Kristina Myftaraga

200069406

Prof. Vladimir Georgiev

March 5, 2020

Fundamental Data Structures

Project Specification

**Project Description**

A web-like program where users can choose to purchase an Editor depending on their needs. It can be PhotoEditor or a VideoEditor. PhotoEditors are classified as Simple and Complex and VideoEditors are separated into VirtualReality and Social Media. This classification is done in accordance of the purpose of usage of each Editor.

Each User has their own menu where they can perform different operations like register, login, add or purchase editors depending on their status. Once the User enters our web-like program, in the OnlineStore Menu he/she can Log in/ Register or just View Editor and Companies. After this, the User Menu contains search options for Companies and Editors and Buy or Add Editor option. An example of one operation would be: for the User to Buy one Editor, first the list of Editors is shown and then he can choose one. This bought editor is going to be added to his Collection. For the User to Add one Editor, there will be some conditions. If User is part of some Company, he can proceed.

Company\*currentCompany= “X”;//he can add one Editor

If user.company==null; //he can only buy one editor. He has no rights to add.

**Classes Description and Hierarchy**

* Class *Editor* – An abstract class that has as data slots a name, operating system, format, getTotalPrice, Version, Format, ID
* Class *PhotoEditor –* A class which inherits from class Editor and owns the data slots layers, original saved and color mode.
* Class *VideoEditor*- A class inherited form Editor and owns the data slots AudioMaster, Montage, Overlay.
* Class *Simple* – A class inherited from PhotoEditor class. Data slots: name, version, release date, exposure, noise reduction, resizing and cropping etc.
* Class *Complex* – A class inherited from PhotoEditor class. Data slots: portrait corrections, photo masking, photo stitching.
* Class *VirtualReality* – A class inherited form VideoEditor class. Data slots: platforms (VR set), photography (digital environment), mobility (immerse world that you can walk in), timeline (video can progress through a series of events or experiences), story (filmmaker does not control physical location of the viewer)
* Class *SocialMedia* – A class inherited form VideoEditor class. Data slots: operating systems (apps, platforms), templates, animation tools, integrated social media platforms,
* Class *User* – data slots: Name, Address, Phone, Budget, ID and class Editor pointer.
* Class *Company* – data slots: Name, Phone Contact, Address, Access Code.

The object will be collected in the following vectors:

vector < Company\* > Companies; Initialized from a file

vector < Company\* > Promoted\_Companies; Initialized from a file

vector < Editor\* > Editors; Registered Editors

vector < User\* > Users; Registered Users

(more can be added during the development process of the project)

I will implement MERGE-SORT **Algorithm** to sort the collections **alphabetically.**

**When selecting their menus, users are prompted to LOG IN or REGISTER.**

* *A user is registered when an instance of the type-of-user class object is created.*

*Example: When a User is registered, one new object will be created and inserted in the vector of User.*

* *After Registered, to log in, a user is prompted to write his/her ID. If an object with the data slots -ID the same with the entered ID- exists in the specified vector, the user will have access to his account (interface) and can perform its own operations mentioned in the beginning.*

**POLYMORPHIC FUNCTIONS:**

**virtual void Description()** function that shows information about the candidates, firstly implemented in the class Editor, then overridden in the derived classes (PhotoEditor, Simple, Complex, VideoEditor, VirtualReality, SocialMedia).

**virtual double getTotalPrice()** function first declared in class Editor, then overridden in derived classes with the different prices.

**virtual string getName()** function first declared in the class Editor and then overridden in derived classes with the different Names.

Hierarchy: