

## Universo de Discurso:

O objetivo do trabalho é descrever um sistema (SI) de compra e entrega à domicílio de comida. Foi tomado como base a plataforma de **clientes** do site do [iFood \(ifood.com.br\)](http://iFood.com.br) (não está sendo modelada a plataforma de prestadores de serviço).

O sistema do iFood é baseado em uma experiência de uso simples, em que um cliente escolhe uma refeição para comprar e ser entregue em sua casa baseado em seus gostos, restaurantes e pratos favoritos.

### Clientes:

Os clientes (usuários) são os atores principais do sistema, Para usar o sistema, cada cliente cadastra seu CPF (único), nome, telefone e e-mail (único). Cada cliente pode vincular ao seu perfil uma lista de cartões para utilizar como forma de pagamento.

### Restaurantes:

★ 4.6 (210 Avaliações)

**Woking Thai Food** ★

Já imaginou o sabor da culinária tailandesa preparado do seu jeito? Peça logo o seu.


Comida Tailandesa • 50-60 min • 2.28 km • \$\$\$ • ENTREGA R\$ 9,90

Ver mais

Promoção

Q Buscar no cardápio


**Destaques**



**Promo thai 1**

Massa de ovos com carne de gado, brócolis e amendoim, salteadas na...


**R\$ 24,50** R\$ 35,50



**Promo thai 2**

Massa udon com frango e abacaxi, salteados na wok, com molho shoyu...

**R\$ 22,90** R\$ 33,00



**Rolinho primavera**

Veggie (5 unidades de rolinhos bem crocantes) ;D

**R\$ 15,00** R\$ 19,50

Figura: Exemplo de restaurante.


No sistema, **restaurantes** são os responsáveis por receber os pedidos dos clientes e preparar a comida de cada pedido. Cada restaurante tem como dados seu CNPJ (único), nome da empresa, nome fantasia, endereço, sua nota geral de avaliações, e um cardápio. Para cada restaurante, deve ser informado ao cliente o tempo médio de espera para receber uma compra e a taxa de entrega dos pedidos do restaurante.

## Pratos:

**Favorito 1**

Massa de ovos + gado + ovo + molho woking (teriyaki - a base de shoyu e gengibre). Servido com repolho verde e roxo, cebola, cenoura e cebolinha verde. Vegetais já estão misturados (não é possível selecioná-los)

R\$ 25,90

 Woking Thai Food

50-60 min

**Ingredientes extras:**  
Escolha até 3 opções.

Com farofa de amendoim

+ R\$ 1,00

+

Com pimenta extra

+ R\$ 0,60

+

Com gergelim torrado

+ R\$ 0,60

+

- 1 +

Adicionar R\$ 25,90

Figura: Exemplo de prato.

Ao selecionar um **prato (item do cardápio)**, o cliente pode ver as características do mesmo. Cada item do cardápio do restaurante tem seu nome, descrição, preço, tempo médio de preparo e tempo médio de entrega (mas o cliente só precisa saber a soma dos dois), e adicionais (opcionais) que o cliente queira colocar em seu pedido. Os adicionais são compostos de nome, preço e quantidade em cada prato. Cada prato é categorizado em uma ou mais **cozinhas**, para que o usuário possa filtrar os tipos de prato que quer ver ao fazer uma busca por pratos.

Ao decidir os detalhes de cada prato, o cliente pode colocá-lo em seu carrinho, que só aceita pratos de um único restaurante por vez, na quantidade que o cliente bem entender. Ao

adicionar seu prato ao carrinho, o cliente pode continuar navegando e adicionar mais pratos do restaurante em seu carrinho, seguindo o mesmo fluxo anteriormente descrito.

