“Lemon Storage” Documentation

Team name: “Lemon”

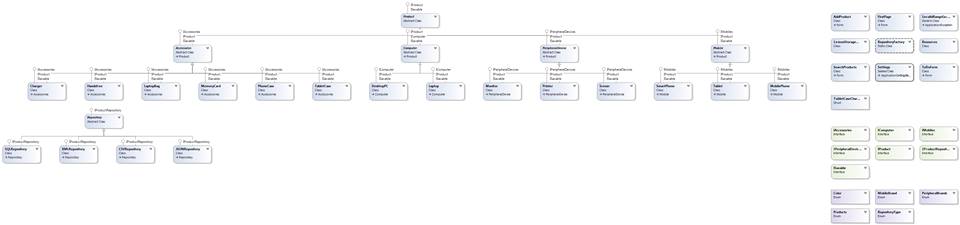
Members:

* Kristina Yoveva(KYoveva) – GitHub: KYoveva
* Ivan Vasilev(ivanvasilev\_90) - GitHub: ivanvasilev
* Plamen Kostadinov(plamenti) - GitHub: plamenti
* Plamen Paunov(Puncky) - GitHub: Puncky
* Georgi Kozhuharov(apch2) - GitHub: apch
* Vladi Vladev(vladi.p.vladev) - GitHub: Vladeff
* Rosen Balkandzhiev(Rosen.balkandzhiew) - GitHub: RBalkandjiew
* Emiliya Georgieva(Emiliya93) - GitHub: Emiliya93
* Lyubomir Durankev(lubzey) - GitHub: lubzey

Project purpose:

Our team project is a program, which saves and loads different kinds of products in a storage. It helps the people that work in the storage to track the inventory in the storage. The users can add and remove different kinds of products using the program. When some of the products is out of stock the program alerts for insufficient quantity and helps to make business decisions.

Class diagramm:

**GitHub URL**: <https://github.com/LemonStorage>

# **General** Requirements:

Interfaces:

* IAccessories
* IComputer
* IMobiles
* IPeripheralDevices
* IProduct
* IproductRepository
* ISavable

Ordinary Classes:

* Laptop
* DesktopPC
* Monitor
* Scanner
* Printer
* MobilePhone
* SmartPhone
* Tablet
* Charger
* Handsfree
* LaptopBag
* MemoryCard
* PhoneCase
* TabletCase
* HeadPhone
* CSVRepository
* JSONRepository
* Repository
* RepositoryFactory
* SQLRepository
* XMLRepository

Abstract classes:

* Product
* Accessories
* Computer
* Mobile
* PeripheralDevice

Custom exception class:

* InvalidRangeException

Levels of depth in inheritance:

IProduct -> Product -> Computer -> Laptop

Polymorphism usage:

We use polymorphism for creating instances for the different Mobiles.

Structure:

* TabletCaseCharacterustics

Enumerations:

* RepositoryType
* Products
* MobileBrand
* Color
* PeripheralBrands

Events:

We use event handling in the graphic user interface when we click on different buttons.

Design patterns:

* Repository
* Factory

# **Optional** Requirements:

Static members:

* public static class RepositoryFactory
* public static IProductRepository GetRepository(string repositoryType)
* public static SqlConnection CreateConnection()

Constants:

We use constants in almost every class to validate the data.

Namespaces:

We have different namespaces for the different types of products.

User interface (UI):

Windows Forms with connection to SQL Database.