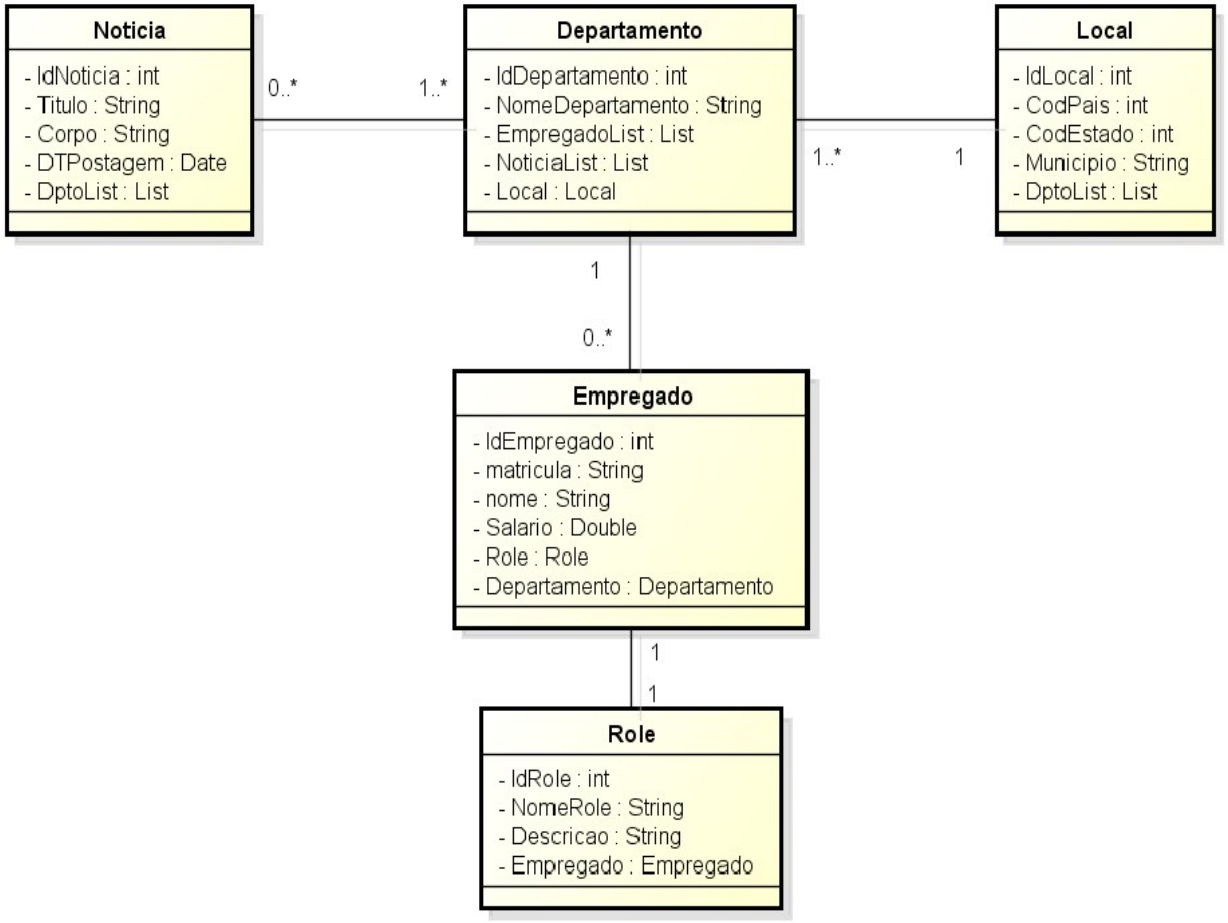
1 – a . Tendo em vista o crescente número de funcionários na empresa e a quantidade de espaço que o novo quadro ocuparia, foi acordado com o Diretor que a criação de um sistema web (Intranet) para os departamentos divulgarem suas notícias e calendários seria uma alternativa mas viável. Pois além de ocupar menos espaço físico, a criação da Intranet também contribuiria para a economia de papel.

Fará parte deste sistema as seguintes entidades:



2 – Para poder realizar a personalização da ferramenta **nopCommerce** foi necessário entendê-la. Para tal precisei ler, cuidadosamente, a documentação.
Após ler a documentação e obter certo entendimento da arquitetura e de alguns padrões de projetos (utilizados pelos criadores da ferramenta) decidi debuggar o código e tentar entender, um pouco, o que estava acontecendo.
Como ia ser preciso fazer alterações no banco de dados, utilizando o padrão **"Code First",** eu tentei fazer algumas **"migrations"** para alterar a tabela de Produtos (Products), porém não obtive êxito devido alguns problemas de configuração do meu ambiente.
Diante deste pequeno inconveniente, tive que fazer como a documentação orientava! Alterei a tabela "Products" manualmente. (Adicionei um campo do tipo **nvarchar**, denominado **"Minha Nova Propriedade"**).
Feito as alterações no banco de dados, as seguintes alterações foram feitas no código fonte:

a – No assembly "`Nop.Core`" foi necessário adicionar a propriedade do tipo string: `MinhaNovaPropriedade` à classe Produto do namespace `Nop.Core.Domain.Catalog`;

b – No assembly "`Nop.Data`" foi necessário adicionar o seguinte código ao construtor da classe

ProductMap (namespace `Nop.Data.Mapping.Catalog`):

```
this.Property(p => p.MinhaNovaPropriedade).HasMaxLength(400);
```

c – No assembly "`Nop.Admin`", o qual é um projeto do tipo MVC referente a área de administração do sistema, foram feitas alterações (adição da nova propriedade) nas classes ProdutoModel e ProdutoLocalizedModel, as quais pertencem ao namespace `Nop.Admin.Models.Catalog,` para que o novo campo seja exibido nas views;

d – Ainda no assembly "`Nop.Admin`" foi observado que as informações referentes ao cadastro de produto são exibidas na View : Create.cshtml e em diversas Partial Views:, dentre elas: _CreateOrUpdate.Info.cshtml, _CreateOrUpdate.SEO.cshtml e _CreateOrUpdate.Categorie.cshtml;

// Para esta atividade, foram realizadas alterações na partial view: _CreateOrUpdate.Info.cshtml

```
<tr>
    <td class="adminTitle">
       @Html.NopLabelFor(model => model.MinhaNovaPropriedade):
     </td>
     <td class="adminData">
        @Html.EditorFor(model => model.MinhaNovaPropriedade)
        @Html.ValidationMessageFor(model => model.MinhaNovaPropriedade)
     </td>
 </tr>
```

e – Por último foi necessário alterar o ActionResult Create* na classe ProductController :

```
product.MinhaNovaPropriedade = model.MinhaNovaPropriedade;
```

* Em ProductController existem dois métodos Create, o primeiro faz um GET e é responsável por "renderizar" parte do form na página de cadastro de produto, já o segundo faz um POST e é responsável por fazer o submit dos dados presentes neste form.
Para esta atividade a seguinte alteração foi feita no segundo método:

```
//product
var product = model.ToEntity();
product.MinhaNovaPropriedade = model.MinhaNovaPropriedade;
```