**Challenge Natura - Sprint 2**Murilo Santini Chequer RM: 550198  
Ronaldo Kozan Júnior RM: 98865  
João Victor Oliveira Avellar RM: 550283  
Francisco Henrique Lima RM: 99545  
Enzo Duarte RM: 552486  
  
**Identificar os objetos envolvidos na aplicação, ou seja, aqueles que serão utilizados na resolução do problema:**  
Classes: Consultora, Cliente, Produto, SuperApp,   
  
**Pedido identificar as classes e seus atributos. Elaborar diagrama de classes; o retângulo dividido em três partes.  
As novas alterações estão em vermelho.**  
  
**abstract Usuario**- id: int,  
- nome: String,   
- endereco: String,  
- idade: int  
+ cadastrarUsuario  
 **Consultora : Usuario**  
- IdConsultora : int,  
- pontuacaoConsultora : int,  
- prodDisponiveis : list  
+ cadastrarUsuario  
  
  
**Cliente : Usuario**  
- idCliente : int,  
- listaConsultorasFavoritas : string  
+ cadastrarUsuario  
  
  
**Produto**  
- idProduto : int,  
- nomeProduto : string,  
- descricaoProduto : string,  
- preco : double,  
- categoria : string,  
- estoque : int

**Pedido**   
- idPedido : int,  
- cliente : Cliente,  
- consultora : Consultora,  
- produtos : list,  
- data : datetime  
  
**SuperAPP**  
clientes : list  
consultoras : list  
produtos : list  
pedidos : list  
‘

**há aplicação de herança? Se tiver, identificar onde. Caso não tenha, apenas informar que não haverá.**Após analisar as nossas classes, vimos que era possível evitar a repetição de código utilizando herença, os atributos nome, idade e endereço se repetiam para Cliente e Consultora, por isso, foi implementada uma classe base Usuário que extendesse para Cliente e Consultora, assim, eviando repetição de código nas classes.

**há aplicação de polimorfismo? Se tiver, identificar onde. Caso não tenha, apenas informar que não haverá.**

Foi utilizado o polimorfismo ao adicionar clientes/consultoras, onde a classe Usuário se tornou abstrata e o método passou a ser obrigatóriamente utilizado nas classes Cliente e Consulta.