**Documento de Arquitetura de Software**



**PosterCommerce**

**Histórico de Versões**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data** | **Versão** | **Descrição** | **Autor** | **Revisor** |
| 17/04/2016 | 0.1 | Modelagem e desenvolvimento | Vitor | - |
| 16/05/2016 | 0.2 | Melhorias | Vitor | - |
| 18/06/2016 | 0.3 | Versão final | Vitor | Rodrigo Rocha |

|  |  |
| --- | --- |
| **Cliente** | FATEC - Interno |
| **Documento** | Documento de Arquitetura de Software: *Poster Commerce* |
| **Data** | 18 de junho de 2016 |
| **Autor** | **Vitor Duque**  vtrduque@gmail.com |

**Página de Assinaturas**

|  |  |  |
| --- | --- | --- |
| Revisado e Aprovado por: |  |  |
|  |  |  |

**Índice**

Camada de Apresentação 7

Camada de Persistência 11

Diagrama de caso de uso geral do sistema 5

Escopo 4

Objetivo 4

Pacote Model 10

Qualidade 13

Realização dos Casos de Uso Significativos 11

Representação Arquitetural 4

Tamanho e Performance 13

Visão de Dados 13

Visão de Implementação 13

Objetivo

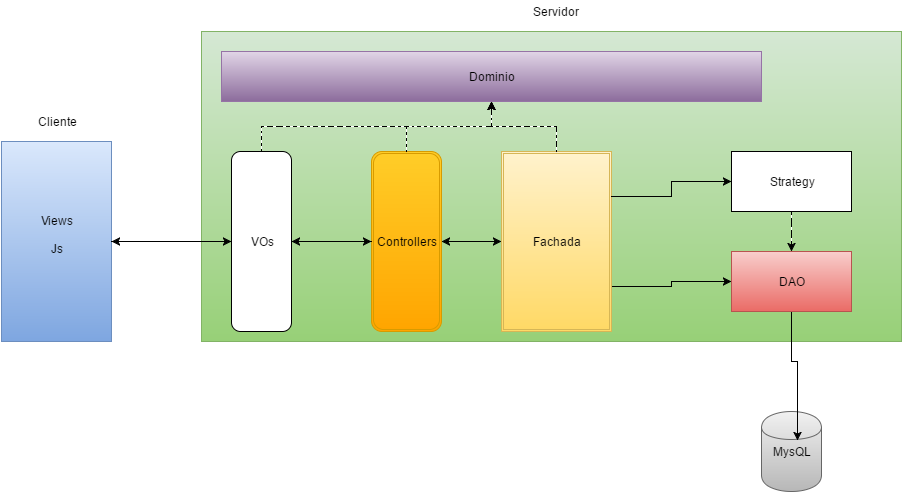
Este documento tem por objetivo apresentar uma visão arquitetural do sistema PosterCommerce. O intuito é salientar diferentes aspectos deste produto, obtidos a partir de decisões arquiteturais realizadas no âmbito do sistema.

Escopo

O escopo deste documento é documentar as partes significativas do ponto de vista da arquitetura do modelo de design, como sua divisão em subsistemas e pacotes. Além disso, mostra sua divisão em classes e utilitários de classe.

Representação Arquitetural

Os sistemas serão desenvolvidos tendo como base a arquitetura ilustrada na Figura 1. Toda a arquitetura será baseada nos padrões de projetos tradicionais do GoF e também nos padrões do framework Ruby on Rails (existindo a particularidade do não uso do padrão ActiveRecord), executados dentro de um Servidor de Aplicações.



**Figura 1 - Modelo Arquitetural Genérico**

As Views abrigaram os componentes da arquitetura responsáveis pela camada de apresentação. Será utilizado para esta camada o framework Ruby on Rails na implementação do modelo MVC.

As Classes de Domínio são as classes que representam os Value Object, contendo somente os atributos e os “métodos getters/setters”.

