Automation of Library (Codes) Development for Content Management System (CMS)

Mustapha Mutairu Omotosho
Department of Computer Science, University Of Ilorin, Ilorin. Nigeria.

Balogun Abdullateef Oluwagbemiga
Department of Computer Science, University Of Ilorin, Ilorin. Nigeria

ABSTRACT
Previous researchers have found that most programmers or web developer usually have problems with the generation of libraries, creation of databases and the integration of the two aforementioned. These problems also include the re writing of libraries for different applications, maintenance of already written libraries in case of any changes in database or vice versa. Various kinds of open source softwares are usually used in the creation of web applications or content management systems. However, these problems are relative due to the level of the programmer and the programming paradigm employed by the programmer. This paper presents a Library (Codes) Generating Machine that can be used to solve these problems, for it automatically generates libraries and creates databases based on the approach employed by the programmer. The two ways that are adopted in the usage of this system are either starting from the creation of database and its tables (Backward Approach i.e. starting from database design) or from the creation of class(es) (Forward approach i.e. starting from object identification).

General Terms
Software Engineering, Algorithm and System Automation, Code Generator

Keywords
Backward Approach, Class, Code Generator, Database Design, Forward Approach, Object, Object Oriented Programming (OOP).

1. INTRODUCTION
Since the the creation of world wide web(www), several web modeling methods have been proposed in order to provide efficient and structured ways of developing web applications. However, these web modeling methods only provide design primitives for building web applications from scratch, and not including the possibility to design preexisting components related to a particular web domain [1]. Web Content Management (WCM) systems are computer softwares that are used for the overall management (include but not limited to creating, editing, deleting etc.) and control of web contents. The merger of WCM systems with web applications resulted into Contents Management System (CMS) based web applications [2]. The implementation of WCMSs is a complex task, since one has to cope with fast-changing requirements which have a high impact on architecture and design. Key in this process is the translation of business requirements into preexisting components. In order to make this process more transparent and effective, domain-specific web design methods are needed [3].

Current web modeling methods provide their own specific set of models including a various number of design primitives. These models address design primitives which have proven their usefulness in many cases. Therefore, it seems obvious to select existing web modeling methods for the creation of a situational web design method with regard to the WCM domain. Due to the high amount of web modeling methods, each with their set of models, it is complex to make a decision on which web modeling method to choose for applying this to the WCM domain.

In this research, we present an approach where in libraries and databases can be automatically generated and integrated using a code generating system in a uniform way.

The code generating system, generates libraries which are made up of useful methods or functions that can be used by the programmer or developer, and it also creates databases suitable for a particular domain specific application based on the programmer's choice of approach. Usually there are two approaches used by programmers or web developers in creating an application. It could either be the forward approach (in this case the programmer starts the creation of the application by first creating the class(es) that will be used in the implementation of the application and there on) or the backward approach (in this case the programmer starts the creation of the application by first creating the database of the application and there on). Another advantage of using the code generator is that it can be used alongside other softwares used by programmers and it also adopts the object oriented programming style as its programming style. **Object-orientation** (**OO**) is a programming paradigm that concieved a problem as collection of objects (object is a collection of datafield i.e. properties and methods i.e. behaviours together with their interactions.) and solution to the problem as the interactions of the objects. Thus the modelling of these objects and their interaction (i.e. sending messages to each other) into computer programs is known as Object Oriented Programming (OOP). This programming techniques may include features such as data abstraction, encapsulation, messaging, modularity, polymorphism, and inheritance. Many modern programming languages now support OOP, at least as an option. Simple, non-OOP programs may be one "long" list of statements (or commands).

The remainder of this paper is organized as follows. In section two, we describe the code generating system. In section three, we describe the implementation of the system, section four deals with the testing, output and the concept of the usability of the code generating system. We end this paper with a brief conclusion.

2. ARCHITECTURE OF THE SYSTEM
The code generator is a web application designed for the automatic generation of libraries and creation of databases for content management systems. It is created to remove the redundancy in re-writing codes, provides flexibility in making changes subsequently, ensure great readability, increase

development speed and to save the programmer from the rigorous of having to create libraries, create databases and linking them together.

The software is coded using Hypertext PreProcessor (PHP) language, Cascading Style Sheet [CSS], Extensible Hypertext Markup Language [XHTML]. The programming paradigm used is Object Oriented Programming [OOP].

Basically there are two ways employed in the usage of this

system which are discussed below.

BACKWARD APPROACH

In this approach, the creation of the CMS starts form the Database, the programmer or developer will have to create a database for the CMS and then the code generator will generate the libraries with the respective classes of the CMS. The code generator will map each table as a class, each field in a table as a variable.

