

Aplicação de SOLID em um sistema Web para intermediação de compra e venda de comida na UNESP

Orientador: Prof. Dr. Higor Amario de Souza
Julio Cesar Benelli Varela
RA 181024594



Sumário

- Introdução
- Objetivos
- Fundamentação
- Desenvolvimento
- Conclusão



Introdução

- Permanência Estudantil
 - Se manter na universidade durante a graduação pode ser uma grande dificuldade.
 - Mudar de cidade
 - Longas jornadas
 - Dificuldade em obter auxílios e bolsas, muitas vezes insuficiente
- Nesse contexto, a venda de alimentos é uma atividade que surge para dar suporte financeiro
 - Alunos
 - Projetos de extensão
- Essa atividade é feita de forma desorganizada e descentralizada.



Introdução

- Manutenibilidade de código
 - “A natureza do software é mutante”. (PRESSMAN; MAXIM, 2016)
 - “Qualquer programador provavelmente já foi significativamente desacelerado pelo código confuso de outra pessoa” (MARTIN, 2008)
 - Necessidade comum de manutenção e extensão de código
 - Importância de práticas de desenvolvimento facilitadoras



Justificativa

- É uma ferramenta de contribuição para a comunidade.
- Proporcionar um ambiente unificado e organizado para facilitar a compra e venda de comida.
- O software foi desenvolvido seguindo boas práticas de desenvolvimento visando a facilidade de manutenção.
- O sistema será disponibilizado como código aberto e poderá ser facilmente estendido.



Objetivos

Gerais

- Desenvolver um sistema Web para facilitar e centralizar a compra e venda de alimentos no campus da UNESP e colaborar com a permanência estudantil.
- Desenvolver o sistema seguindo boas práticas de desenvolvimento e disponibilizar em código aberto para contribuição da comunidade.

Específicos

- Levantar e analisar requisitos relevantes ao sistema.
- Aplicar metodologias de Engenharia de Software como Padrões de Projeto e Arquitetura de Software.
- Utilizar ferramentas modernas de desenvolvimento como um Framework Web e MVC.
- Analisar o código desenvolvido e fazer a refatoração para que se encaixe nos conceitos de SOLID e Clean Code.



