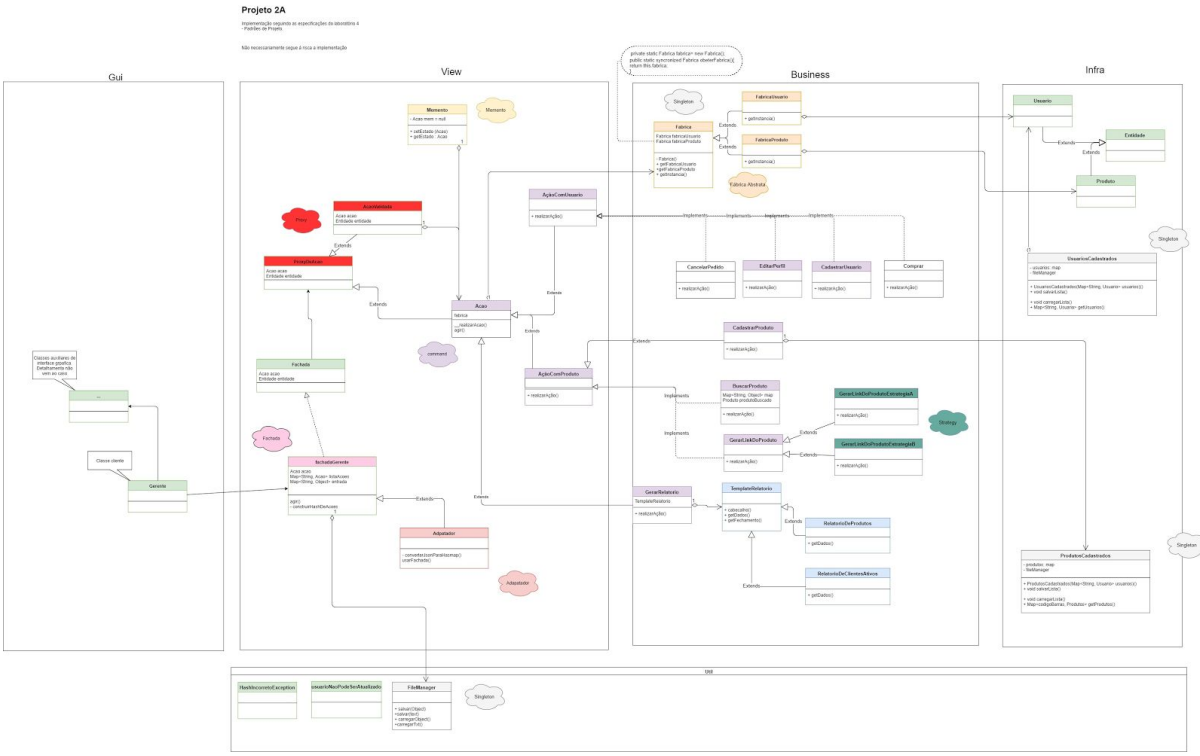
Grupo:

Luiz Henrique Freire Barros
Aline Moura Araújo



[link para melhor visualização]

Padrões Obrigatórios

Template

Representado pela cor azul claro no diagrama.

A classe **business.TemplateRelatorio** é o modelo
As classes **business.RelatorioDeProdutos** e **business.RelatatorioCliente** são as classes concretas

Fábrica Abstrata

Representado pela cor laranja.

A classe **business.Fabrica** é a classe abstrata que fabrica outras fábricas
As classes **business.FabricaProduto** e **business.FabricaUsuario** são as fábricas que produzem as entidades.

Memento

Representado em amarelo.
A classe **view.Memento** representa a classe memento que salva o estado anterior, ele guarda a **ação de editar o perfil de usuário** e essa ação pode ser desfeita.

Fachada

Representada pela cor rosa.
A classe **view.FachadaCliente** é a fachada pelo qual o cliente vai se comunicar com a aplicação.

Adapter

Representada pela cor vermelha.
Vamos supor que uma nova versão de gerar relatório de clientes tenha que ser enviada por JSON em vez de hashmap, a classe **view.Adaptador** faz a conversão. de forma implícita transparente.

Comando

Representado pela cor roxa.
A Classe **view.acao** é a interface de comandos genéricos.
As classes **business.EditarPerfil**, **business.CadastrarUsuario**, **business.CadastrarProduto**, **business.BuscarProduto**, **business.GerarLinkDoProduto** são os comandos concretos.
A **fachada** é o executor.

Padrões extras

Singleton

Representado pela cor cinza
As classes **business.UsuariosCadastrados**, **infra.UsuariosCadastrados**, **infra.ProdutosCadastrados** e **util.FileManager** implementam o singleton

Strategy

Representado pela cor verde escuro
Existem duas maneiras gerar um link do produto, uma com a estratégia A, e outra pela estratégia B. Essas estratégias são as classes **business.GerarLinkDoProdutoEstrategiaA** e **business.GerarLinkDoProdutoEstrategiaB.** As classes foram criadas, mas a implementação não importa no nosso caso (foram colocados apenas prints diferentes)

Proxy

Representado pela cor vermelha escura.
Antes da aplicação fazer alguma ação, é feito uma validação de login e senha do gerente/cliente.
A interface é o **view.ProxyDeAcao**, a implementação real está na **view.Acao**, e o intermediário é o **view.AcaoValidada**

Observações

- As classes em branco não foram implementadas pois não acrescentariam o entendimento do padrão.
- As classes em verde são classes auxiliares que ajudam no funcionamento do projeto.