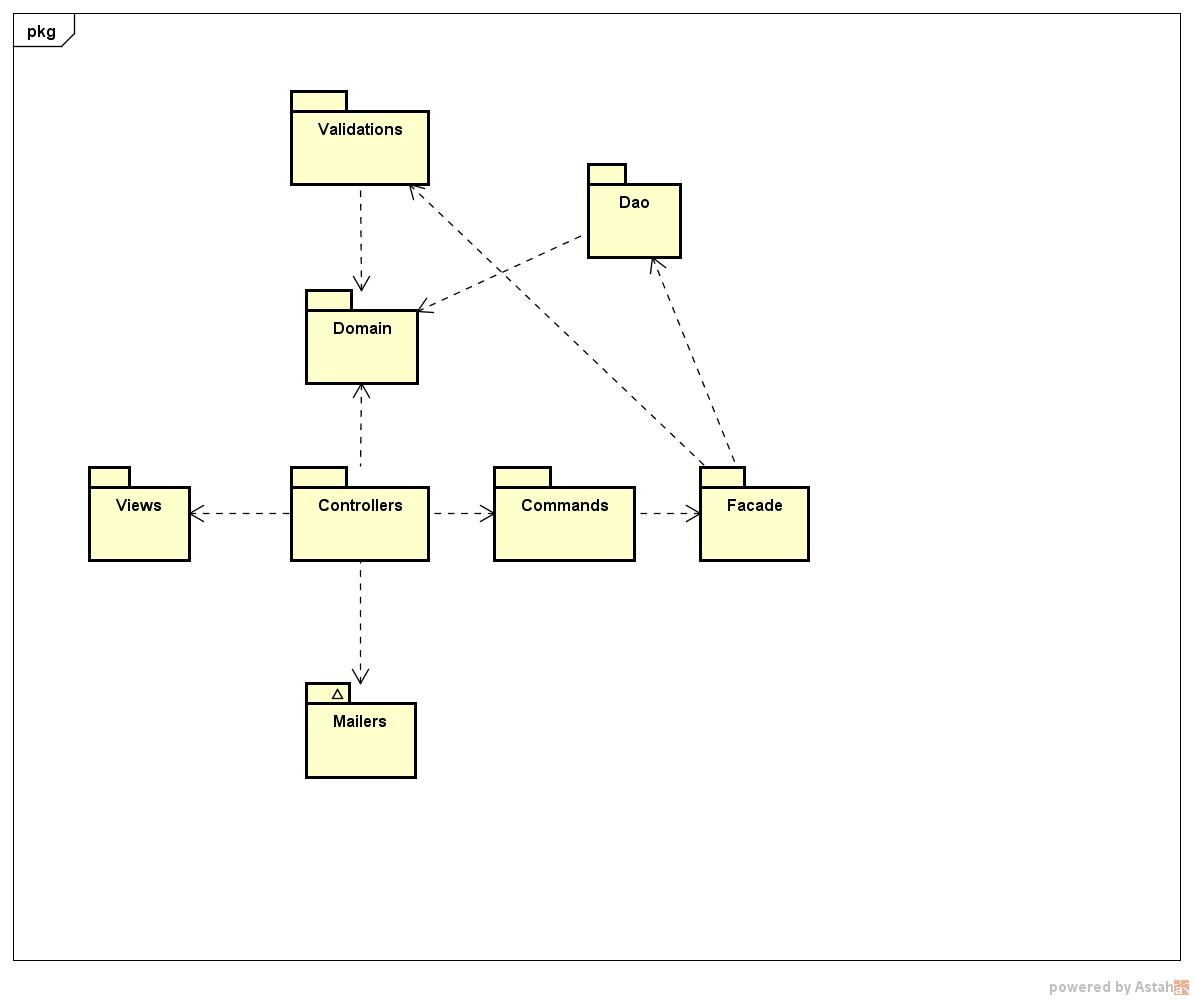
As Classes de Negócio representam as classes responsáveis por aplicar as regras de negócio do sistema como, por exemplo, cadastrar pôster. Constarão também nas classes de negócio o relacionamento com os DAOs responsáveis por persistir e recuperar os objetos no banco de dados.

