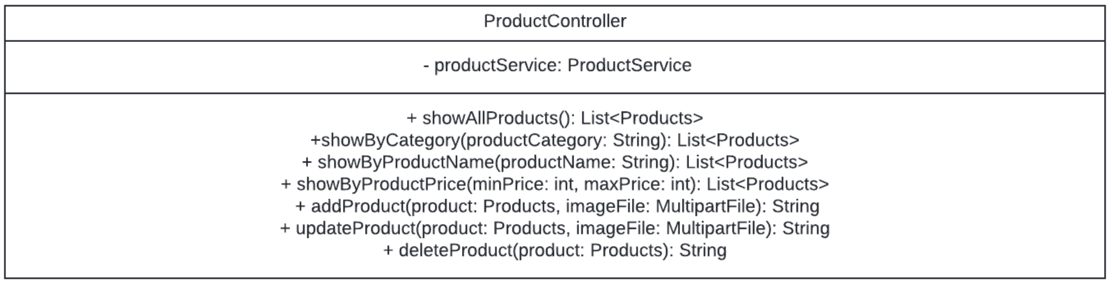

ProductController
- productService: ProductService
+ showAllProducts(): List<Products>
+showByCategory(productCategory: String): List<Products>
+ showByProductName(productName: String): List<Products>
+ showByProductPrice(minPrice: int, maxPrice: int): List<Products>
+ addProduct(product: Products, imageFile: MultipartFile): String
+ updateProduct(product: Products, imageFile: MultipartFile): String
+ deleteProduct(product: Products): String

Invariant: -

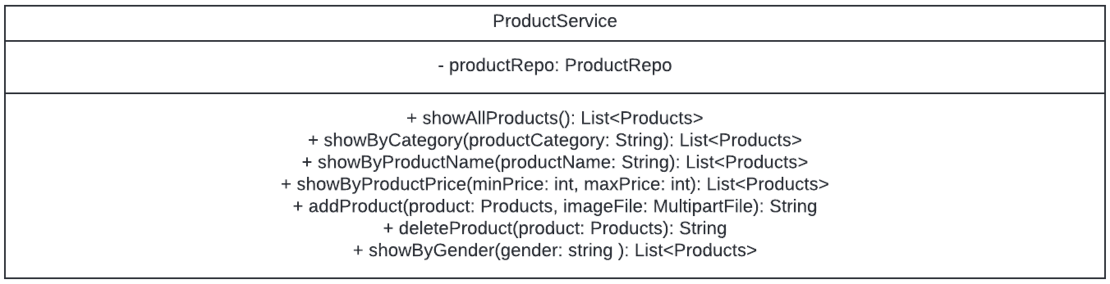
- context ProductController
  inv: self. productService <> null

Pre-Condition: -

- context ProductController:: showByCategory(productCategory: String):
  pre: productCategory->notEmpty() and self.categories->includes(productCategory)
- context ProductController::showByCategory(productCategory: String):
  pre: productCategory->notEmpty()
- context ProductController::addProduct(product: Products, imageFile: MultipartFile):
  pre: product <> null and imageFile <> null and productService.products->excludes(product)
- context ProductController::deleteProduct(product: Products):
  pre: product <> null and productService.products->includes(product)

Post- Condition: -

- context ProductController::showByCategory(productCategory: String):
  post: result = self.products->select(p | p.category = productCategory)
- context ProductController::showByCategory(productCategory: String):
  post: result.body = productService.products->select(p | p.category = productCategory)
- context ProductController::addProduct(product: Products, imageFile: MultipartFile):
  post: productService.products->includes(product)
- context ProductController::deleteProduct(product: Products):
  post: productService.products->excludes(product)


ProductService
- productRepo: ProductRepo
+ showAllProducts(): List<Products>
+ showByCategory(productCategory: String): List<Products>
+ showByProductName(productName: String): List<Products>
+ showByProductPrice(minPrice: int, maxPrice: int): List<Products>
+ addProduct(product: Products, imageFile: MultipartFile): String
+ deleteProduct(product: Products): String
+ showByGender(gender: string): List<Products>

