***Implementation for the Pocket Closet App***

***Spring 2020, Phase 3***

***Prepared by:***

***For Fairfield University:***

***Logan Pensa***

***Adrian Gallant***

***Aaron Banson***

***Hemant Maheshwari***

***Daniel Wilson***

**Table of Contents**

**Implementation Languages**………………………………………………………………..………………………..3

**Coding Conventions for Pocket Closet**…………………………………………………...……….4

**Comments**………………………………………………..………………………………………..5

**Variable Names**………………………………….………………………………………..6

**Method Arguments**……………………………………………………………………….7

**Parenthesis, Braces and Indentation**…………………………………...………………..8

**Control Structures**…………………………..…………………………………………….9

***Implementation Languages***

The Pocket Closet team choose to use both C# and mySql for the development of the mobile application. C# is an object-oriented programming language that is an excellent language for implementing screen handling procedures. Bearing in mind that this app is a mobile application, the team decided to use Xamarin as the main IDE for the development of the app. MySQL is a relational database management system based on SQL. MySQL serves purposes like data warehousing, e-commerce, and logging applications. Though, the most common use of mySQL is for creating web databases and it is also possible to create websites that interact with mySQL servers when the language is used in tandem with other languages like PHP or Perl. For the database, the team decided to use MySQL Workbench as the main IDE. MySQL Workbench was used to develop the database and the necessary store procedures.

The choice was made to choose C# using Xamarin and mySQL using MySQL Workbench was due to a few reasons. C# is a language that is accessible to everyone for use, as there are a few community programs that allow an IDE to be downloaded for free and to use. C# is a language that allows for excellent implementation of screen handling which is a necessity for an app like Pocket Closet. Xamarin was chosen due to the fact that this platform could use C# as a programming language (given the purposes it can serve), but this IDE can specifically allow for using C# to develop mobile applications. So bearing all these details in mind, this was why C# and Xamarin were chosen for the development of the mobile application and the Web API. The reason mySQL with MySQL workbench was chosen, was because these were both free. Some of the purposes of mySQL included developing a relational database to be set up on a server. Since there would be many potential users for this app, a server would certainly need to be set up for the app in order to manage that many users. Bearing all this in mind, the integration of the mobile portion, the Web API and the Database was successful with these two IDEs.

***Coding Conventions for Pocket Closet***

*Purpose*

Gives a description of the coding standards that are used in the Pocket Closet application.

*File Names*

SQL files end with the extension .sql

MySQL Workbench specific files end with the extension.mwb

Xamarin files files end with the extension .cs

Examples include: Pocket\_Closetbk.sql, Pocket\_Closet.mwb, AccountPage.xaml.cs

***Comments***

Commenting in C# breaks down as follows. (//) are used for single line comments, (/\*) for multi line comments and (///) XML tags displayed in a code comment. Examples for all of these are shown below. The comments within the class bodies of the file usually outline the main function of particular methods. Comments are also used when describing the purpose of important variables. The comments are placed near the method or variable that usually appears after the comment.

Examples of 2 different ways to comment in C#

//On click of the button, the login screen will be open

<!-- This is how a comment would look in a XAML file-->

Comments in mySQL are usually commented as follows. # comment and commenting with --, both span 1 line and must be at the end of the line (not preceding the words/characters used). Commenting with /\*\*/ will span multiple lines and the words/characters used go in between. Examples of all these comments are shown below.

Examples of all three ways to comment in mySQL (not all are used)

# This store procedure will create a post table with all the necessary attributes

--This store procedure will retrieve all outfits

/\* This store procedure relies on a primary key to delete all entries,

and this does not need any additional attributes for it to work

\*/

***Variable Names***

Names for both C# and mySQL all follow their respective naming conventions.

Throughout this entire process, the team did their best to maintain naming conventions for the development of the mobile portion, the Web API and the Database. There should really be no case where such conventions were not upheld.

C#

|  |  |  |
| --- | --- | --- |
| Identifier | Convention | Examples |
| Class names | PascalCase | public class UserLog{} |
| Methods | camelCase | public void addUser() |
| Class Member Names | camelCase | int itemCount …. |
| Named Constants and Method Arguments | ALL\_UPPER | public const USER\_TYPE = “regular”  public void AddUser(LogEvent) |

mySQL

Naming Conventions

In general:

* Use lowercase letters
* Spaces are replaced with underscores
* Alpha English letters
* No prefixes

Database naming convention:

* Can be singular or plural
* Avoid prefixes if possible

mySQL table name:

* Table names should be singular
* Use prefixes if necessary

Field names:

* Lowercase
* No spaces but underscores
* No numbers
* No prefixes
* Choose short names (typically no more than two words)

***Method Arguments***

Methods are used for carrying out a set amount of functions. Usually the name will reflect the function that the method is supposed to execute. Providing the correct arguments with the appropriate data types, the methods take in such parameters and execute their function accordingly.

If the method calls for a string parameter, there is no need to notate it differently than it is supposed to be notated. The same convention will still be upheld if the parameter is of int or any other parameter. Also the same conventions hold true for booleans as well.

Keeping in mind that a method can have an argument that calls another method in order for the method to output the intended return value.

Example- method call as arg1, int as arg2

Create.table(this.getClass.getUsername(), 1);

Example- int for both arg1 and arg1

Update.cloth(12,13);

***Parenthesis, Braces and Indentation***

Public void example(bool variable)

{

//codes goes here

}

This would be an example of how parenthesis can be used. The parenthesis in this example is used for taking in arguments. Braces in this example are used to section off certain parts of the code and go directly underneath the method signature and the body of code. Everything inside of the braces gets indented one time.

***Control Structures***

If statements

if(condition)

{

//code goes here

}

else

{

//code goes here

}