As Classes denominadas Controllers são responsáveis por criar os objetos com informações das Views, manda-los para a fachada, tratar o seu retorno e se comunicar com as Views novamente.

A fachada é uma classe responsável por invocar as classes de negócios responsáveis pela classe de domínio que lhe passada (executar validações e salvar no banco de dados).

Banco de dados: Banco onde as informações serão persistidas. Banco utilizado: MySQL.

O diagrama a seguir, figura 2, representa a organização das classes dentro dos pacotes, tanto da arquitetura como da aplicação que fará uso da arquitetura.



**Figura 2 - Diagrama em pacotes**

* 1. Restrições Arquiteturais

Foram identificadas algumas orientações / restrições pertinentes ao desenvolvimento deste subsistema:

* Utilização do Ruby 2.3.1;
* Utilização framework Rails 4.2.6;
* Utilização do SGBD MySQL.

1. Visão de Use Case

Esta seção apresenta os Casos de Uso arquiteturalmente significativos.

A classificação dos casos de uso, em termos de significância, foi realizada com base na observação de pelo menos um dos seguintes critérios:

* Casos de Uso que são incluídos em outros Casos de Uso
* Casos de Uso de Condução
* Casos de Uso de Configuração
* Casos de Uso de Análise
  1. Diagrama de Casos de uso

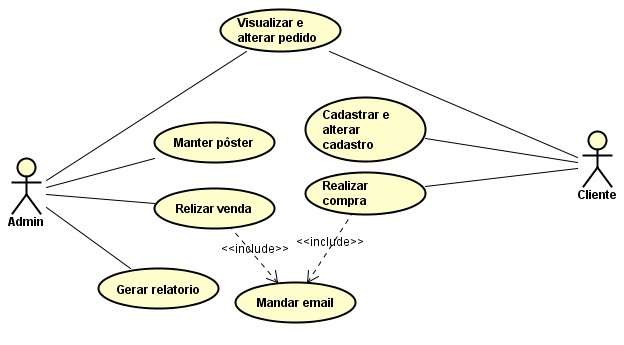


Figura 4.1 Diagrama de Casos de uso

* 1. Descrição dos Casos de Uso Arquiteturalmente Significativos

**Manter pôster**

Este caso de uso representa as funcionalidades do sistema que permite o administrador cadastrar, alterar, excluir e visualizar pôsteres.

**Realizar Venda:**

Este caso de uso representa a funcionalidade do sistema que permite o administrador alterar os status do pedido

**Gerar relatório**

Este caso de uso representa a funcionalidade do sistema que permite o administrador visualizar relatórios dentro de uma data especifica.

**Visualizar e alterar pedido**

Este caso de uso representa a funcionalidade do sistema que permite o administrador e o cliente visualizarem e alterarem os pedidos.

**Cadastrar e alterar cadastro**

Este caso de uso representa a funcionalidade do sistema que permite o cliente se cadastrar e alterar seu endereço.

**Realizar compra**

Este caso de uso representa a funcionalidade do sistema que permite o cliente realizar compras no PosterCommerce

**Mandar email**

Este caso de uso representa a funcionalidade do sistema de mandar e-mails sempre que o status do pedido for alterado, tanto pelo usuário (desistência da compra ou devolução dos pôsteres), quanto pelo administrador (alterando o status da compra).

1. Visão de Lógica

Esta visão apresenta elementos de design significativos do ponto de vista da arquitetura, descrevendo a organização do Sistema PosterCommerce em pacotes, bem como a organização desses pacotes em camadas.

O Diagrama com as camadas do PosterCommerce é ilustrado na figura 5.1.

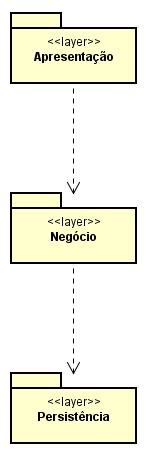


Figura 5.1 – Diagrama de camadas do PosterCommerce

**Apresentação**: Contém classes para as interfaces gráficas com os usuários (GUI). Através destas interfaces os usuários conseguem interagir com o PosterCommerce, com o intuito de pelo menos realizar uma das funções básicas (incluir, alterar, excluir e visualizar).