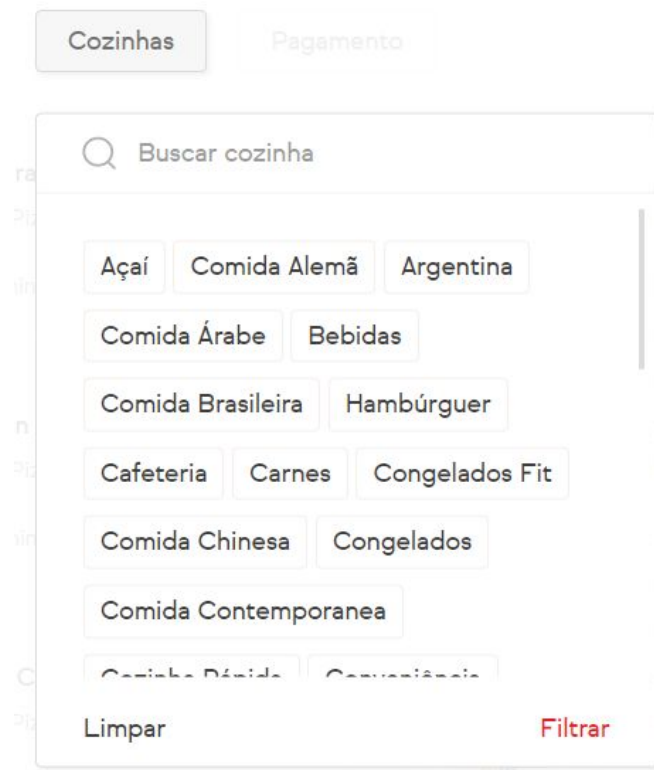


Figura: Exemplo de cozinhas.

## Carrinho:

O carrinho de cada cliente é composto de todos itens que o cliente selecionou, é como um pedido ainda editável, não finalizado. Dentro do carrinho de cada cliente é possível ver informações dos pratos, um subtotal, a taxa de entrega do restaurante e o somatório dos preços. O usuário tem a opção de editar seus pratos ou seguir para a finalização do pedido.

Seu pedido em

## Woking Thai Food

1x Favorito 1 R\$ 26,90

1x Com farofa de amendoim

[Editar](#) [Remover](#)

1x Thai raiz 3 R\$ 35,20

[Editar](#) [Remover](#)

Subtotal R\$ 62,10

Taxa de entrega  R\$ 9,90

**Total R\$ 72,00**

[Escolher forma de pagamento](#)

Figura: Exemplo de carrinho.

## Finalização:

Na finalização do pedido, o cliente pode escolher a forma de pagamento, que pode ser pago no momento da entrega, ou antecipadamente. Caso o pagamento seja feito na entrega, pode ser por cartão ou dinheiro, já se for feito online, é obrigatoriamente feito no cartão. Nesse momento da compra, o cliente deve escolher também o endereço de entrega.

## Finalize seu pedido

### Entregar em

 Avenida Coronel Lucas de Oliveira, 2367  
Apto. 28 - Porto Alegre/RS

[Trocar](#)

### Pagamento

[Pague na entrega](#)

[Pague online](#)

 Dinheiro

 Débito - Visa

Seu pedido em

## Woking Thai Food

1x Favorito 1 R\$ 26,90

1x Com farofa de amendoim

[Editar](#) [Remover](#)

1x Thai raiz 3 R\$ 35,20

[Editar](#) [Remover](#)

 **Cupom**  
Insira um código



Figura: Exemplo de finalização de pedido.

## Cupons:

Ainda na finalização do pedido, o cliente pode adicionar um cupom de desconto ao seu pedido. Cada cupom de desconto pode ser vinculado a mais de um usuário, mas cada usuário pode usar ele determinada quantidade de vezes, sem interferir na quantidade desse cupom disponível para outro usuário. Os cupons têm seu identificador único, validade, e dão um valor em reais de desconto no total do pedido do cliente. Ao utilizar o cupom e finalizar o pedido, uma unidade do cupom é retirada do usuário.