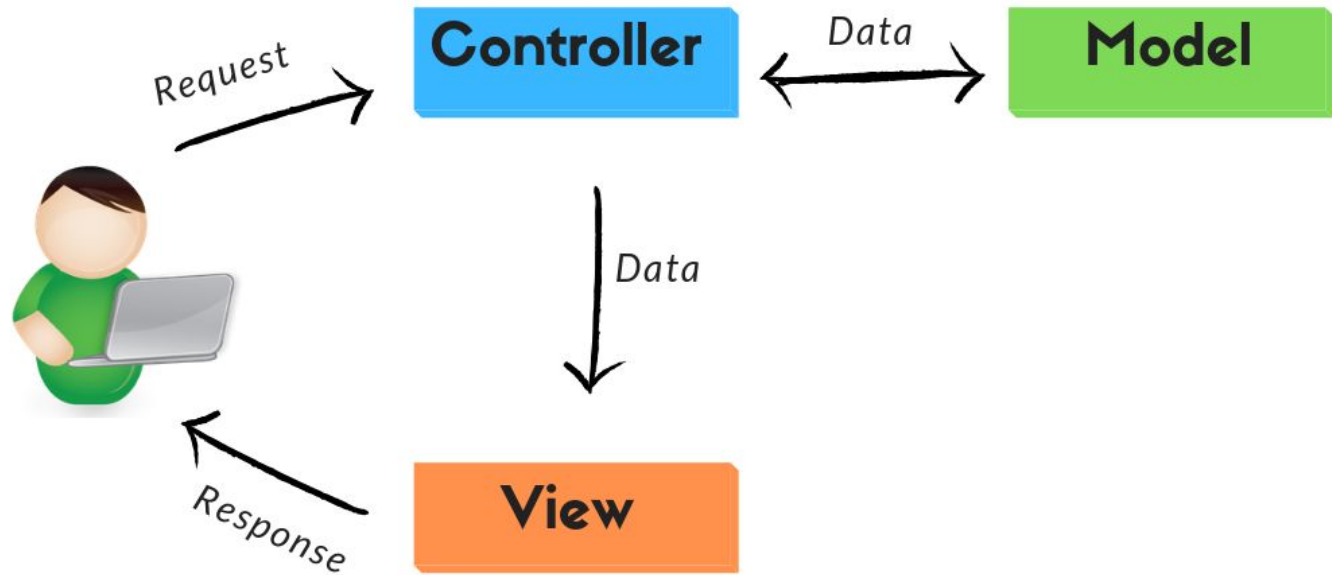
Fundamentação



Arquitetura de Software

- Forma do sistema
 - Divisão, disposição e interação entre componentes.
- Qual é a arquitetura de um sistema?
- Model View Controller (MVC)
 - Model: lógica de negócio - Entidades
 - View: interface gráfica
 - Controller: tratamento de eventos, aceita requisições e envia respostas.

Arquitetura MVC





SOLID

Forma de escrever código visando manutenibilidade, extensibilidade e adaptabilidade.

- Como organizar as classes e métodos
- Código adaptável à mudanças
- Código compreensível e claro
- Criação de componentes reutilizáveis



SOLID

- Princípio de Responsabilidade Única (SRP)
 - Separação de responsabilidades
 - Uma classe deve ter apenas um motivo para mudar
 - Banco de dados - interface gráfica

Código 1 – Exemplo de código PHP que não segue o SRP

```
class Employee {  
  
    private $id;  
    private $name;  
  
    public function __construct($id, $name) {  
        $this->id = $id;  
        $this->name = $name;  
    }  
  
    public function calculatePay() {  
        // Logica para calcular o salario do empregado  
    }  
  
    public function reportHours() {  
        // Logica para reportar as horas trabalhadas  
    }  
  
    public function save() {  
        // Logica para salvar informacoes do empregado  
        // no banco de dados  
    }  
}
```

Código 2 – Exemplo de código refatorado PHP seguindo o SRP

```
class Employee {
    private $id;
    private $name;

    public function __construct($id, $name) {
        $this->id = $id;
        $this->name = $name;
    }

    public function calculatePay() {
        // Logica para calcular o salario do empregado
    }
}


class EmployeeReport {
    public function reportHours(Employee $employee) {
        // Logica para reportar as horas trabalhadas
    }
}

class EmployeeRepository {
    public function save(Employee $employee) {
        // Logica para salvar informacoes do empregado
        // no banco de dados
    }
}
```