**Negócio**: Contém classes que controlam a execução das funcionalidades do PosterCommerce.

**Persistência**: Contém classes responsáveis por persistir as entidades de modelo. Por exemplo,

contém as classes que permitem ler e gravar os objetos no banco de dados relacional.

A Figura 5.2 ilustra o diagrama de camadas com as tecnologias utilizadas no desenvolvimento, já descritas na figura 5.1.

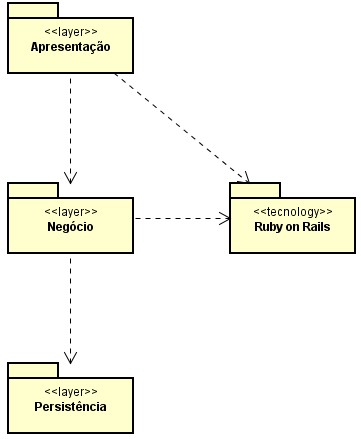


Figura 5.2: Camadas do PosterCommerce com as dependências de tecnologia

* 1. Camada de Apresentação

Nesta camada, temos o pacote Views que contém todos os arquivos relacionados à exibição de informações para usuário, o que engloba páginas html.erb. Já o pacote Controller desta camada, contém as classes responsáveis pela comunicação com as classes da camada de negócio. Estes pacotes podem ser vistos na Figura 5.3.

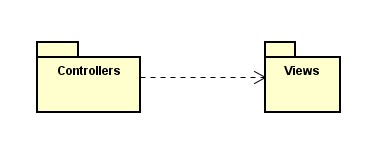


Figura 5.3: Camada de Apresentação

## Camada de Negócio

Nesta camada, temos o pacote Controller que contém as classes responsáveis por controlar as regras de negócio da aplicação. O pacote Domain, contém as classes que representam o modelo, ou seja, aquelas que contém as informações. O pacote Mailer contém as classes responsáveis pelos envios de e-mails.

A imagem 4.4 ilustra os pacotes descritos.

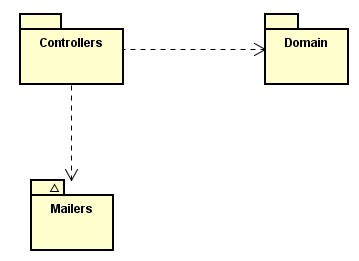


Figura 5.4: Camada de Negócios

* + 1. Pacote Controllers

A figura 5.5 ilustra as principais classes de controle.