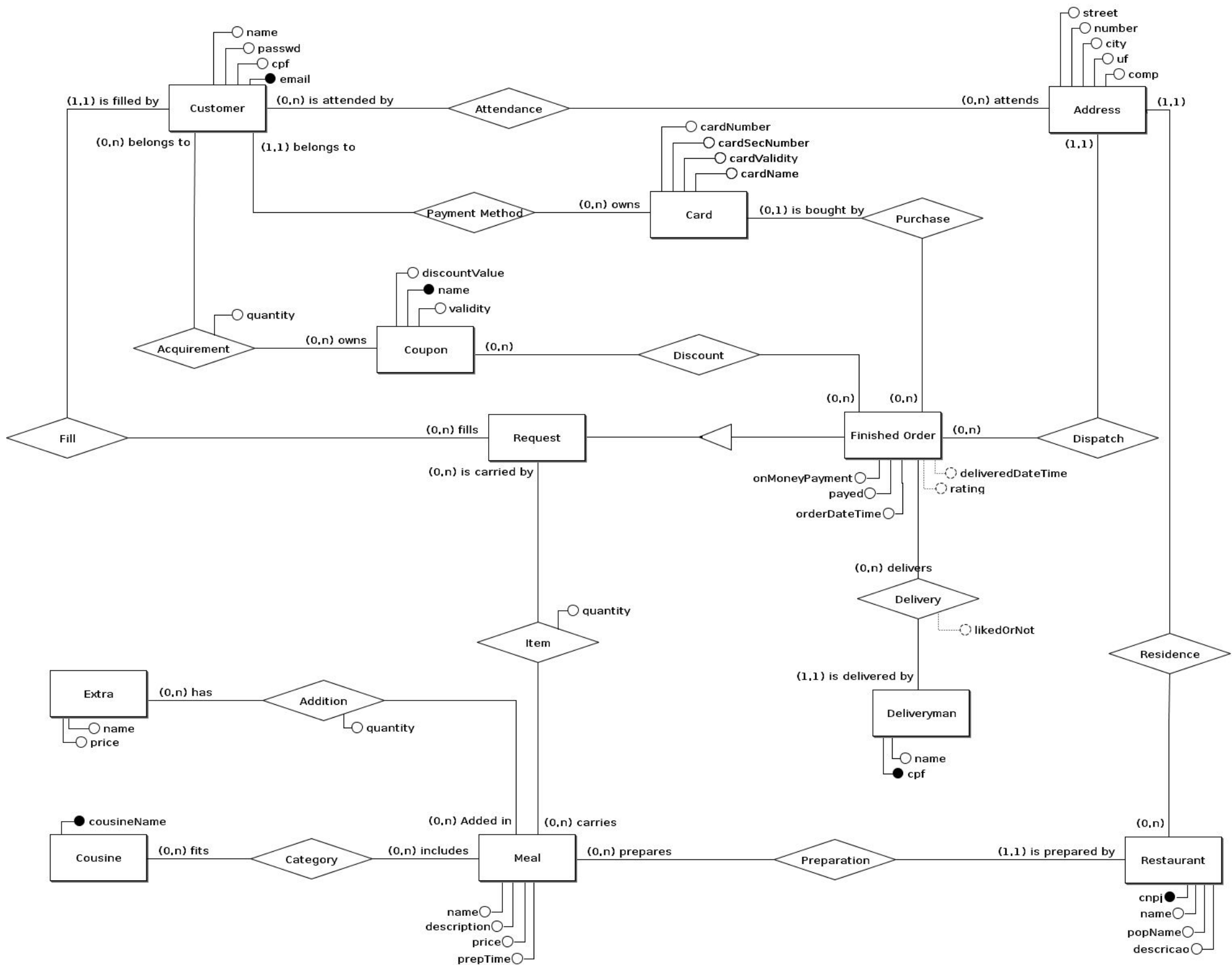
## Entrega:

Total	R\$ 11,90
Entrega em	
Avenida Coronel Lucas de Oliveira, 2367 - Petrópolis - RS - Porto Alegre/RS	
Nº do pedido	
2669	
Data do pedido	
09/10/2019 • 16:12	
Pagamento pelo App	
MASTERCARD	

Figura: Exemplo de pedido finalizado.