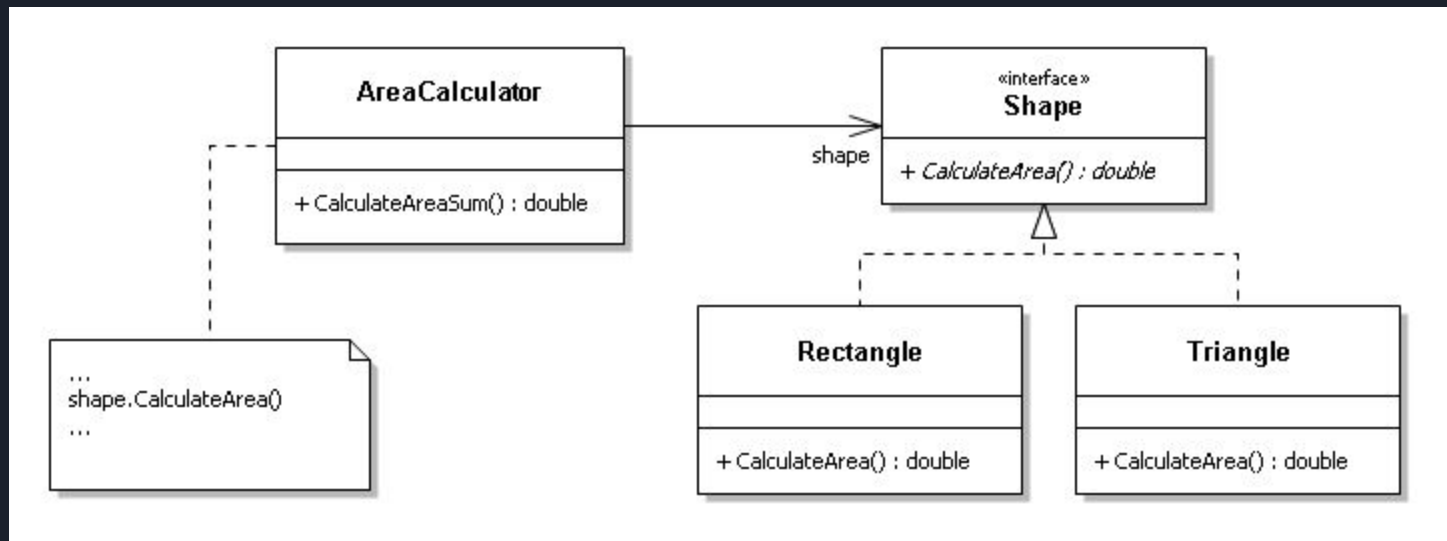


SOLID

- Princípio Aberto-Fechado (OCP)
 - Aberto para extensão, fechado para modificação
 - Evitar que o código testado seja alterado
 - Encoraja o uso de interfaces



```
void DrawAllShapes(ShapePointer list[], int n) {
    int i;
    for (i=0; i<n; i++) {
        struct Shape* s = list[i];
        switch (s->itsType) {
            case square:
                DrawSquare((struct Square*)s);
                break;
            case circle:
                DrawCircle((struct Circle*)s);
                break;
        }
    }
}
```

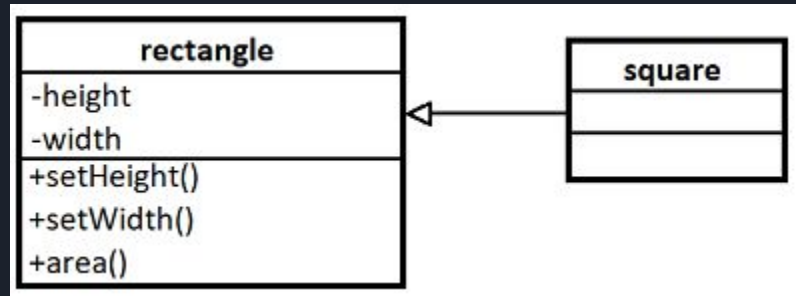




SOLID

- Princípio de Substituição de Liskov (LSP)
 - Capacidade de objetos de uma superclasse serem substituídos por objetos de uma subclasse.
- Herança: objeto A estende objeto B, portanto, A é um B.
- LSP: A deve ser substituível por B.