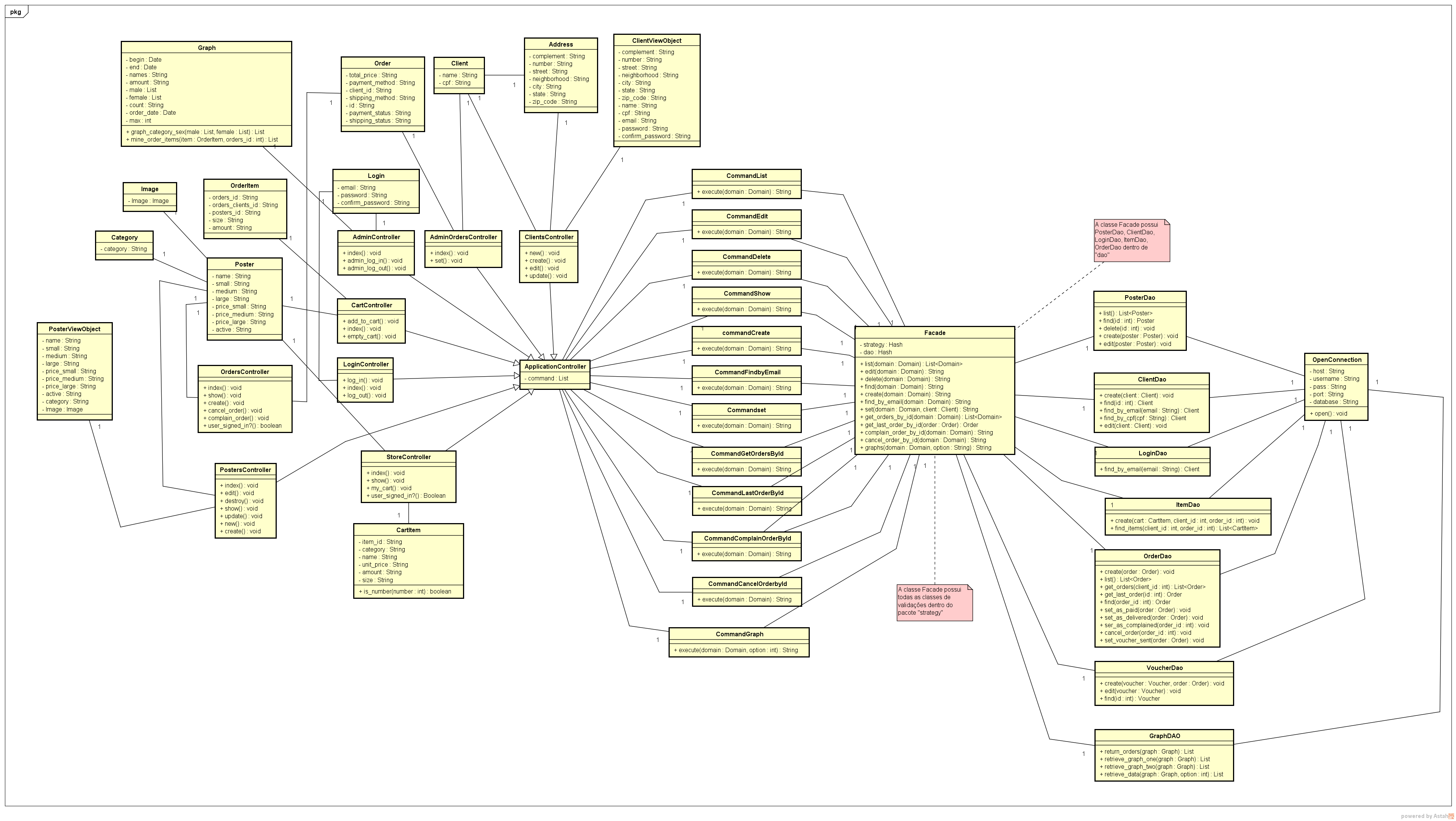


Figura 5.5: Classes de controle

* + 1. Pacote Model

A figura 5.6. Ilustra as principais classes do modelo.

