Relatório do Trabalho Prático 2

Lista de Compras em Java

Artur Reis - 17552

José Ribeiro - 17062

Desenvolvimento Web e Multimédia – Pós-laboral

Novembro de 2019

|  |
| --- |
| Afirmo por minha honra que não recebi qualquer apoio não autorizado na realização deste trabalho prático. Afirmo igualmente que não copiei qualquer material de livro, artigo, documento web ou de qualquer outra fonte exceto onde a origem estiver expressamente citada. |

Artur Reis - 17552

José Ribeiro - 17062

Índice

[INTRODUÇÃO 3](#_Toc25611649)

[Conceitos sobre GIT 3](#_Toc25611650)

[Conceitos sobre Java 3](#_Toc25611651)

[CLASSES - utilizador 5](#_Toc25611652)

[CLASSES - address 7](#_Toc25611653)

[CLASSES – category 8](#_Toc25611654)

[CLASSES – product 9](#_Toc25611655)

[CLASSES – shopping list 10](#_Toc25611656)

[CLASSES – prints Código 11](#_Toc25611657)

[Conclusão 12](#_Toc25611658)

[Bibliografia 13](#_Toc25611659)

## INTRODUÇÃO

Neste trabalho foi-nos pedido para implementar em JAVA uma aplicação Android de uma lista de compras.

Inicialmente para que o projeto fosse desenvolvido da maneira correta tivemos de modelar em UML todas as classes necessárias para implementar a lista de compras. Depois disso passamos para a parte do Código ( Android Studio ) onde vamos colocar todas as classes, referir as suas variáveis e dar valores a essas mesmas e aprender a trabalhar com elas como vamos mostrar de seguida.

## Conceitos sobre GIT

Git é um sistema de controle de versão de arquivos. Através deles podemos desenvolver projetos na qual diversas pessoas podem contribuir simultaneamente no mesmo, editando e criando novos arquivos e permitindo que os mesmos possam existir sem o risco de suas alterações serem sobrescritas.

Se não houver um sistema de versão, imagine o caos entre duas pessoas abrindo o mesmo arquivo ao mesmo tempo. Uma das aplicações do git é justamente essa, permitir que um arquivo possa ser editado ao mesmo tempo por pessoas diferentes. Por mais complexo que isso seja, ele tenta manter tudo em ordem para evitar problemas para nós desenvolvedores.

## 

## Conceitos sobre Java

POO - Programação Orientada a Objetos. Este tipo de programação trata os elementos da linguagem de maneira semelhante aos objetos reais.

PACKAGES – É semelhante ao conceito de biblioteca de funções, sendo que um package é um conjunto de classes, que ficam num diretório com o mesmo nome do pacote.

CLASSES – É um conjunto de objetos com características comuns. Uma classe é como um modelo para a criação de objetos, que tem as mesmas características da classe à qual pertence.

OBJETOS – É um elemento de uma classe. Temos uma classe ”gatos” que é formada pelos objetos “gato”. Todos os objetos têm em comum o fato de serem gatos (mesma espécie), mas podem ter características diferentes entre si. Objetos tem variáveis e métodos como também classes.

INSTÂNCIA – Uma instância de uma classe é um novo objeto criado dessa classe, com o operador new. Instanciar uma classe é criar um novo objeto do mesmo tipo dessa classe. Uma classe somente poderá ser utilizada após ser instanciada.

MÉTODOS – Os métodos representam os estados e ações dos objetos e classes.

VARIÁVEIS – As variáveis e constantes representam as características dos objetos.

# CLASSES - utilizador

***Utilizador*** – Primeiramente foi instanciada a classe em um diagrama UML passando sim depois para o código como segue nas imagens. Esta classe User é composta por cinco variáveis do tipo “string” e uma variável do tipo “inteiro” (id). Nas quais foram feitos get&set para poder entregar valores as mesmas

Estes campos são as informações que foram achadas necessários pertencer a um utilizador.