The code generator will take each class as a table in the database and each field in each class will be mapped as a field in the table created from that class.

The code generator will map each table in the database to be a specific class in the libraries and the fields of a table will be the variables of the class mapped to that table. The field and the class will have the same property as specified by the programmer or developer.

FORWARD APPROACH

In this approach, it involves the programmer or developer starting the creation of CMS by first identifying all the objects that constitutes the system and then creates classes that all those object can be created but its only the attributes of those objects that will be in the classes created. Then the code generator will generate the libraries which contain the Data Access Methods and other useful methods that will be used by the programmer and it will also create the database for the CMS as specified by programmer or developer. Each field in the class will have an accessor (i.e. get and set) method generated by the code generator.

In this approach, the programmer will have to save the folder containing the classes into the code generator folder.

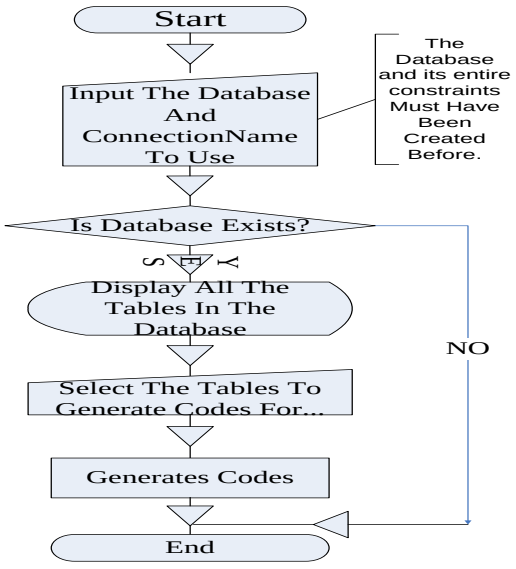

Fig 1: Architecture of the system using Backward Approach

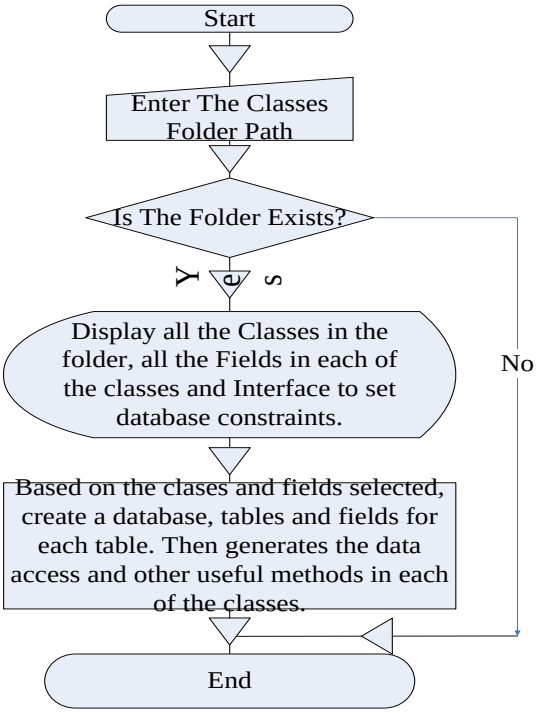

Fig 2: Architecture of the system using Forward Approach

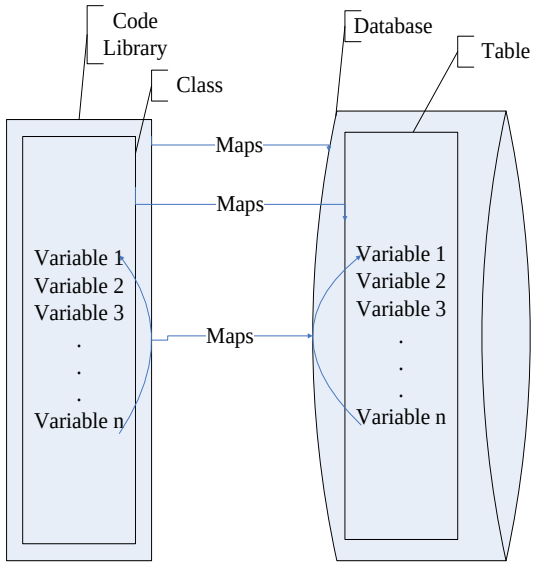

Fig 3: Diagrammatic representation of the Library generated by the system and the Database.

3. IMPLEMENTATION OF THE SYSTEM

The Code generator software is made up of the following

- **INDEX PAGE**
- **GENERIC PAGE**
- **THE MAIN LIBRARY** which contain the **CONFIGURE CLASS, THE FIELD CLASS, TABLECREATOR CLASS AND THE ENGINICODUU CLASS**

3.1.1 INDEX PAGE: This is the first page on the software, which serves as the interface for the programmer to use. The index page is the starting point where the programming starts from. The programmer is to choose one out of the two approaches made available by the code generator.