Assim que decididas as opções de pagamento, cupons e endereço, o cliente finaliza o pedido. Ao finalizar, o pedido é enviado ao restaurante, que o prepara e envia para entrega. No ato da finalização, o pedido recebe os dados da finalização: sua data/hora, forma de pagamento e cupons utilizados, se o pedido foi pago antecipadamente, já é adicionado o *status* de pago. Além disso, é destinado ao pedido um entregador. O gerenciamento dos entregadores é feito pelo sistema central do ifood, e tudo que o cliente precisa ser informado é o nome do seu entregador, para poder identificá-lo no momento do recebimento.

Ao chegar no endereço de entrega, o é adicionado o *status* de entregue, caso o pagamento fosse feito na hora, o status do pagamento é atualizado no momento da entrega. Assim que o pedido é entregue, o cliente pode adicionar uma classificação de 0 a 5 estrelas para o seu pedido, que é utilizada na classificação do restaurante, e uma indicação de *gostei* ou *não gostei* para a entrega do pedido.



## Dicionário de Dados:

Abaixo, encontra-se o dicionário de dados para o diagrama ER criado. Cada tabela representa uma entidade ou relacionamento presente. As entidades são representadas em **cinza**, os relacionamentos em **azul**.

Chaves candidatas das entidades são definidas em **negrito**. Atributos opcionais são definidos com asteriscos\*. Para simplificar as definições de tamanho, os dados descritos foram modelados utilizando como base os tipos de dados que o PostgreSQL apresenta em sua implementação.

Customer			
Atributo	Tipo	Descrição	Exemplo
<b>cpf</b>	char(11)	CPF do cliente, armazenado apenas os números, sem caracteres de espaçamento.	12312312310
<b>email</b>	varchar(80)	E-mail do cliente, validado de acordo com a RFC 5322[3]	user@example.com
name	varchar(100)	Nome do cliente, com no máximo 100 caracteres.	Fulano de Tal
passwd	varchar(35)	Senha do cliente armazenada em hash md5.	5ba0d4762135f9dbb8867955e413423a

A **entidade Customer** representa todos os clientes em questão. O cliente é o ator principal do sistema, e realiza todas as ações de seleção e compra no sistema.

Acquirement			
Atributo	Tipo	Descrição	Exemplo
quantity	smallint	Quantidade de cupons que cada cliente tem, para cada cupom.	4

O **relacionamento Acquirement** representa o relacionamento que os clientes (**Client**) têm com seus cupons (**Coupon**), sempre que o sistema central do iFood dá cupons ao usuário, sua quantidade é armazenada nesse relacionamento.

Coupon			
Atributo	Tipo	Descrição	Exemplo
name	varchar(10)	Nome único do cupom, um identificador de até 10 caracteres.	bestBuy02
discountValue	numeric(5,2)	Valor numérico, em reais, a ser descontado no pedido final do cliente. Limitado a R\$999,99	8.50
validity	timestamp	Data e hora em que o cupom expira sua validade.	2020-01-08 04:05:06 -3:00 *

A **entidade Coupon** representa cada cupom que um usuário pode ter.

\* Exemplo de timestamp traduzido para um exemplo humanamente legível, de acordo com a [documentação](#) do PostgreSQL.

PaymentMethod
---------------

O **relacionamento Payment Method** representa a posse de cartões (**Card**) de um cliente (**Client**).

Card			
Atributo	Tipo	Descrição	Exemplo
cardNumber	varchar(19)	Identificador do cartão de crédito	1234 1234 1234 1234
cardSecNumber	char(3)	Identificador do número de segurança do cartão.	123
cardValidity	date	Data em que o cartão expira sua validade.	2020-01-08
cardName	varchar(100)	Nome conforme escrito no cartão do cliente.	FULANO D. TAL

A **entidade Card** representa os cartões que cliente pode cadastrar para usar em seus pagamentos.



### Attendance

O **relacionamento Attendance** representa a possibilidade de comparecimento de um cliente (**Client**) em um endereço cadastrado (**Address**), para que ele possa receber o pedido que fez.