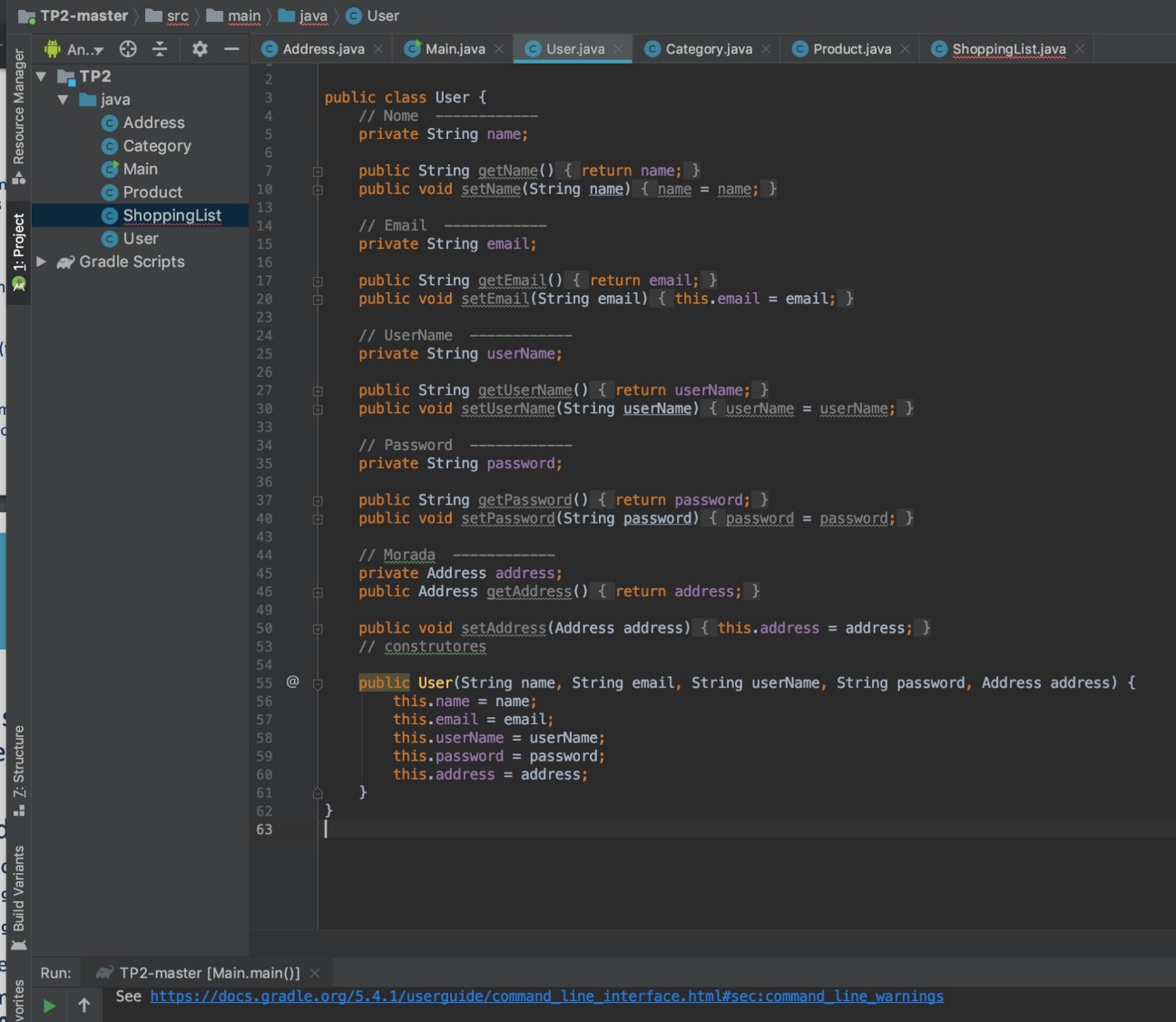
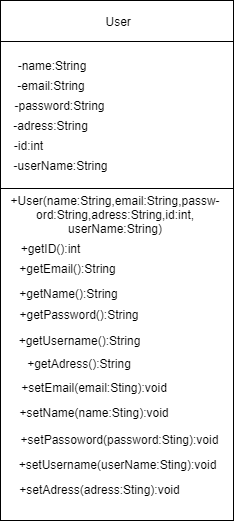
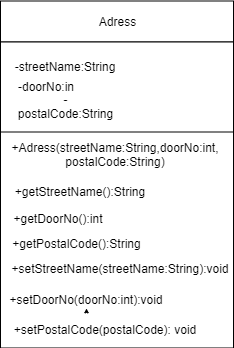
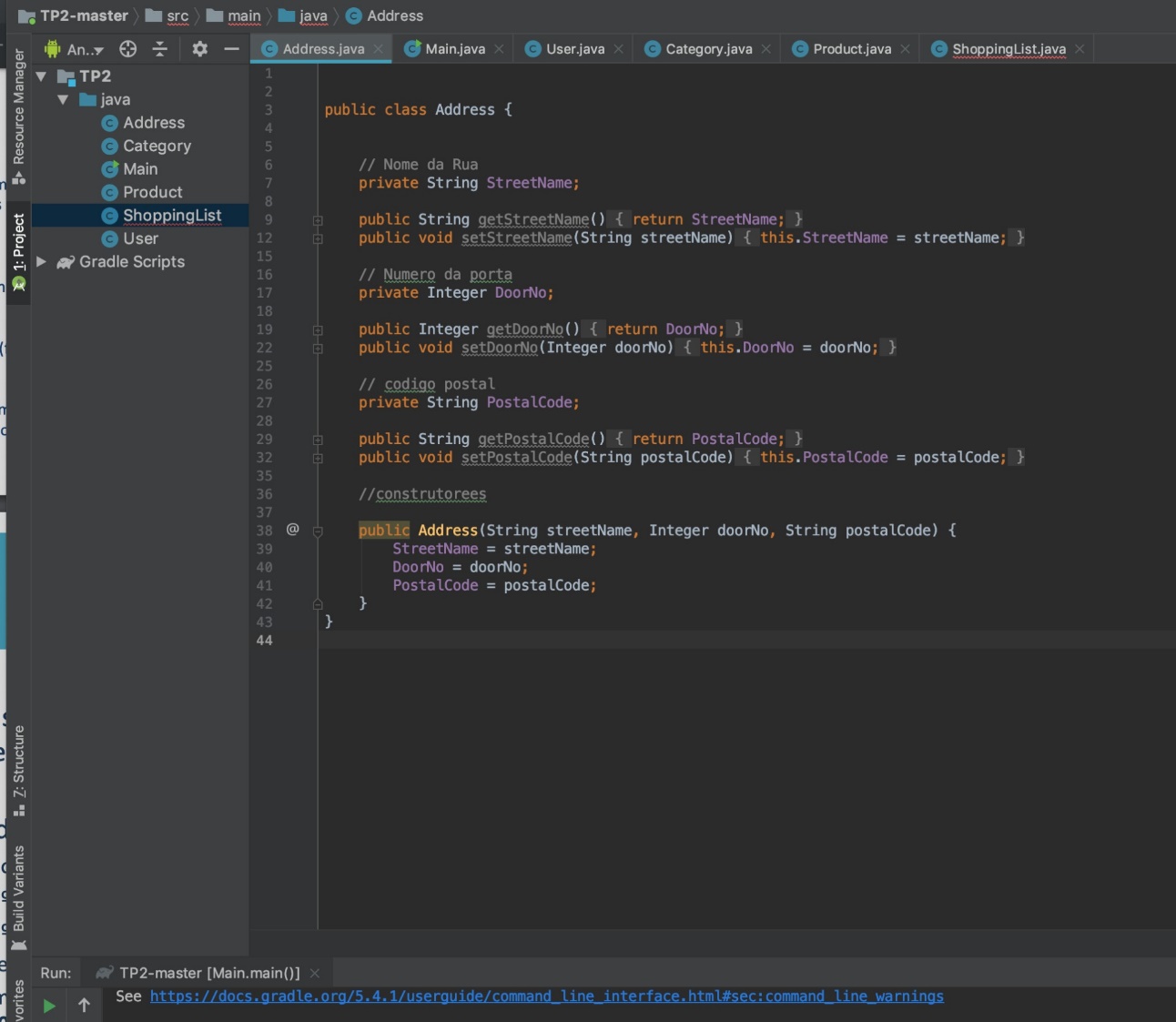


