

**TECHNOLOGICAL UNIVERSITY DUBLIN**

**Student management system**

by

Dahir Mussa

Bacheior of Science in Computing

Word count: 1614

in the

School of Computing

Department of Informatics and Engineering

March 2020

# Declaration of Authorship

I, AUTHOR NAME, declare that this thesis titled, ‘THESIS TITLE’ and the work presented in it are my own. I confirm that:

This work was done wholly or mainly while in candidature for a research degree at this University.

Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.

Where I have consulted the published work of others, this is always clearly attributed.

Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.

I have acknowledged all main sources of help.

Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

**Contents**

[**Declaration of Authorship** **i**](#_Toc4244)

[**Abstract** **ii**](#_Toc4245)

[**List of Figures** **iv**](#_Toc4246)

[**List of Tables** **v**](#_Toc4247)

[**Abbreviations** **vi**](#_Toc4248)

[**1 Application Overview** **1**](#_Toc4249)

[**2 Functional Requirements** **2**](#_Toc4250)

[**3 User Requirements/User Interface** **3**](#_Toc4251)

[**4 Design Pattern Explanation** **4**](#_Toc4252)

[4.1 Abstract Factory design patterns 4](#_Toc4253)

[4.2 Builder design pattern 4](#_Toc4254)

[4.3 Prototype Patterns 5](#_Toc4255)

[**5 Design-code Explanation** **6**](#_Toc4256)

[5.1 Abstract Factory design patterns 6](#_Toc4257)

[5.2 Builder design pattern 6](#_Toc4258)

[5.3 Prototype Patterns 6](#_Toc4259)

**Chapter 1**

# Application Overview

this application is student management system it helps student as well as the school or colleges, A School system generates and uses a large amount of data. This data must be communicated appropriately to students and parents, the student system is design to help college for management of student. Extensive information is available at your fingertips through the system, viewing student data, viewing student course.

In the software we can register as a user and user has two types, student and administrator has the power to add mew user and can edit and delete a user, a student can register as user and can add edit , delete his profile, the administrator can add , edit and delete marks for the student . all the users can see the marks.

**Chapter 2**

# Functional Requirements

### 

**Login/Register System**

* The system should allow a regular user to create an account using a personalised username, password and their own full name and age and so on
* When the account is registered, the system should allow the user to log back into this profile using just the username and password.

**add new user/edit user:**

* the system allows the user or the teacher to add a new student to the system. User can edit username

**remove the user:**

* it should also allow the user to remove a student from the system

**view marks/ add /edit Marks Details:**

* the system should the student to view their marks and it should also allow the user to add and edit marks

**Edit student Details/ delete:**

* the system should allow the user to edit their personal details and it should allow the user to delete their personal details.

**add new Subjects/edits/delete**

* the system should allow the user to add a new subjects and edit subject and delete subjects.

**Chapter 3**

# User Requirements

## 3.1 User login

The feature used by the user to login into the system, they are required to enter username and password before they are allowed to enter the system. The user username and password will be verified and if name the user is allowed to not enter the system.

* Username and password is provided when they register
* The system must only allow user with valid username and password to enter the system
* The system performs authorization process which decides what the user level can access or not.
* The user must be able to logout after they finished using the system if it’s possible

## 3.2 Register new user

The feature can be performed by all users to register new user to create account

* System must be able to verify information
* System must be able to delete information if information is wrong

# User Interface

**Login:**

* the user can login in into the system just using the username and password

**Add new user:**

* the user can add a new user using their username and password

**Edit user type:**

* the user can edit their username then updated it

**Delete user:**

* the user can delete their account

**Registration:**

* the user can register as a student using their personal details

**Edit Details/delete details:**

* the user can edit their details or delete their personal details

**Add/Edit/Mark Details/view marks:**

* the user can add new marks and they can edit the marks, they can view their marks

**view user details:**

* the user can view their details

**add new subject/edit/delete:** user can add new subject and they can edit and delete a subject

**Chapter 4**

# Design Pattern Explanation

## 4.1 Abstract Factory design patterns

the Abstract Factory pattern is used to provide a client with a set of related or dependent objects. the family of objects created by the factory are determined and also it provide an interface for creating families of dependent objects without specifying their concrete classes one. if you contrast this to the factory method pattern the single different between the abstract factory pattern and the factory method is that the factory pattern constructs a single object and the abstract factory pattern construct multiple objects.

in the abstract Factory pattern an interface responsible for creating a factory of related objects without explicitly specifying their classes and each generated factory can give the objects as per the Factory pattern.

abstract factory pattern is one level higher then the factory patterns, you could use this pattern to return one of several related classes of objects and each of which can return several different objects on request, the abstract factory is a factory object that returns one of several groups of classes.

The design pattern , Abstract Factory Pattern, this type of design pattern comes under creational pattern as this pattern provides one of the best ways to create an object.

## 4.2 Builder design pattern

The builder pattern is used to create complex objects with constituent parts that must be created in the same order or using a specific algorithm. An external class controls theconstruction algorithm and this design pattern is designed to provide a flexible solution to various object creation problems in object-oriented programming.