Address			
Atributo	Tipo	Descrição	Exemplo
street	varchar(100)	Nome da Rua de um endereço.	Av. Cel. Lucas de Oliveira
number	varchar(6)	Número de um endereço	1234
city	varchar(50)	Cidade de um endereço	Porto Alegre
uf	char(2)	Unidade federativa do endereço	RS
comp	varchar(50)	Complemento do endereço, pode ser número de apartamento, descrição visual, etc.	Apto. 38

A **entidade Address** representa um endereço cadastrado no sistema. Esse endereço servirá para definir os locais em que estão os clientes e restaurantes, e pode ser comparado em suas respectivas relações, para que se possa aproximar tempos de entrega de cada restaurante em cada cliente.

### Request

A **entidade Request** representa a sacola/pedido do usuário, e é preenchido com os itens de compra que o cliente escolher.

### Carry

O **relacionamento Carry** representa a relação que as comidas do pedido (**Meal**) tem com o pedido (**Request**) em si.

Item			
Atributo	Tipo	Descrição	Exemplo
quantity	smallint	Quantidade de pratos que um item contém	3

A **entidade Item** representa os itens de um pedido do cliente e se tiver, sua personalização do pedido. Essa entidade permite que um cliente possa fazer pedidos que contenham o mesmo prato (**Meal**), mas que os pratos tenham diferentes adicionais (**Extra**).

Meal			
Atributo	Tipo	Descrição	Exemplo
name	varchar(80)	Nome de um prato de comida.	Xis Salada
description	varchar(1000)	Descrição do prato de comida, em até 1000 caracteres.	"Lorem ipsum dolor sit amet, consectetur...
price	numeric(5,2)	Preço de um prato, em reais, limitado em até R\$999,99.	13.50
prepTime	smallint	Tempo médio de preparo do prato, em minutos, após confirmado o pedido.	35

A **entidade Meal** representa os pratos que o cliente pode comprar.

Addition			
Atributo	Tipo	Descrição	Exemplo
quantity	smallint	Quantidade do adicional colocado na comida.	2

O **relacionamento Addition** representa a adição de um adicional (**Extra**) em uma quantidade de pratos (**Item**) do cliente.

Extra			
Atributo	Tipo	Descrição	Exemplo
name	varchar(80)	Nome do adicional colocado na comida	Queijo Prato

price	numeric(5,2)	Preço, em reais, da instância de adicional colocado na comida, limitado em R\$999,99	2.00
-------	--------------	--	------

A **entidade Extra** representa os adicionais que podem ser colocados no prato de um cliente.

### Category

O relacionamento Category representa a relação de cozinhas/categorias de comida (**Cousine**) com pratos (**Meal**) que o cliente escolhe.

Cousine			
Atributo	Tipo	Descrição	Exemplo
cousineName	varchar(50)	Nome da cozinha/categoria que a comida se encaixa.	Italiana

A **entidade Cousine** representa as categorias e cozinhas com as quais um prato pode ser relacionado, é parte fundamental do sistema pois é utilizada para filtrar resultados de pratos na busca do cliente.

### Preparation

O **relacionamento Preparation** representa o relacionamento de cada prato (**Meal**) com o restaurante (**Restaurant**) que o prepara.

Restaurant			
Atributo	Tipo	Descrição	Exemplo
cnpj	varchar(14)	CNPJ, em números, sem caracteres especiais, da empresa que faz a comida	14281714000120

name	varchar(100)	Nome oficial da empresa que faz a comida	Augusto Bardini LTDA.
popName	varchar(100)	Nome fantasia da empresa que faz a comida.	Torradas dos Guri.

A entidade **Restaurant** representa os restaurantes cadastrados no sistema.

### Residence

O **relacionamento Residence** representa a relação de cada restaurante (**Restaurant**) com o endereço (**Address**) em que reside.

### Purchase

O **relacionamento Purchase** representa a o uso de um cartão sendo para finalizar um pedido (**Finished Order**).

Finished Order			
Atributo	Tipo	Descrição	Exemplo
onMoneyPayment	bool	Indicativo de o pedido ser pago em dinheiro ou cartão.	1
orderDateTime	timestamp	Data e hora em que o pedido foi feito.	2020-01-08 04:05:06 -3:00 *
deliveredDateTime*	timestamp	Data e hora em que o pedido foi entregue.	2020-01-08 04:05:06 -3:00 *
payed	bool	Indicativo de o pedido já ter sido pago ou não.	0
rating*	smallint	Nota, de 1 a 5, que o cliente pode conferir ao pedido.	5

A entidade **Finished Order** é uma **generalização de Request**, e representa um pedido após a confirmação do cliente.

### Dispatch

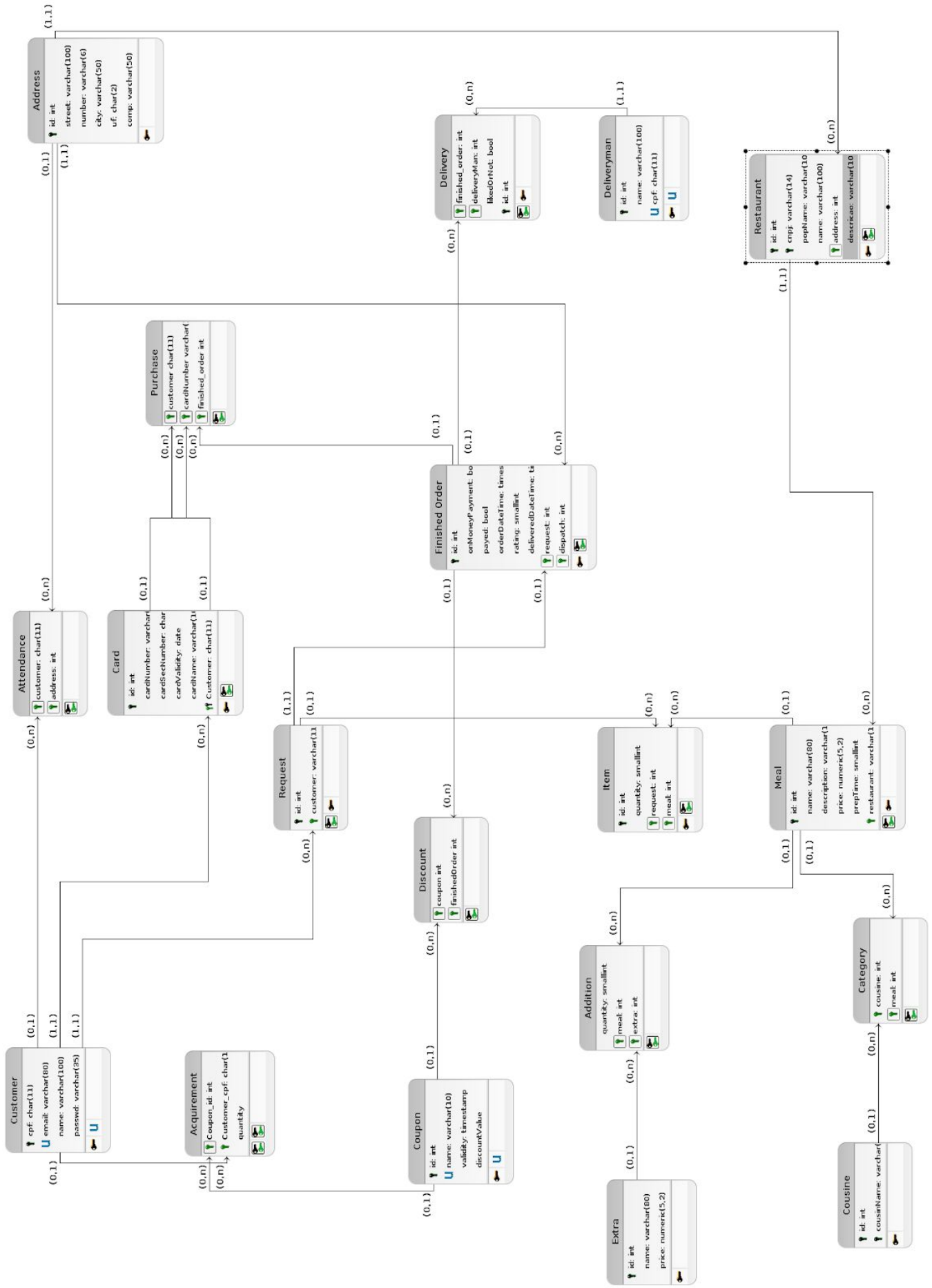
O **relacionamento Dispatch** representa a relação do pedido (**Request**) com o endereço de entrega (**Address**).