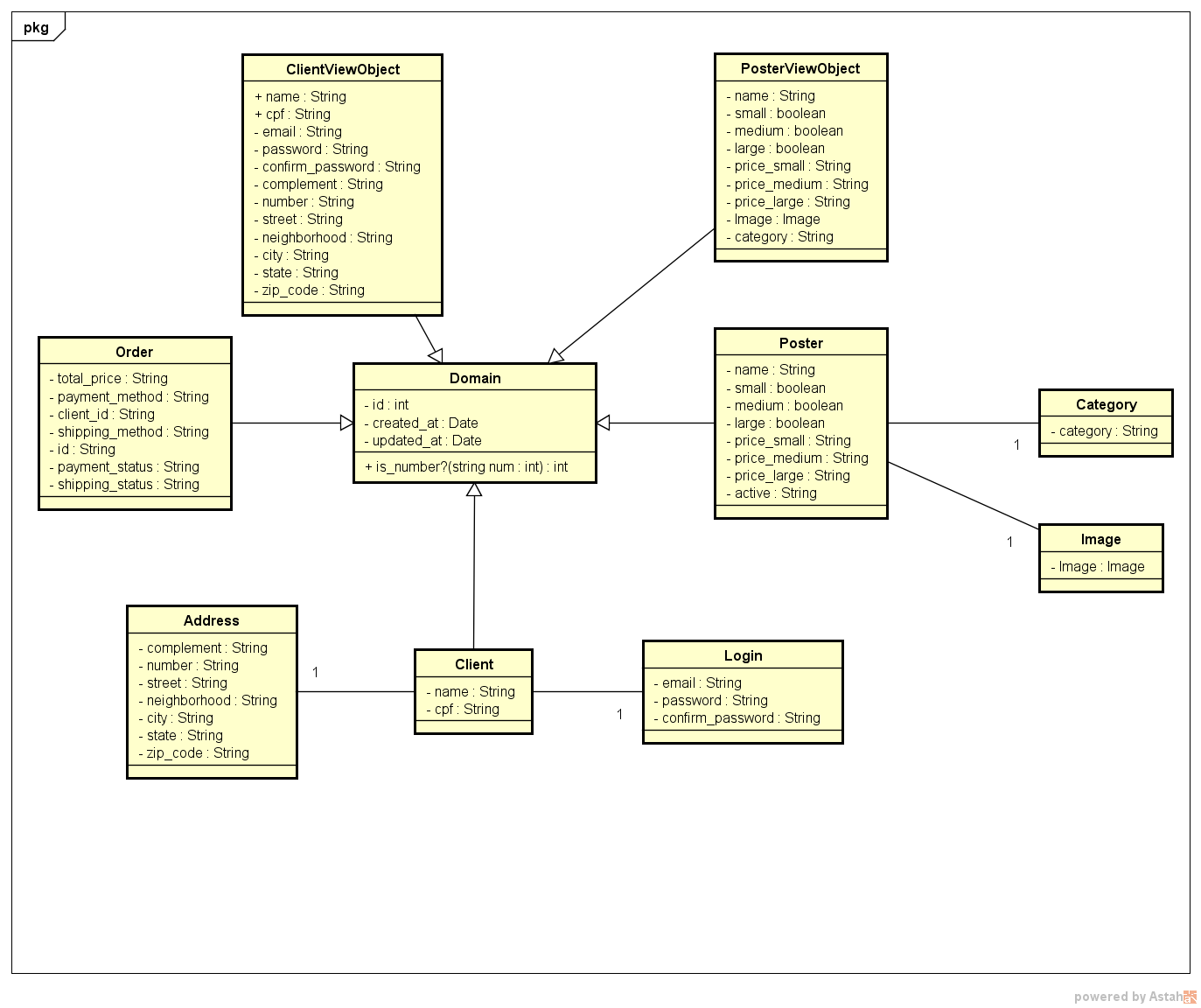


Figura 5.6: Classes do Modelo

## Camada de Persistência

Nesta camada as classes DAOs responsáveis por persistir as informações do Poster Commerce no BD relacional.