Invariant: -

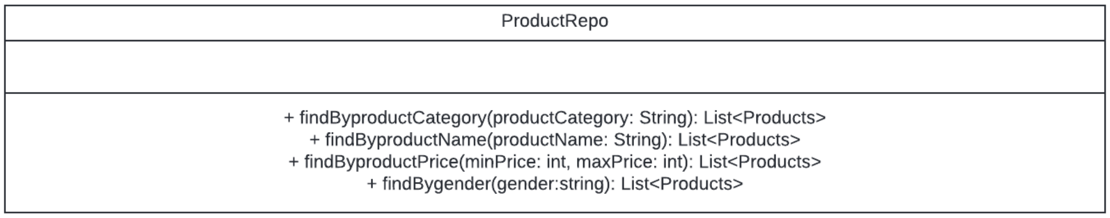
- context ProductService
  inv: self.productRepo <> null

Pre-Condition: -

- context ProductService::showByCategory(productCategory: String) :
  pre: productCategory->notEmpty()
- context ProductService::showByProductName(productName: String) :
  pre: productName->notEmpty()
- context ProductService::showByProductPrice(minPrice: int, maxPrice: int) :
  pre: minPrice >= 0 and maxPrice >= minPrice
- context ProductService::addProduct(product: Products, imageFile: MultipartFile)
  pre: product <> null and imageFile <> null and productRepo.products->excludes(product)
- context ProductService::deleteProduct(product: Products)
  pre: product <> null and productRepo.products->includes(product)

Post- Condition: -

- context ProductService::showAllProducts() :
  post: result = productRepo.products->asSequence()
- context ProductService::showByCategory(productCategory: String) :
  post: result = productRepo.products->select(p | p.category = productCategory)
- context ProductService::showByProductName(productName: String) :
  post: result = productRepo.products->select(p | p.name = productName)
- context ProductService::showByProductPrice(minPrice: int, maxPrice: int) :
  post: result = productRepo.products->select(p | p.price >= minPrice and p.price <= maxPrice)
- context ProductService::addProduct(product: Products, imageFile: MultipartFile)
  post: productRepo.products->includes(product)
- context ProductService::deleteProduct(product: Products)
  post: productRepo.products->excludes(product)

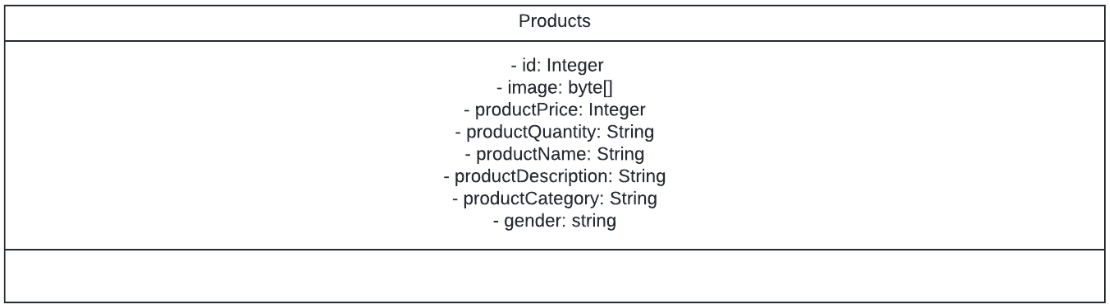

ProductRepo
+ findByproductCategory(productCategory: String): List<Products>
+ findByproductName(productName: String): List<Products>
+ findByproductPrice(minPrice: int, maxPrice: int): List<Products>
+ findBygender(gender:string): List<Products>

Pre-Condition: -

- context ProductRepo::findByProductCategory(productCategory: String):
  pre: productCategory->notEmpty()
- context ProductRepo::findByProductName(productName: String):
  pre: productName->notEmpty()
- context ProductRepo::findByProductPrice(minPrice: int, maxPrice: int):
  pre: minPrice >= 0 and maxPrice >= minPrice

Post- Condition: -

- context ProductRepo::findByProductCategory(productCategory: String):
  post: result->forAll(p | p.category = productCategory)
- context ProductRepo::findByProductName(productName: String):
  post: result->forAll(p | p.name = productName)
- context ProductRepo::findByProductPrice(minPrice: int, maxPrice: int):
  post: result->forAll(p | p.price >= minPrice and p.price <= maxPrice)



Invariant: -

-
  ```
  context Product
  inv:      self.id <> null and
            self.image <> null and
            self.productPrice <> null and
            self.productQuantity <> null and
            self.productName <> null and
            self.productDescription <> null and
            self.productCategory <> null
            self. gender <> null
  ```