The Builder pattern also solves some of the problems with Factory and Abstract Factory design patterns when the object contains a lot of attributes.

The intent of the Builder design pattern is to separate the construction of a complex object from its representation. It is one of the Gang of Four design patterns.

## 4.3 Prototype Patterns

The prototype patterns is a creational design pattern in a software development its used when the type of objects to create a determined by a prototypical instance which is cloned to produce new objects, this practise is particularly useful when the construction of new object is inefficient.

Prototypes are usually the first draft of a product, the prototype pattern is slightly different. The prototype pattern involve copying something that already exists. an example of this in the real world could be splitting a cell where two identical cells are created.

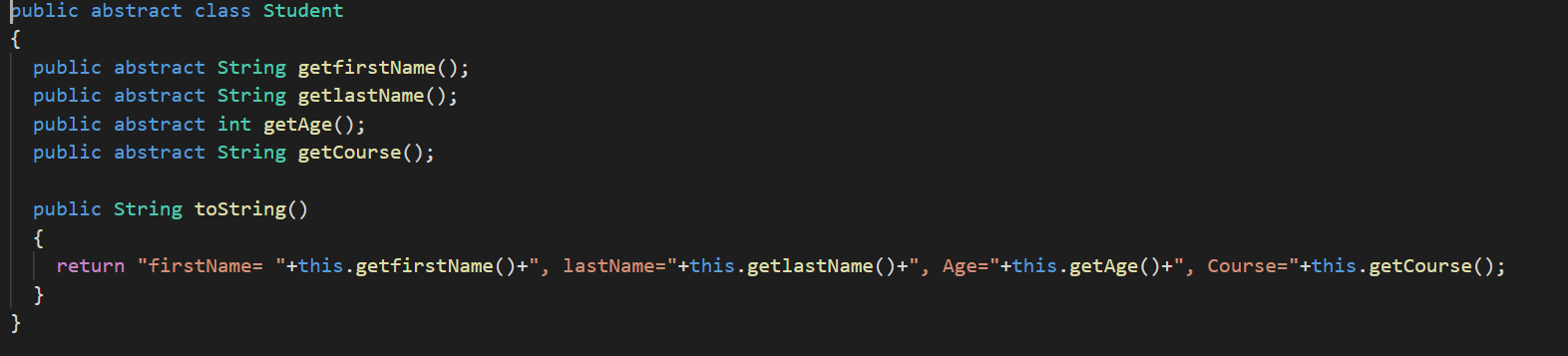
5

**Chapter 5**

# Design-code Explanation

## 5.1 Abstract Factory design patterns

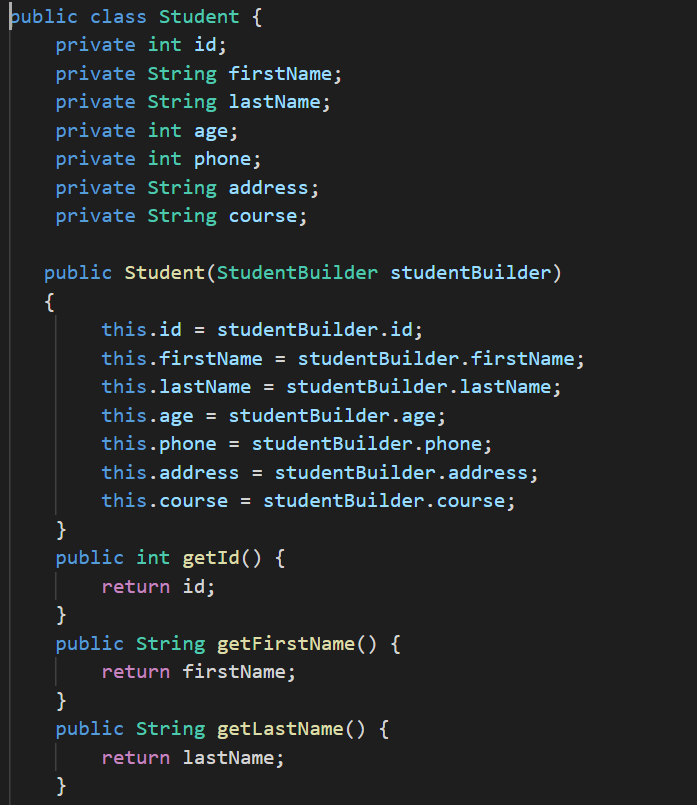
The first step of implementing the abstract factory pattern is to build an hierarchy of products for it to build and the abstract super class is called Student, since the Student is an abstract class then we have to extend from it to create the concrete classes that abstract factory can build This step is to create the abstract Factory , this will be an abstract super class factory that will form a the basis to other factory classes.