Figure 2 UML - User

# CLASSES - address

***Address*** – Primeiramente foi instanciada a classe em um diagrama UML. Esta classe Address é composta por duas variáveis do tipo “string” e uma variável do tipo “inteiro” (doorNo). Está mesma classe é uma variável da classe User.

Estes campos são as informações que foram achadas necessários conter na classe Address.

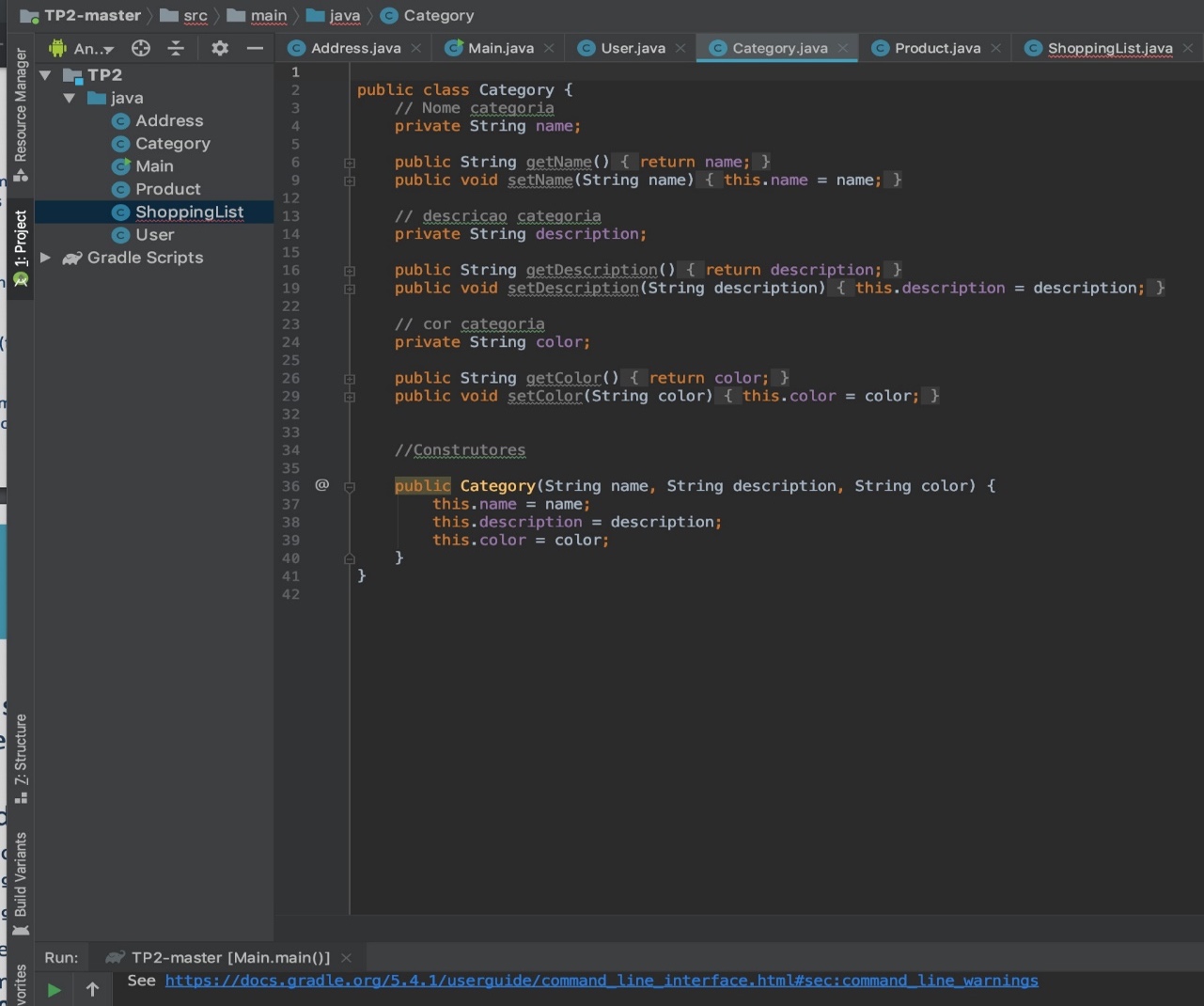
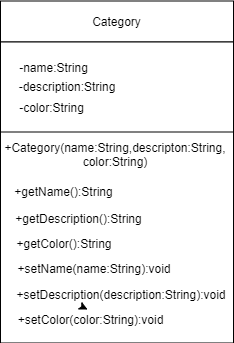
******

# CLASSES – category

***Category*** – Primeiramente foi instanciada a classe em um diagrama UML. Esta classe Category é composta por três variáveis do tipo “string”.

Estes campos são relativos as categorias que cada produto que poderão existir, como por exemplo se o produto for um “bife” a categoria do mesmo vai ser “carnes” ou algo desse género.

A categoria tem de ter um nome, uma descrição, e uma cor.

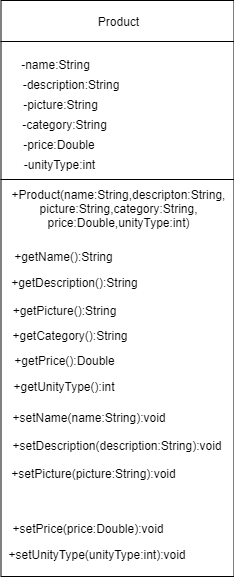
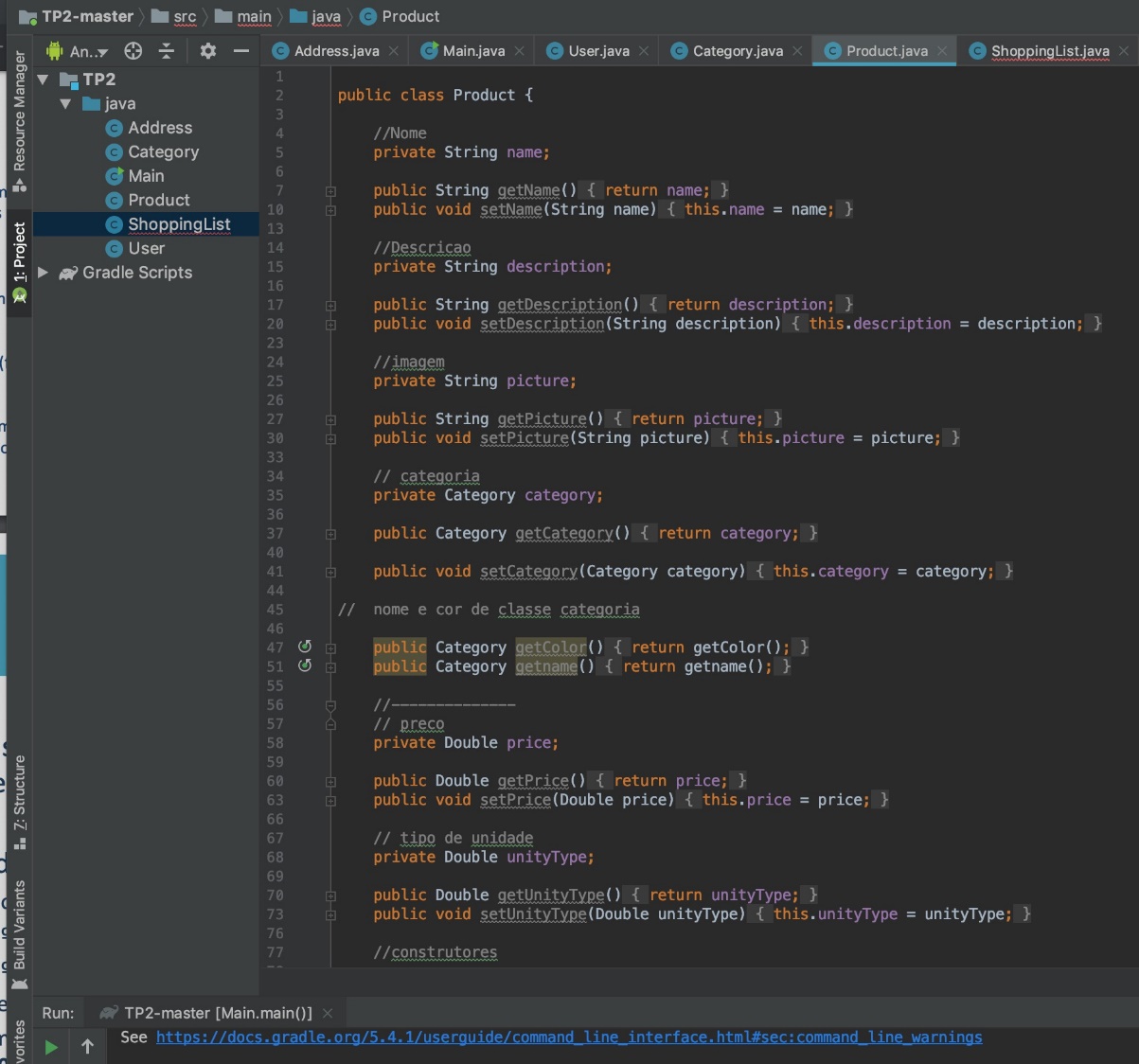
******