Problema do quadrado e do retângulo - violação do LSP

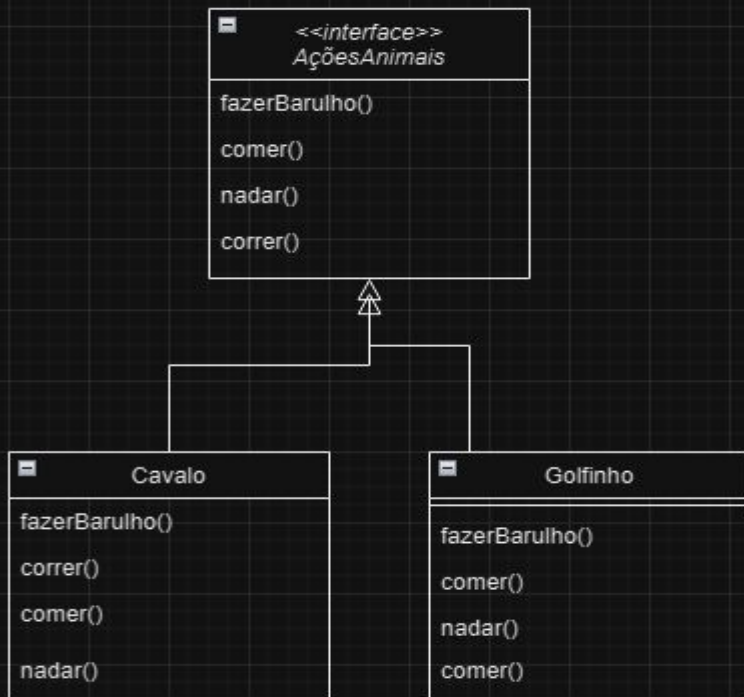


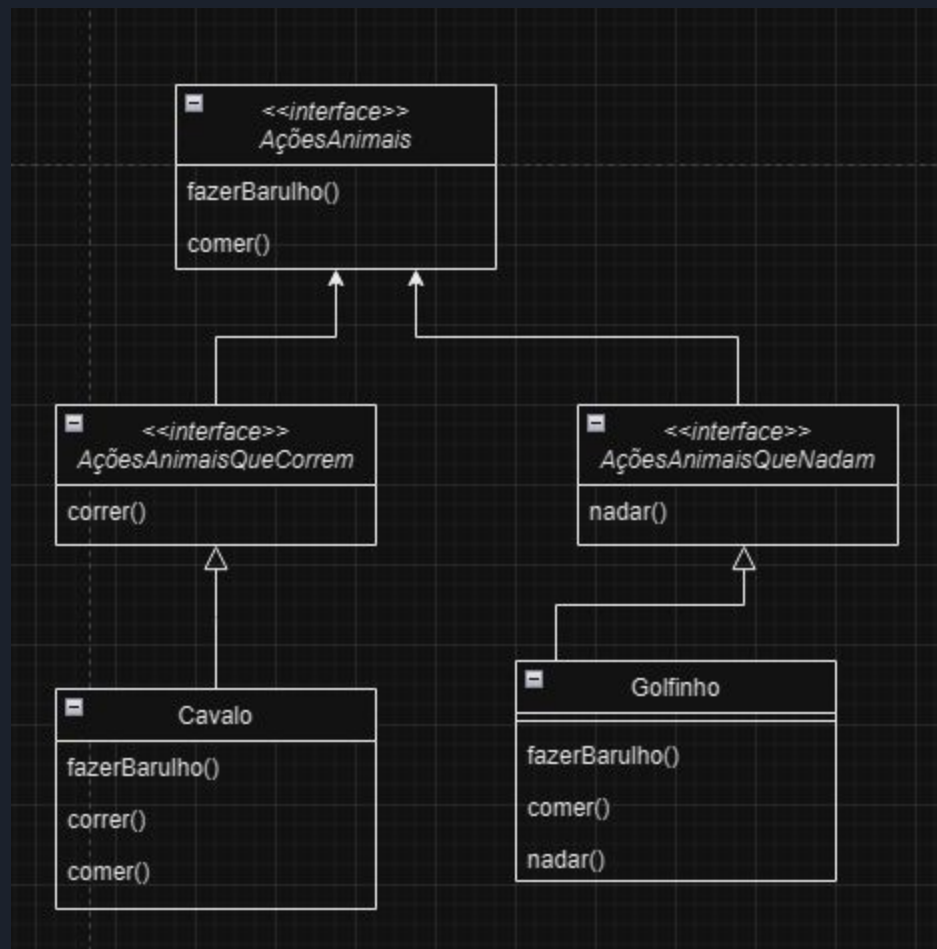


SOLID

- Princípio da Segregação de Interfaces (ISP)
 - Garantir que uma classe não seja forçada a implementar métodos que não utilizará