## 

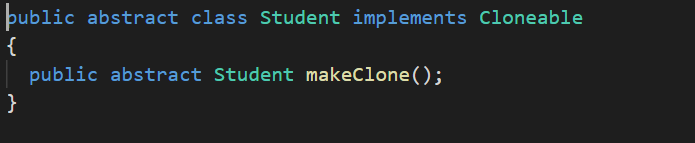
## Builder design pattern

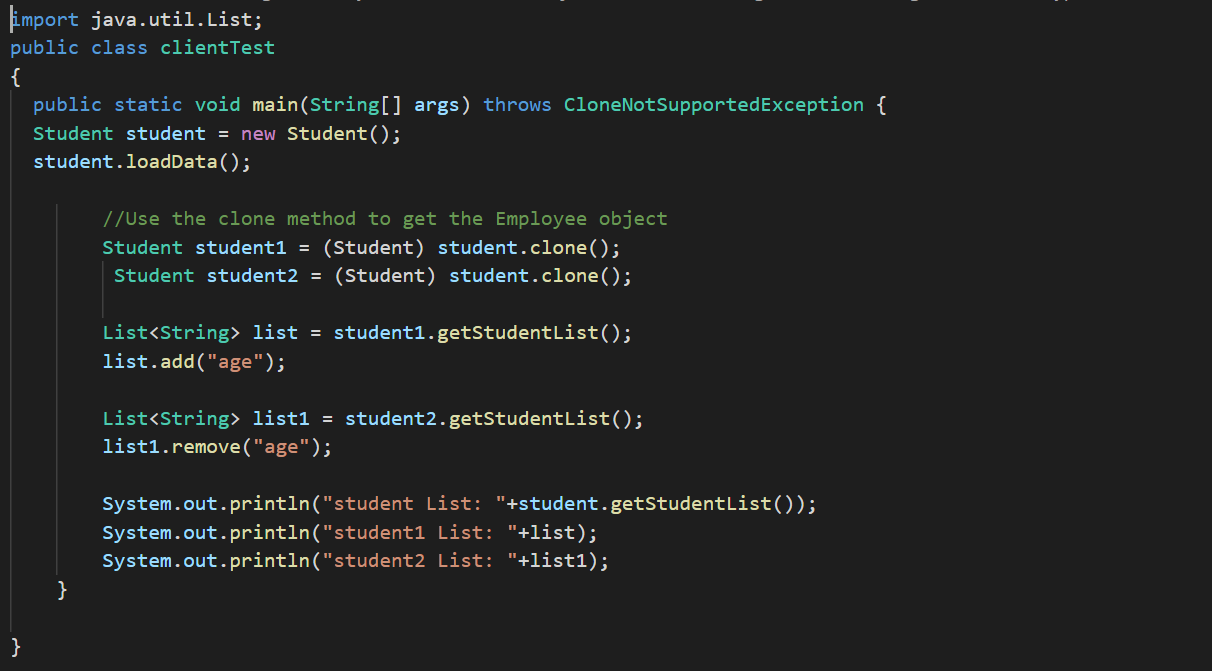
lets consider a Person class that holds the following information firstname lastname age course in some situations we many want to instantiate a person object with full set of data items and in other situations we many want to create an instance of person with just a firstname and lastname things are easy if there are only 4 fields but adding more fields make it diffcult to keep track of the order and number of the minimum required parameters.



## 5.3 Prototype Patterns

The first step of building the example below is to create an abstract class The cloning is process of creating an exact copy of an existing object in the memory. The second step is to extend from the student class to create a cloneable class. course class contains a person object as one of its data members.





**Chapter 6**

# Bibliography

2020.[online] Available at: <https://www.academia.edu/35228990/STUDENT\_MANAGEMENT\_SYSTEM\_PROJECT\_REPORT> [Accessed 28 March 2020].

2020. [online] Available at: <https://www.youtube.com/watch?v=eD0Qj0qdiFc&t=386s> [Accessed 28 March 2020].

Moodle - TU Dublin Blanchardstown Campus. 2020. *Moodle - TU Dublin Blanchardstown Campus: Log In To The Site*. [online] Available at: <https://moodle.itb.ie/pluginfile.php/253812/mod\_resource/content/1/OODP\_L2\_AbstractFactory.pdf> [Accessed 28 March 2020].

Moodle - TU Dublin Blanchardstown Campus. 2020. *Moodle - TU Dublin Blanchardstown Campus: Log In To The Site*. [online] Available at: <https://moodle.itb.ie/pluginfile.php/258090/mod\_resource/content/1/OODP\_L4\_Builder%20Pattern.pdf> [Accessed 28 March 2020].

Moodle - TU Dublin Blanchardstown Campus. 2020. *Moodle - TU Dublin Blanchardstown Campus: Log In To The Site*. [online] Available at: <https://moodle.itb.ie/pluginfile.php/260115/mod\_resource/content/1/OODP\_L5\_Prototype%20Pattern.pdf> [Accessed 28 March 2020].

JournalDev. 2020. *Abstract Factory Design Pattern In Java - Journaldev*. [online] Available at: <https://www.journaldev.com/1418/abstract-factory-design-pattern-in-java> [Accessed 28 March 2020].

JournalDev. 2020. *Builder Design Pattern In Java - Journaldev*. [online] Available at: <https://www.journaldev.com/1425/builder-design-pattern-in-java> [Accessed 28 March 2020].

JournalDev. 2020. *Prototype Design Pattern In Java - Journaldev*. [online] Available at: <https://www.journaldev.com/1440/prototype-design-pattern-in-java> [Accessed 28 March 2020].