# CLASSES – product

***Product*** – Primeiramente foi instanciada a classe em um diagrama UML. Esta classe Product é composta por quatro variáveis do tipo “string”, uma variável “Double” pois como se trata de preços os valores que a variável vai assumir podem ou não ser inteiros.

Estes campos são relativos as produtos que cada categoria poderá ter, como por exemplo “maça, banana, pêra” são tudo produtos da categoria “Fruta”.

O produto tem de ter um nome, uma descrição, e uma foto, uma categoria como foi referido em cima, e o preço .

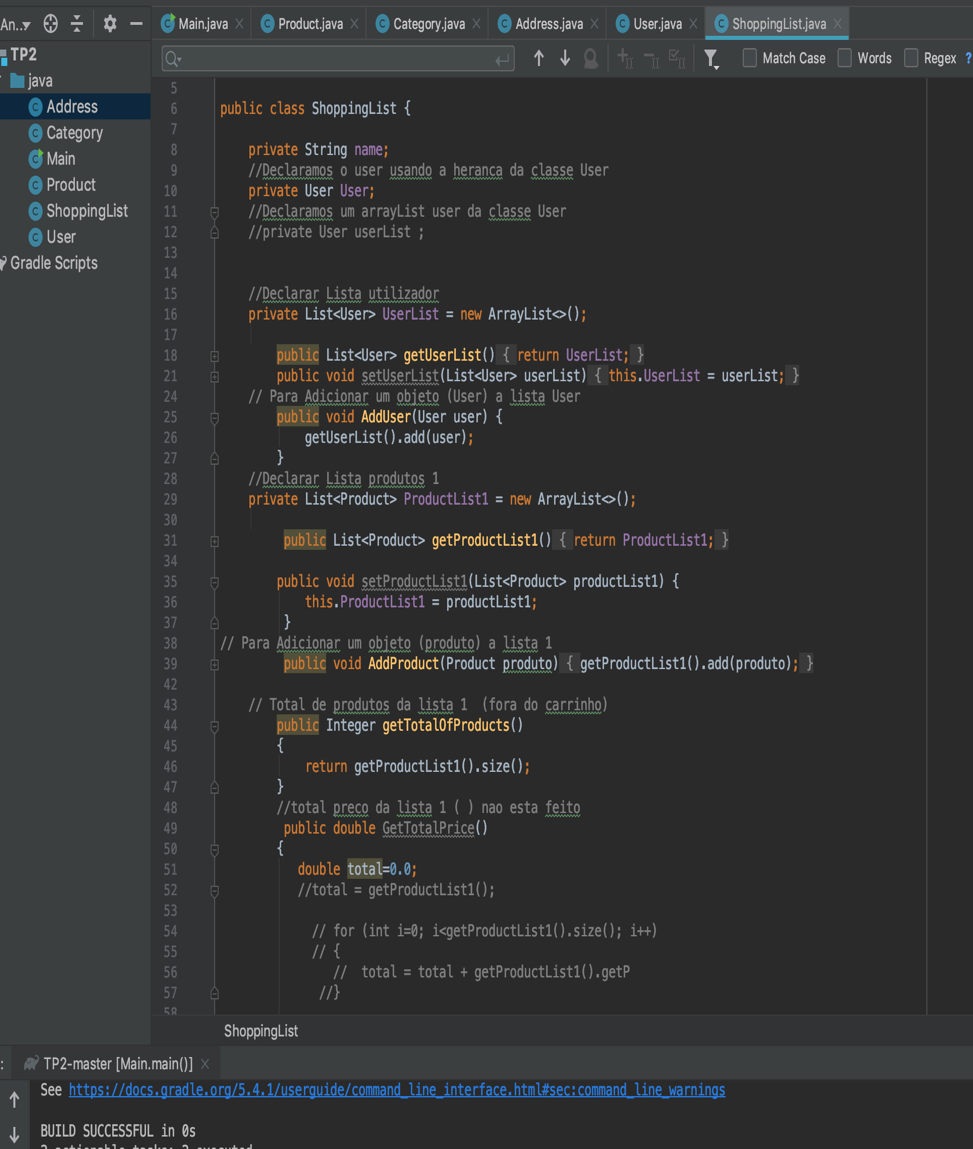
******

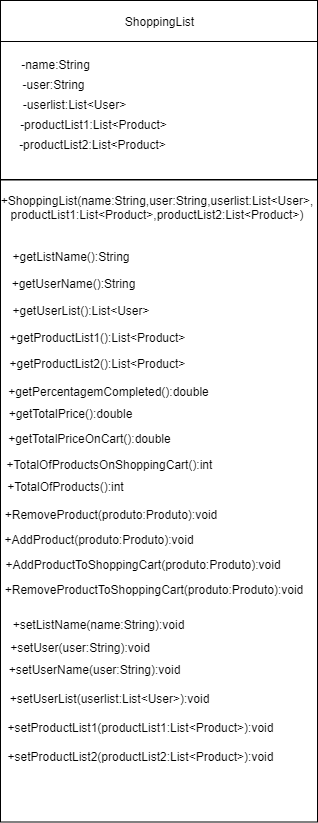
# CLASSES – shopping list

***Shopping List*** – Primeiramente foi instanciada a classe em um diagrama UML. E finalmente chegamos a classe shopping list que acaba por ser o ponto de conclusão deste trabalho no qual usa variáveis “userlist:List<User>” que é a lista de produtos que pertence ao respetivo utilizador que estiver a usar a aplicação. E também as variáveis “productList1:List<Product>” que refere os produtos que vão estar presentes na lista.

Utilizamos o “Add” nas listas, que nos ajuda a que sempre que criemos um objeto novo ele acrescenta a Lista mais um em vez de o substituir.

Criamos um “GetTotalOfProducts” e inserimos o “.size” que nos vai calcular o tamanho da lista .

Calculamos o tamanho das listas, multiplicamos por 100 para nos dar a percentegem e inserimos no “GetPerecentageCompleted” que foi criado para receber tal valor e nos dizer a percentagem total das listas.



# CLASSES – main

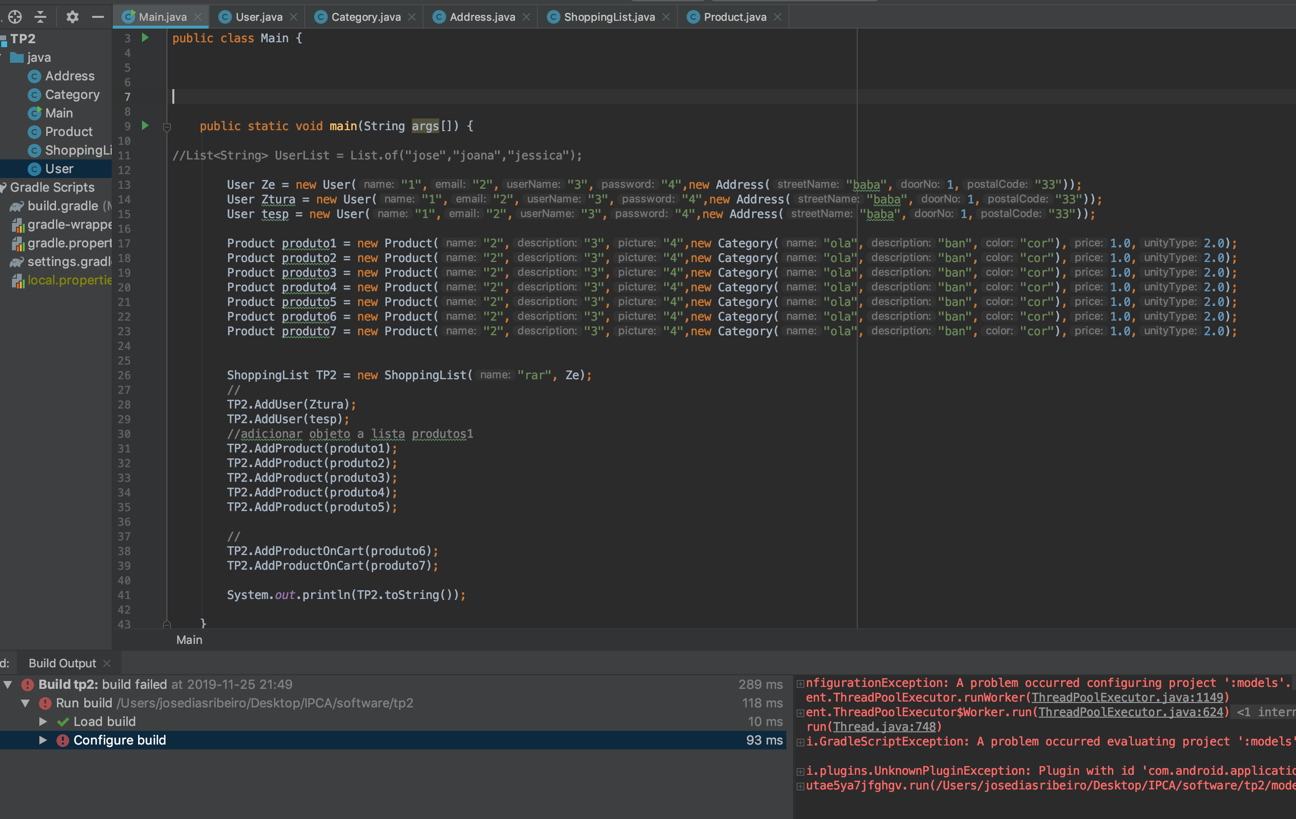
***Main***– No Main começamos por Instanciar as classes para as listas criadas na ShoppingList e inserimos valores.

De Seguida começamos por criar uma instância de ShopingList universal na pagina com o nome de “TP2”, com um utilizador associado crido anteriormente (tudo que criarmos mandamos para o TP2 que manda para o ShoppingList).

Comecamos por adicionar utilizadores com quem vai ser partilhada as Listas de Produtos.

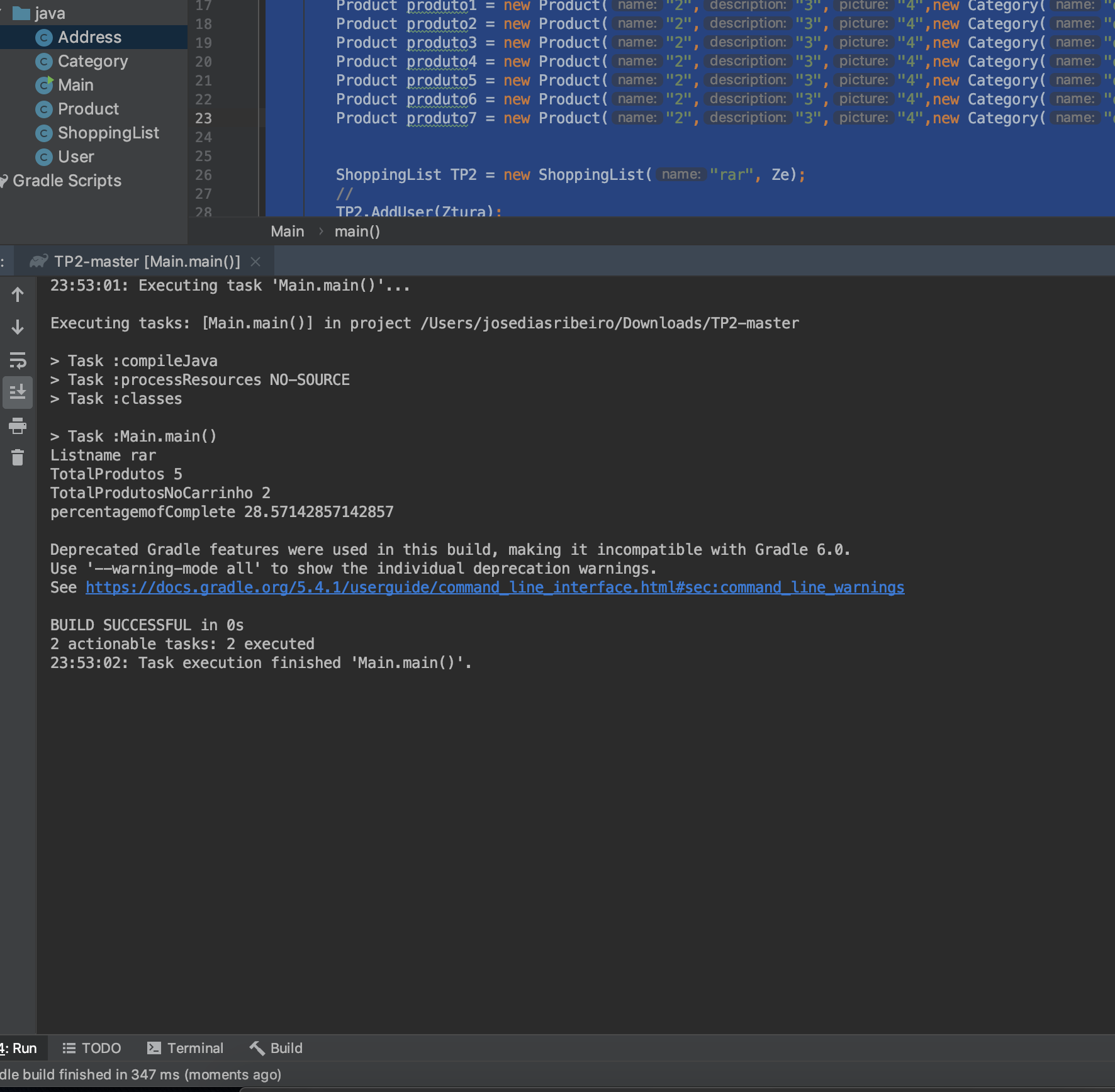
De seguida inserimos as listas de produtos associados ao utilizador “dono” do TP2.