Violação do LSP

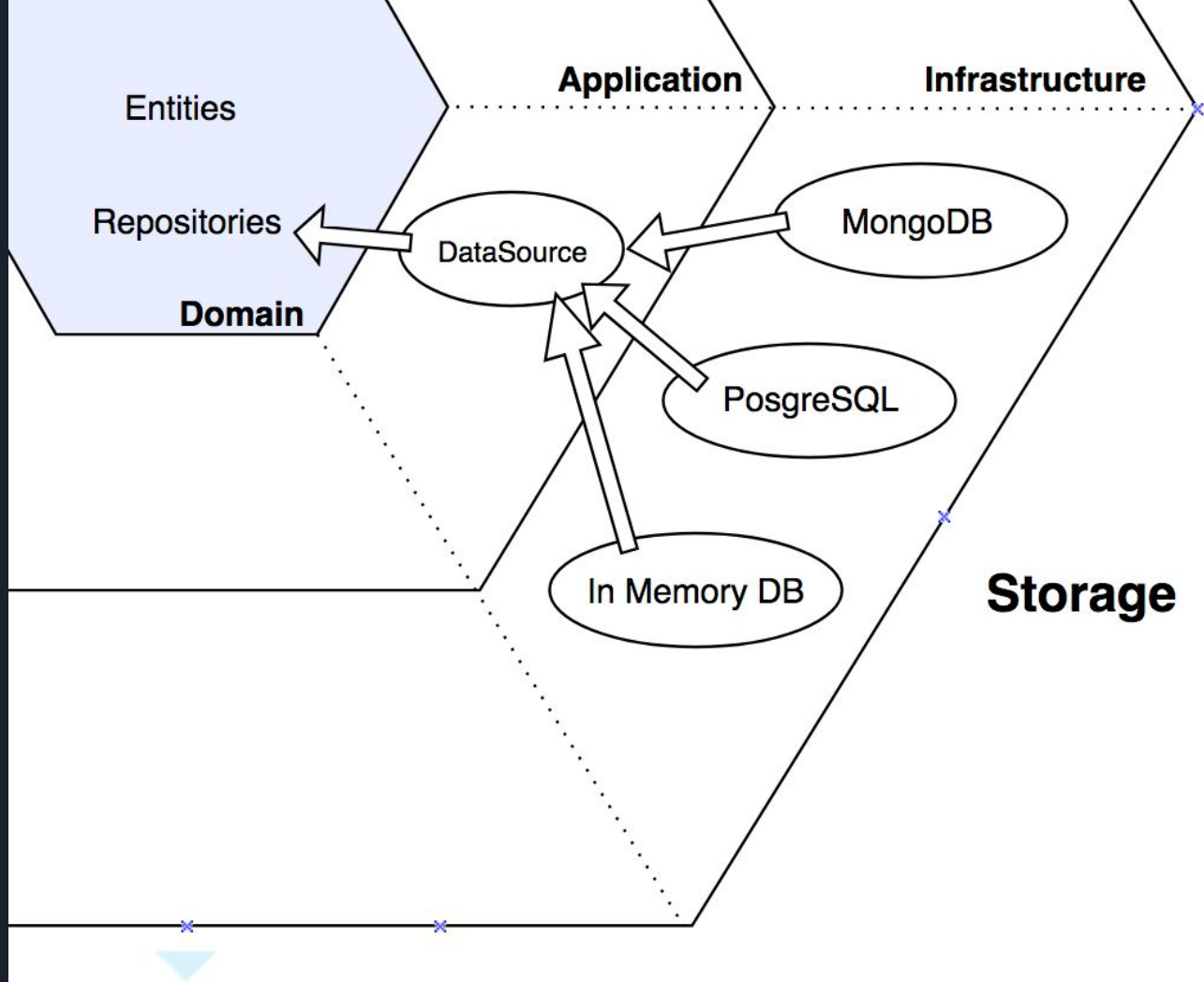






SOLID

- Princípio da Inversão de Dependência (DIP)





Desenvolvimento

Definição de ferramentas:

- **Symfony, PHP**
 - MVC
 - Conhecimento prévio
 - Geração de código
- **Bootstrap**
 - Componentes fáceis de usar
 - Responsividade
- **Git**
 - Versionamento
 - Compartilhamento

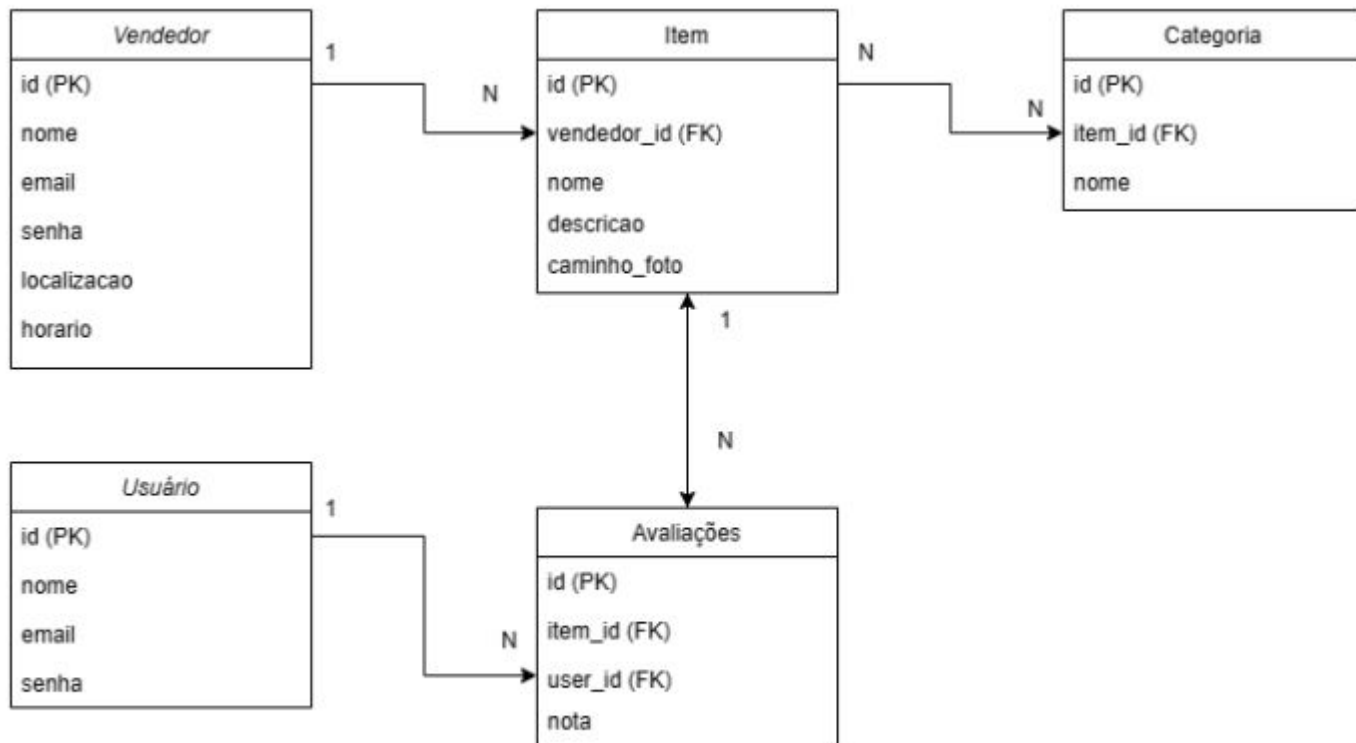


Definição dos requisitos

Entender as necessidades do sistema:

- Vendedor
 - Autenticação
 - Criação do perfil de vendedor
 - Criação de anúncios
- Usuário
 - Descrição dos alimentos
 - Contato do vendedor
 - Responsividade
 - Filtros

Modelagem de dados



Fonte : Elaborado pelo autor.



Implementação das funcionalidades

Login



[Login](#) [Registrar](#) [Pesquisar Comidas](#)

Log in!

Email

email@email.com

Password

Sign in

❤ Contribuir com o projeto no [GitHub](#)

Perfil de vendedor



[Logout](#) [Criar Anúncio](#) [Criar Perfil de Vendedor](#) [Pesquisar Comidas](#)

Criar Perfil de Vendedor

Nome

Localização

Disponibilidade

Whatsapp

Save

Voltar

♥ Contribuir com o projeto no [GitHub](#)

Criação de anúncio (upload de imagem)



[Logout](#) [Criar Anúncio](#) [Criar Perfil de Vendedor](#) [Pesquisar Comidas](#)

Criar Anúncio

Name

Description

Seller

vendedor

Save

Imagem do Anúncio

Escolher Arquivo

Nenhum arquivo escolhido

Price

Category

Doces

[Voltar](#)

Navegação



[Logout](#) [Criar Anúncio](#) [Criar Perfil de Vendedor](#) [Pesquisar Comida](#)

Filtrar por categoria

[Doce](#)[Salgados](#)[Veganos](#)[Vegetarianos](#)

Comidas



Cookies de Chocolate Duplo

R\$4.00

Cookies crocantes por fora e macios por dentro, repletos de pedaços generosos de chocolate meio amargo e branco. Cada mordida é uma explosão de sabor de chocolate duplo que derreterá na boca.

Vendedor: [Vendedor](#)

[Ver](#)

Sanduíche Natural de Frango

R\$8.00

Um sanduíche saudável e delicioso recheado com tiras de frango grelhado, alface fresca, tomate suculento e um toque de maionese caseira. Embalado em um pão integral macio, é uma opção nutritiva e satisfatória para o almoço.

Vendedor: [Vendedor](#)

[Ver](#)

Bolo no Pote de Red Velvet

R\$10.00

Camadas de bolo de veludo vermelho úmido e macio, alternadas com camadas generosas de creme de queijo com baunilha. Montado em um pote para ser saboreado em porções individuais, é a combinação perfeita de doçura e textura.

Vendedor: [Felipe](#)

[Ver](#)[Editar](#)

Filtro por categoria

[Logout](#) [Criar Anúncio](#) [Criar Perfil de Vendedor](#) [Pesquisar Comidas](#)

Filtrar por categoria

[Doces](#)[Salgados](#)[Veganos](#)[Vegetarianos](#)

Comidas



Sanduíche Natural de Frango

R\$8.00

Um sanduíche saudável e delicioso recheado com tiras de frango grelhado, alface fresca, tomate suculento e um toque de maionese caseira. Embalado em um pão integral macio, é uma opção nutritiva e satisfatória para o almoço.

Vendedor: [Vendedor](#)

[Ver](#)

Página do anúncio

[Login](#) [Registrar](#) [Pesquisar Comidas](#)

Detalhes do Anúncio

| | |
|------------|---|
| Id: | 1 |
| Nome: | Cookies de Chocolate Duplo |
| Descrição: | Cookies crocantes por fora e macios por dentro, repletos de pedaços generosos de chocolate meio amargo e branco. Cada mordida é uma explosão de sabor de chocolate duplo que derreterá na boca. |
| Imagem: |  |
| Preço: | 4 |
| Categoria: | Doces |

[Voltar](#)

Perfil do vendedor



[Login](#) [Registrar](#) [Pesquisar Comidas](#)


Detalhes do Vendedor

vendedor

Localização: FEB

Disponibilidade: de manhã

WhatsApp: 1499999999

 Chamar no WhatsApp

[Back to list](#)

♥ Contribuir com o projeto no [GitHub](#)

Criar Anúncio

Name

Description

Seller

vendedor

Save

Imagem do Anúncio

Escolher Arquivo

Nenhum arquivo escolhido

Price

Category

Doces



[Voltar](#)