Delivery			
Atributo	Tipo	Descrição	Exemplo
likedOrNot*	bool	Indicativo de o cliente ter gostado ou não da entrega feita pelo entregador.	1

O **relacionamento Delivery** representa a relação do pedido (**Request**) com o entregador (**DeliveryMan**).

DeliveryMan			
Atributo	Tipo	Descrição	Exemplo
name	varchar(100)	Nome do entregador, em até 100 caracteres.	Epaminondas Emanuel
cpf	char(11)	CPF do entregador, armazenado apenas os números, sem caracteres de espaçamento.	12312312310

A **entidade Deliveryman** representa os entregadores que fazem o trajeto para entregar os itens comprados ao cliente.



# Descrição do Mapeamento

Na descrição do mapeamento realizado, leva-se em conta que:

- Entidades do ER mapeadas diretamente tornam-se tabelas com o mesmo nome do esquema lógico.
- Atributos do ER mapeados diretamente tornam-se atributos de sua respectiva tabela no esquema lógico.

- Entidade **Customer** mapeada diretamente
  - Atributo cpf mapeado diretamente, obrigatório, chave primária, pois é o identificador único que representa uma pessoa no brasil, e serve aos propósitos do sistema.
  - Atributo email mapeado diretamente, obrigatório, único
  - Atributo name mapeado diretamente, obrigatório,
  - Atributo passwd mapeado diretamente, obrigatório
- Entidade **Coupon** mapeada diretamente
  - Atributo id adicionado como chave primária
  - Atributo name mapeado diretamente, obrigatório, único
  - Atributo discountValue mapeado diretamente, obrigatório
  - Atributo validity mapeado diretamente, obrigatório
- Relacionamento **Acquirement** mapeado para tabela própria, por ser relacionamento N para N.
  - Adicionado Coupon\_id de chave estrangeira para Coupon(id),
  - Adicionado Customer\_cpf de chave estrangeira para Customer(cpf),
  - Atributo quantity mapeado para quantity, obrigatório
- Entidade **Card** mapeada diretamente
  - Atributo id adicionado como chave primária
  - Atributo cardNumber mapeado diretamente, obrigatório
  - Atributo cardSecNumber mapeado diretamente, obrigatório
  - Atributo cardValidity mapeado diretamente, obrigatório
  - Atributo cardName mapeado diretamente, obrigatório
  - Relacionamento **PaymentMethod** mapeado para a chave estrangeira Customer, referenciando Customer(cpf), por ser 1 para N

Não foi utilizado o cardNumber como chave primária, pois o mesmo cartão poderia ser utilizado mais de uma vez, por clientes diferentes, decidiu-se então manter múltiplas instâncias do cartão. Então, utilizou-se um id numérico de chave primária e a junção de

cardNumber com Customer tornou-se única, para ser usada de *match* para chaves estrangeiras.

- Entidade **Address** mapeada diretamente
  - Atributo id adicionado como chave primária,
  - Atributo street mapeado diretamente, obrigatório,
  - Atributo number mapeado diretamente, obrigatório,
  - Atributo city mapeado diretamente, obrigatório,
  - Atributo uf mapeado diretamente, obrigatório,
  - Atributo comp mapeado diretamente, obrigatório
- Relacionamento **Attendance** mapeado para tabela própria, por ser relacionamento N para N.
  - Adicionado address de chave estrangeira para Address(id)
  - Adicionado customer de chave estrangeira para Customer(cpf)
- Entidade **Request** mapeada diretamente
  - Atributo id adicionado como chave primária
  - Relacionamento Fill mapeado para a chave estrangeira Customer, referenciando Customer(cpf)
- Entidade **Restaurant** mapeada diretamente
  - Atributo cnpj mapeado diretamente, obrigatório, chave primária, pois é único e serve aos propósitos do sistema.
  - Atributo name mapeado diretamente, obrigatório,
  - Atributo popName mapeado diretamente, obrigatório,
  - Relacionamento **Residence** mapeado para a chave estrangeira address, referenciando Address(id), por ser 1 para N
- Entidade **Meal** mapeada diretamente
  - Atributo id adicionado como chave primária
  - Atributo name mapeado diretamente, obrigatório
  - Atributo description mapeado diretamente, obrigatório
  - Atributo price mapeado diretamente, obrigatório
  - Atributo prepTime mapeado diretamente, obrigatório
  - Relacionamento **Preparation** mapeado para a chave estrangeira restaurant, referenciando Restaurant(cnpj), por ser 1 para N
- Relacionamento **Item** mapeada para tabela Item
  - Atributo id adicionado como chave primária
  - Adicionado request de chave estrangeira para Request(id), por conta de entidade associativa ser um relacionamento N para N
  - Adicionado meal de chave estrangeira para Meal(id), por conta de entidade associativa ser um relacionamento N para N
  - Atributo quantity mapeado diretamente, obrigatório



- Entidade **Extra** mapeada diretamente
  - Atributo id adicionado como chave primária
  - Atributo name mapeado diretamente, obrigatório
  - Atributo price mapeado diretamente, obrigatório
  
- Relacionamento **Addition** mapeado para tabela própria, por ser relacionamento N para N.
  - Adicionado meal de chave estrangeira para Meal(id),
  - Adicionado extra de chave estrangeira para Extra(id),
  - Atributo quantity mapeado diretamente, obrigatório
  
- Entidade **Cousine** mapeada diretamente
  - Atributo id adicionado como chave primária
  - Atributo cousinName mapeado diretamente, obrigatório, único
  
- Relacionamento **Category** mapeado para tabela própria, por ser relacionamento N para N.
  - Adicionado meal de chave estrangeira para Meal(id),
  - Adicionado cousine de chave estrangeira para Cousine(id)
  
- Entidade **DeliveryMan** mapeada diretamente
  - Atributo id adicionado como chave primária
  - Atributo name mapeado diretamente, obrigatório
  - Atributo cpf mapeado diretamente, obrigatório, único
  
- Relacionamento **Delivery** mapeado para tabela própria, por ter atributos, apesar de ser 1 para N.
  - Atributo id adicionado como chave primária
  - Adicionado finished\_order de chave estrangeira para Meal(id),
  - Adicionado deliveryMan de chave estrangeira para Cousine(id)
  - Atributo likedOrNot mapeado diretamente, opcional
  
- Entidade **Finished\_Order** mapeada diretamente
  - Atributo id adicionado como chave primária
  - Atributo onMoneyPayment mapeado diretamente, obrigatório
  - Atributo orderDateTime mapeado diretamente, obrigatório
  - Atributo deliveredDateTime mapeado diretamente, opcional
  - Atributo payed mapeado diretamente, obrigatório
  - Atributo rating mapeado diretamente, opcional
  - Relacionamento **Dispatch** mapeado para a chave estrangeira dispatch, referenciando Address(id), por ser 1 para N
  - Adicionado request de chave estrangeira para Request(id), para fazer o mapeamento da entidade mãe **Request** para a entidade especializada **Finished\_Order**

- Relacionamento **Purchase** mapeado para tabela própria, por ser um relacionamento opcional, apesar de ser 1 para N. Dessa forma, não se armazena muitos atributos nulos e mantém-se a organização e integridade do sistema.
  - Adicionado customer de chave estrangeira para Card(customer)
  - Adicionado cardNumber de chave estrangeira para Card(cardNumber)
  - Adicionado finished\_order de chave estrangeira para Finished\_Order(id)