3.1.2 GENERIC PAGE: This is an helper page that present an interface for the creation of database and other database constraints.

3.1.3 MAIN LIBRARY

This is the main part of the software where the major work is done and the part where all other parts rely heavily on. The main library of the software contains four classes that it used for generation of code for the CMS. Which are;

- **CONFIGURE CLASS**
- **FIELDS CLASS**
- **TABLE CREATOR CLASS**
- **CODENGINE CLASS**

CONFIGURE CLASS: This class contains methods that are used for connecting the content management system to the database.

FIELD CLASS: This class contains methods that are used for creating fields in the database.

TABLECREATOR CLASS: This class contains the methods for creating tables in the database.

CODENGINE CLASS: This is the main class where every other class depends on. Its purpose is quite enormous since it is the central of all other classes. It contains so many methods that are used for different things. The methods that made up the CodEngine class are:

doGenerate Method: This method is used for accepting all the selected tables from the database and it gets all the fields for each of the selected tables and in turn passes each table and its field into another method called the KoCoduuFunClassiKokan Method which actually generates the codes by accepting the table's name and it's fields.

createExtenderClasses Method: This method is used for creating code for extender classes that can be used to add additional functionalities to the classes generated initially by the code generator. It generates an extender class for all the tables selected. It also creates a platform for the user to use user defined code with the generated codes.

KoCoduuFunClassiKokan Method: This method generates the codes by accepting the table's name and field's name which is passed into this method by the doGenerate Method. It also gets all the variable attached to each field and set the variable to a particular value.

getSetproperty Method: This method generates a function that will assign the fieldname properties of an object passed into it to the value supplied.

getGetProperty Method: This method generates a function that will return the value of the assigned fieldname properties of an object passed into it.

getVarOutOfFields Method: This method create PHP variables from the database table's fields by appending dollar sign "$" before them and also assign a private access modifier to each of them and by extension adopting the encapsulation concept of OOP.

startFromDb Method: This is the method invoked when the backward approach is being used by the programmer. It checks, creates and replaces directory for the libraries to be generated. It also establishes connection to the database by creating a connection object.

getAllTables method: This method gets all the tables in a database, it creates array and then passes all the tables name into the array. But if the database is empty, then the array will be null.

getAllFieldsForTable Method: This method creates an array which it stores all the fields in a table into and then setting the properties of the fields appropriately. But if the fields are empty the array will be null. It also sorts the field accordingly.

getConstructor Method: This method generate a constructor to construct object of a class that is being generated. It takes in all the fields in a table and then generates a constructor for them.

generateGeneral Method: This method creates inheritable general classes that will contain all methods to handle all sorts of uploads such as image, file etc.

generateConfigure Method: This method is used for generating the method for configuring an application by connecting the application to a database and then returns the hostname, DNS and all other parameter.

generateDbAccessor Method: This method generates a method that run queries on database and returns the result as a set of query. it is best used for select, select count, etc. This method also establish connection to your database by calling connectTodb method on Configure object.

generatePrepare Method: This method makes proper correction to any parameter supplied and returns the modified value with single quotes appended. So if this method is called in an sql statement, the value should not be enclosed by single quotes as this method will do that for the programmer. It prevent sql injection and prepare sql statements appropriately before it is executed on the server.

getInsertDataAccess Method: This method generates the function that is used to Insert an object of a class Into Its Corresponding database Table. The method takes in the table name and its fields and then generates the method.

getSelectByEachColumnsDataAccess Method: This method generate methods that select records from the database based on each of the table's fields, and also construct objects of the class generated and populate them with the records. It returns an array of those objects.

getDeleteDataAccess Method: This generate a method that is used for deleting any object of a class from the database table. It is also used for deleting tables also. This method deletes based on the value passed into it by the programmer when invoked. It reduces the problem of the programmer having to write lengthy codes for deleting in the program. What the programmer just have to do is to instantiate the class and then pass the argument of the object to delete.

getUpdateDataAccess Method: This generate a method that is used for updating the object of a class in its corresponding database table. Since each record will have their unique field(s), this implies that each object will also have unique property and hence, each object will know how to update

itself by using its unique property. But the software is robust enough for the programmer to create its own update method.

getSelectDataAccess Method: This generate a method that is used for selecting all records in a particular table and in turn constructs an array of object of the generated class from it.

