CYPRUS INTERNATIONAL UNIVERSITY
FACULTY of ENGINEERING Fall 2019-2020
CMPE214,ISYE214,VCP251,MISY214,ITEC214
PROJECT – XML/JSON Backend Visual C# LINQ Application

Given Date: 21.04.2019 Wednesday
Submission Deadline: 25.05.2019 Monday

Instructor: Asst. Prof. Dr. Cain KAZIMOGLU

Important Notes:

o This project is given to you on the indicated date above and should be submitted by 23:55 on the deadline date via Moodle system.

o This a team project and each team member must have unique responsibilities. You must be the sole owner of your project as a team. The code you write ***must be your own work*** and **should not include any plagiarism taken from the Internet**. You also should not use someone else's work as your own work in this project. In other words, this project must be your own unique work.

o ***While the work is given as a team work, each team will or will not be evaluated as a whole. The more coordinated you work, the higher chance you secure a good mark.***

o ***You are responsible to find a team and work with others in this project. Please do not complain to the lecturer as "I do not know anyone", "I do not know any team", "I cannot find a team" and/or "Can I do the work individually?". The work is given to you as team work and no, you cannot do it alone. Software Engineering is a team work, and every single software we use today is developed by a team – not individually. Thus, learning how to be effective within a team and being able to function together with your colleagues is a big part of the learning process in this course. DO NOT SEND MESSAGES/EMAILS and/or ASK QUESTION ABOUT THIS SUBJECT PLEASE AS I WILL SIMPLY IGNORE THEM. You need to learn with others.***

Project Aim and objectives:

The project aim and objectives are listed below.

Aim:

The project aims to practice the Language-Integrated Query (LINQ) technology of Visual C#. Through using LINQ, any external data source such as an XML file or a JSON document can be used in C# in a very structured way. Using LINQ, you will need to either construct JavaScript Object Notation (JSON) or an eXtensible Markup Language (XML) to build up a document to be used as a back end for your project. Having done this, you will build a Graphical User Interface (GUI) within a selected theme to build up a program that will capture the workflow and behavior of a realistic visual software. You need to make sure to create the XML/JSON file well-structured and the file should include all the necessary fields necessary for the theme you selected. The XML/JSON file will be consisted of a series of records by default and through the Visual Program you develop, the records could be selected, updated, deleted or a new records could be inserted. In other words, you will use LINQ to provide access to an XML/JSON file through using a GIU developed in Visual C# Windows form application. In short, XML/JSON will be used as your main data storage.

Objectives (Read this section at least 5 times and highlight important points before you start working on the project.):

* You will need to design a series of different forms – each one having a unique job and all connected and functioning together. All of these forms will communicate with your XML/JSON file for data persistency.

The first form is the ***products form*** which will list all items in the inventory alongside with the quantity, price and other fields you will identify according to your selected theme. The themes are listed in the Project Topics section. There should be at least 10 items in this form – each one having a picture. The picture name of items must match with their ID. The unregistered users and all other users will be able to see the item name, quantity and price. Using this form, unregistered users and others will be able to see all available items in the inventory. What these items can be will change according to your theme.

From the products form, the unregistered user can go to a **sign in form** where they can register themselves to the system. Here if a user does not have a username and/or password, they will be able to register with their email address and password. You can also ask name and surname of each user.

If a user is already registered, they can use **login form** to login. This is for users who have done registration already. They can use username and password to login to the system.

Once a user login, they can buy items from the **purchase form**. This form will allow users to purchase an item or service (whatever you list). For example, if your theme is flower shop the user can see a bouquet of roses on the products rose with 10 in quantity with each bouquet being 6€. Should a registered user successfully buy a bouquet of roses, this will be subtracted from the overall stock. Each user should be able to buy one item at a time. This means that you do not have to develop a basket system.

There also should be an **admin panel** in the application where the admin will be able to connect by entering "admin" username and "admin" password to **login** form. Please do not create a separate login form as the admin will use the same login as every other user. If you design a separate login for admin panel, this will be penalized.

Admin is the person that can increase the number of products in the stock, add new products or permanently delete them. You can do these functionalities separately or all together so at this point it is up to you how to support this feature. It is suggested that you create different forms for add (**addItems**) and update (**updateItems**) and keep the delete functionality on the **listRetailProducts** form which the admin will access. You might want to research and find out how admin panels are design before forming your UI and completing the back-end programming. Remember that every record you have on the list must be matched with a picture in a folder – as this is not optional. You can take admin username and password from a JSON/XML or text file.

* You will need to use at least 6 visual form components in your project throughout the forms. These are textboxes, radioButtons, checkListBox, comboBox, Listbox and errorProvider. All errors must be handled with either errorProvider or with a messageBox customized to display errors. You can of course use more components if you want to but make sure that all fields are validated accordingly.

* All kind of validation has to be done and all exceptions must be handled in the program. This has to be done in LINQ side as well as in the User Interface

* The look and feel of the program must be professional, which means you have to design a unique looking user interface through image buttons, background images. The application must be user-friendly.

***** The program has to be bug free**.**

Project Teams:

This project is designed as a team work. This means that you have to work in a team and each member must have different responsibilities. You will need to keep yourself up to date within the team so that you could efficiently work together. Each team must be consisted of 3

members. However, under certain circumstances, if the team accepts to do extra work – it is possible to form a team with 4 members. No team can be less than 3 – unless there is a good reason behind this. Hence, ideally you need to form your team with 3 members and that each one needs to have unique responsibilities. Remember that members will be evaluated individually based on the work they have done in the project. *"we worked together and we have done everything together"* will eventually lead your project to sink in, so please assign you and your team members unique responsibilities.

Project Topics:

The topics are listed below. If a team picks up a theme, the other teams cannot pick up the same topic. Hence, please select your topic wisely. Make sure to form your team ASAP and discuss at least 3 different topics among you because if first topic you agreed is already picked up, you could easily pick the second or the third choice.

1. Flight Reservation
2. Hotel Reservation
3. Video Game Store
4. Smartphone shop
5. Pharmacy
6. Cinema Theatre
7. Library
8. Gym/ Sports Center
9. Men & Women ware
10. Rent a Car
11. Pet shop
12. Florist
13. Coffee Shop
14. Fast Food Order
15. Computer Shop
16. Patisserie
17. Table-top Games Shop
18. Foot Wear
19. Toy Store
20. Tourism Agency
21. Gift Shop
22. Jewelry Store

Extra work:

While not compulsory, you can increase your chance of getting a high grade by going above and beyond. Here are some areas that are not compulsory but can be implemented in your project easily:

-USE Both XML and JSON in one project so data is saved to both file formats.
- Create fully functional installation package with short cuts and installation wizard.

Project Grading:

JSON/XML File management **%20**
GUI Professionalism **%20**
Validation **%20**
Project works as intended **%20**
Team coordination and work division **%20**
Above and Beyond (Extra work) **%20**