Por fim criamos um “ToString” que vai imprimir o “TP2” com as variáveis que imprimimos em ShoppingList



# POGRAMA EM FUNCIONAMENTO

So para mostrar o código a funcionar

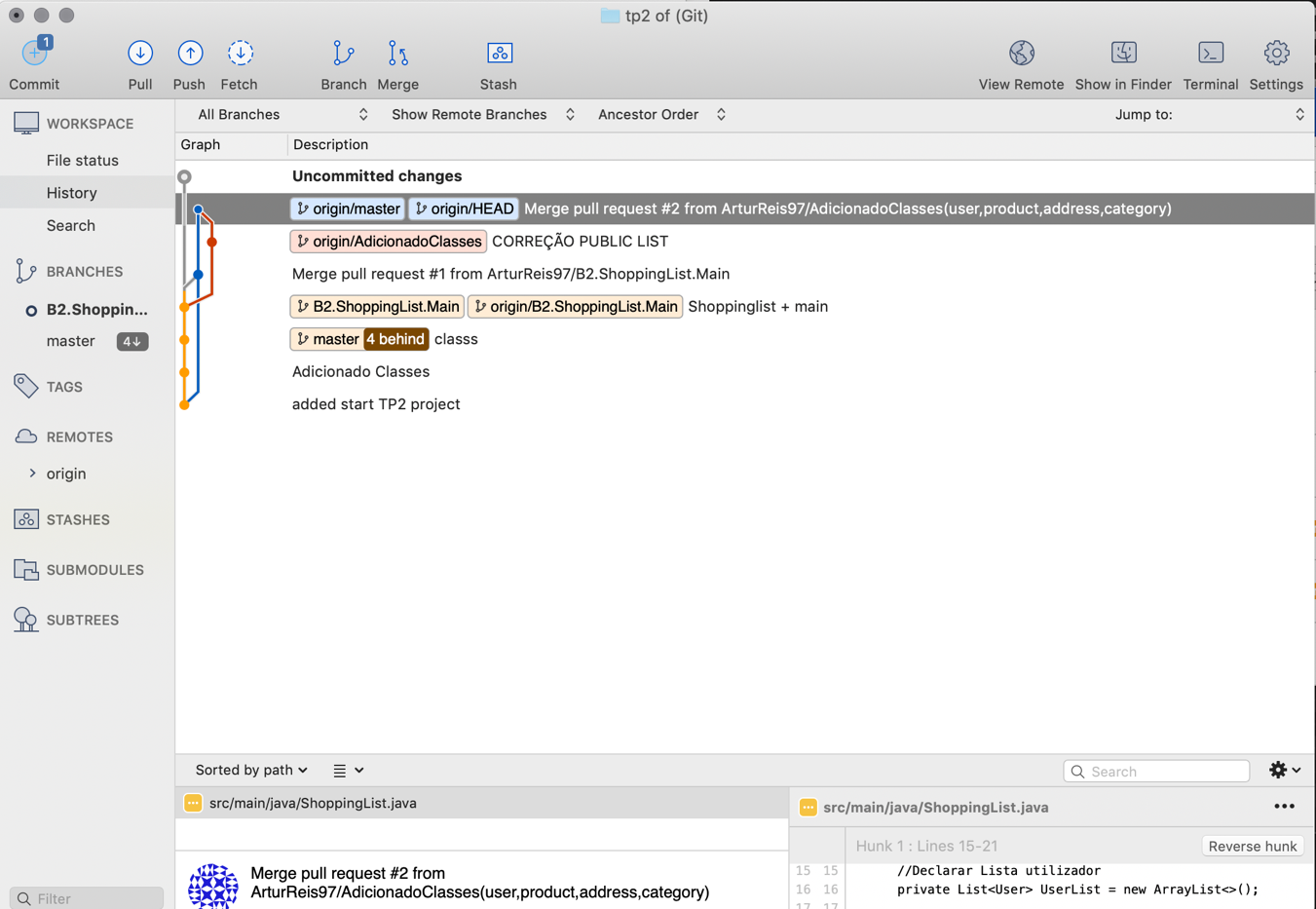


# Conclusão

Em suma este trabalho foi interessante para “apurar” as nossas capacidades no que toca ao java script e a manipulação de vairiáveis, como também no que toca a parte do controle de versões.

Por fim de todo este seguimento do trabalho fazemos um Pull Request para a Master para que todas as partes do trabalho (branches) que foram feitas, se juntem finalmente (como mostra a foto em seguida).

Por conseguinte , não conseguimos acabar todo o que nos é pedido a tempo que nos foi implementado



# Bibliografia

<https://www.devmedia.com.br/conceitos-da-linguagem-java/5341>

<https://tableless.com.br/tudo-que-voce-queria-saber-sobre-git-e-github-mas-tinha-vergonha-de-perguntar/>