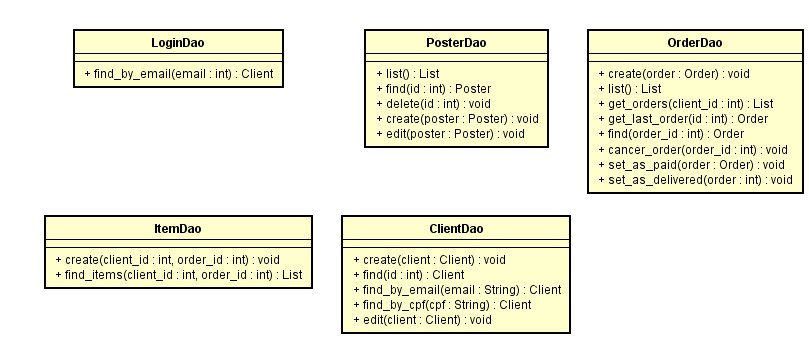
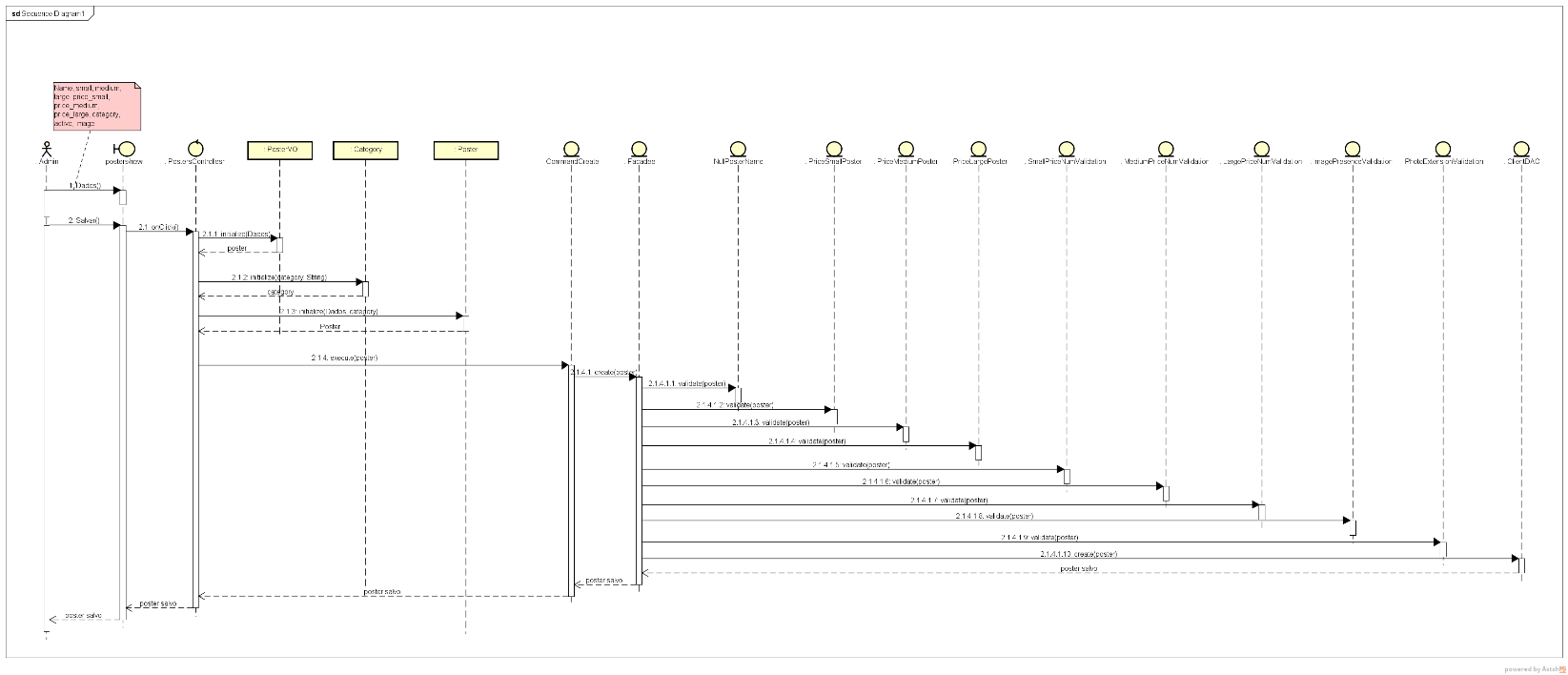


Figura 5.7: Camada de Persistência.

## Realização de um dos Casos de uso Significativos (Cadastrar pôster):



1. Visão de Implantação

Esta seção descreve as configurações da rede física (hardware) na qual o PosterCommerce será implantado e executado.

Trata-se de uma visão do Modelo de Implantação que, para a configuração em questão, indica os nós físicos (computadores, CPUs) e as respectivas interconexões (barramento, LAN, etc). A figura 6 ilustra o modelo de implantação para o PosterCommerce.