Login

Registrar

Pesquisar Comidas

Filtrar por categoria

Doces

Salgados

Veganos

Vegetarianos

Comidas



**Cookies de
Chocolate Duplo**



**Sanduíche Natural
de Frango**



Logout

Criar Anúncio

Criar Perfil de Vendedor

Pesquisar Comidas


Detalhes do Vendedor

vendedor

Localização: FEB

Disponibilidade: de manhã

WhatsApp: 1499999999

 Chamar no WhatsApp

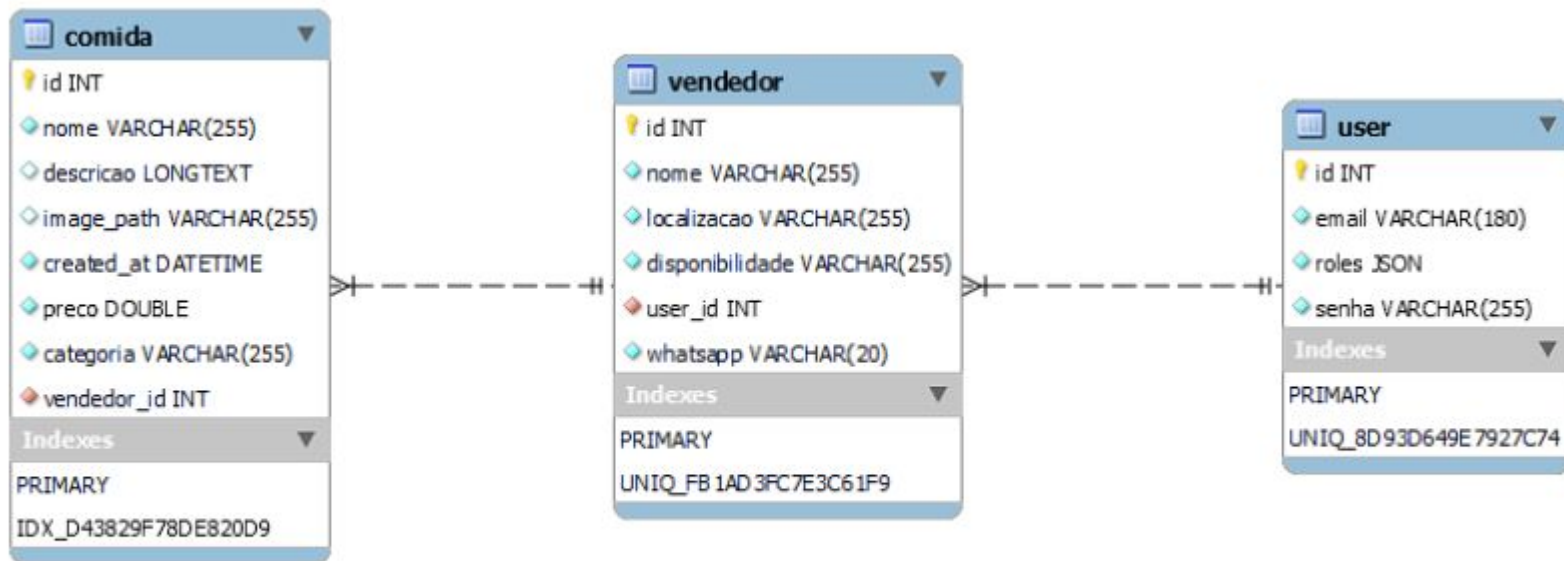
Back to list



Dificuldades no desenvolvimento

- Desenvolvimento de todas as funcionalidades
- Adequação de todos os módulos do sistema ao SOLID

Modelo Entidade Relacionamento final



Fonte : Elaborado pelo autor.



Aplicação dos princípios SOLID

- Criação de classes Repository



Controller -> Repository -> View

```
#[Route('/category/{categoryName}', name: 'app_food_item_by_category', methods: ['GET'])]  
public function indexByCategory(FoodRepository $foodRepository, CategoriesProvider $categoriesProvider, string $categoryName): Response  
{  
    $foods = $foodRepository->findBy(['category' => $categoryName]);  
    $categories = $categoriesProvider->getCategories();  
  
    return $this->render( view: 'food_item/index.html.twig', [  
        'food' => $foods,  
        'categories' => $categories,  
    ]);  
}
```




Método da classe Repository

```
public function findByCategoryName($name): array
{
    return $this->createQueryBuilder( alias: 'f')
        ->join( join: 'f.category', alias: 'c')
        ->where( predicates: 'c.name = :name')
        ->setParameter( key: 'name', $name)
        ->getQuery()
        ->getResult();
}
```

Aplicação de SOLID

- SRP: Lidar com requisições de Food
- OCP: Injeção de dependência permite expandir sem modificar
- ISP: As injeções utilizadas são as necessárias
- DIP: A classe FoodItemController depende de abstrações (construtores)

```
#[Route('/food/item')]
class FoodItemController extends AbstractController
{
    private FoodRepository $foodRepository;
    private CategoriesProvider $categoriesProvider;
    private EntityManagerInterface $entityManager;

    public function __construct(
        FoodRepository $foodRepository,
        CategoriesProvider $categoriesProvider,
        EntityManagerInterface $entityManager
    ) {
        $this->foodRepository = $foodRepository;
        $this->categoriesProvider = $categoriesProvider;
        $this->entityManager = $entityManager;
    }
}
```



Conclusão

- Sistema desenvolvido para centralizar e organizar as vendas
- Ferramentas adequadas para facilitar desenvolvimento
- MVC e SOLID aplicados
- Código de boa manutenção
- Código aberto para contribuição



Trabalhos futuros

- A implementação de um sistema de avaliações
- Melhorias de design das telas e responsividade
- Autenticação restrita ao domínio da UNESP com login Google
- Infraestrutura de hospedagem