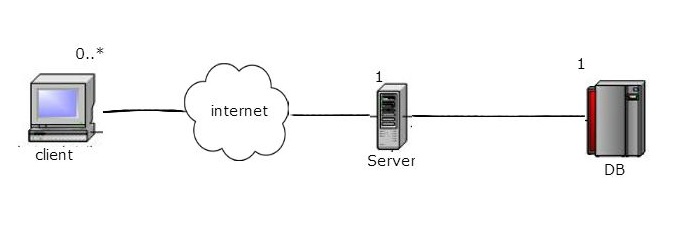


Figura 6: Visão de Implantação do PosterCommerce

Na Figura 6 observa-se os seguintes elementos:

* **Client:** Usuário que irá acessar a aplicação remotamente
* **DB**: BD Central do PosterCommerce.
* **Internet:** O usuário deverá ter conexão coma internet para poder se conectar no site do PosterCommerce
* **Server:** Servidor onde estará localizado a aplicação PosterCommerce

1. Visão de Implementação

Esta visão descreve a estrutura geral de implementação, a decomposição do software em camadas de implementação.

A estrutura geral de implementação para o PosterCommerce é baseada na estrutura da Visão Lógica, assim, não há necessidade de detalhar os diagramas de camadas e pacotes de implementação, uma vez que são fortemente baseados naqueles desenvolvidos para Visão Lógica.

1. Visão de Dados

O mecanismo de persistência utilizado no sistema PosterCommerce utiliza-se o banco de dados Relacional MySQL.

As figuras 8.1 e 8.2, apresentam a visão lógica e física da base de dados do PosterCommerce.



Figura 8.1 – Modelo Lógico

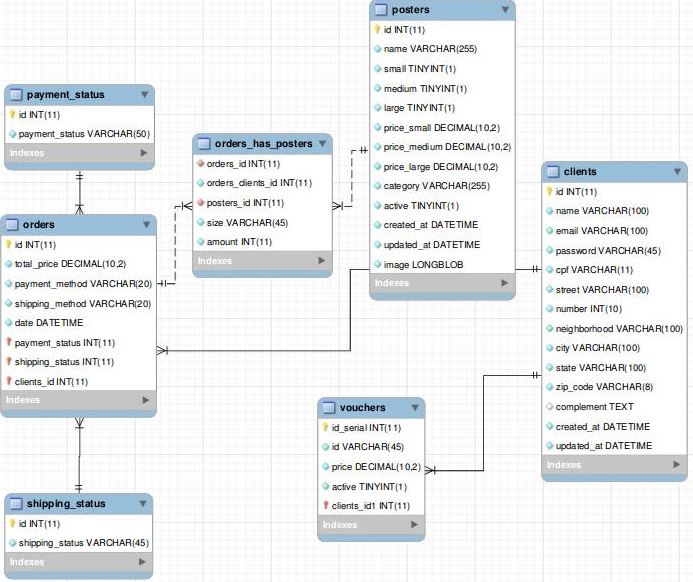


Figura 8.2 – Modelo Físico

Tabela 2: Mapeamento Objeto-Relacional

|  |  |
| --- | --- |
| **Classe** | **Entidade** |
|  |  |
|  |  |
|  |  |
|  |  |

1. Tamanho e Performance

O sistema PosterCommerce será usado para realização de venda de pôsteres online. As mudanças no servidor (tamanho x performance) irá ser alterado quando necessário.

Seus servidores provavelmente irão passar por períodos de picos de utilização (por exemplo, em uma black-friday).

Em questão de performance, o servidor irá aguentar a quantidade de visitas que for necessária, mantendo a performance alta.

1. Qualidade

O sistema PosterCommerce será usado para a realização de vendas de pôsteres online.

Eventuais erros e/ou falhas na sua operação podem levar a prejuízos financeiros significativos, portanto na fase de design deve-se levar em consideração como fatores prioritários a confiabilidade e robustez do sistema.

Adicionalmente, o sistema PosterCommerce pode ser alvo de ataques de “hackers” para roubar ou simplesmente corromper informações, possibilidade aumentada pela interface do sistema estar disponível na Internet. Para evitar que tais ataques sejam bem-sucedidos uma infra-estrutura de segurança deve ser especificada e projetada.

1. Referências

Modelo disponibilizado no dropbox da matéria e os documentos de requisitos.