GetSelectAllDataAccessWithOrderCondition Method: This generate method that selects all records in a particular table and then construct array of objects of its class from it. The object will be ordered based on the condition(s) supplied by the programmer whether ascending or descending order.

getSelectAnyDataAccess Method: This generates a method that select records from database even if it is not all the fields of the database table that is selected, it will still load the selected field for the object and in turn construct an array of objects of this class that will be used by the programmer.

generateCompare Method: This generate methods that are used for comparing the supplied object with the current object calling this method. It returns true if the object are equal by attributes, and value and they are both instances of the generated class otherwise false if there are not. It is also used for generating method that compares the supplied object with the current object calling this method. It returns true if and only if they refer to the same instance of the class otherwise false. The latter method generated by this method strictly compares the two objects while the former just compare based on their parameters.

All the aforementioned makes up the software where each of the methods are related together based on their respective functions.

Below are the Algorithms for the implementation of both Forward and Backward approaches.

```
Algorithm Forward (Classes Folder) {
 If (Classes Folder exists) {
Select Classes to generate code for;
Create Database
For Each Class Selected {
All Fields= Get All the Fields for the Class;
Generate Code (Class Name, AllFields)
Create Database Table (ClassName, AllFields)
If (ExtenderClass){
Generate Extender Class (ClassName, AllFields);
}
}
}
Else
{
  PRINT No ClassFolder;
}

Algorithm Backward (Database){
 if (Database exists) {
While Database Not Empty {
Display Tables in Database;
Select Tables to generate code for;
Create Folder To Keep generated Codes;
For Each Table Selected{
AllFields= Get All Fields for the Table;
Generate Codes( Table Name, AllFields);
If (ExtenderClass){
Generate Extender Class (Table Name, AllFields);
}
}
}
}
}
Else
{
        PRINT  No Database;
}
```

4. TESTING OF THE SYSTEM

Based on the algorithm given in chapter 3 on how libraries or codes for CMS can be generated, we are going to give proper illustration on how it works using both approaches:

4.1 TESTING USING FORWARD APPROACH

For the Forward Approach, this involves starting the creation of CMS from the creation of classes by the programmer. These classes are what the programmer will need in the course of designing the CMS. All the classes must be well defined and specified and they should be saved into a named folder and the name of the file must be the name of the class (i.e. a single file will contain a single class). The named folder must be in the code generator folder for the code generator to gain access into it. The classes to be created by the programmer are not limited to a particular number, but it has to be well specified. Each class should contain the appropriate variable fields meant for that particular class.

For example, a folder named UserReg which contains all the classes as specified by the programmer which was saved into the code generator folder contains the following classes:

File Name is Admin.php

```
<?php
class Admin
{
        private $adminUser;
        private $adminpass;
        private $admin_id;
}
?>
```

File Name is User.php

```
<? php
class User
{
private $Name;
        private $address;
        private $sex;
        private $phone;
        private $user_id;
        private $userName;
        private $password;
}
?>
```

File Name is News.php

```
<? php
class News
{
        private $news_id;
        private $newsBody;
        private $newsHead;
        private $dateTime;
}
?>
```

Clearly we can see that the folder UserReg contains three classes i.e. Admin Class, User Class and News Class (N.B: These classes are just generated just to explain the implementation of the code generator using the forward approach).

The classes are well specified and defined, with each having well defined variable fields. As stated in chapter one that the language for implementing the code generator is PHP and the programming style is OOP which brings about the declaring of the each variable in each class private. The private access modifier just serves as platform for information hiding and hence encapsulation.

The next step is to use the code generator to create the libraries, codes and database for the CMS from the classes created.

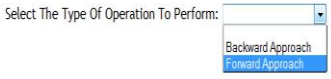

Fig 4: Selection of Development approach interface.

The type of operation we are considering first is the FORWARD APPROACH, after selecting it, the next step is to provide the name that the programmer wants the code generator to give the database and specifying the name of the folder where the created classes are(Project Folder).

The name of our project folder is UserReg as stated above and let the name of our database to be ProjectDB as shown below.

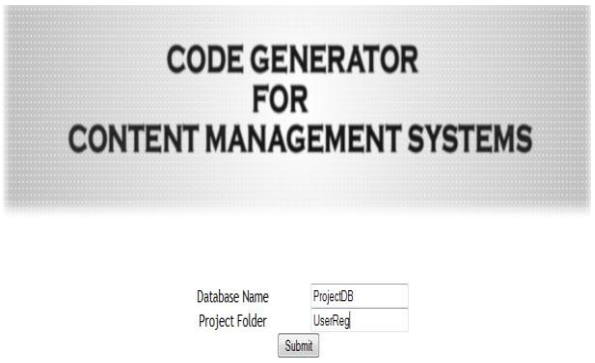

Fig 5: Specifying the name of the database and the project folder.

After specifying the database name and the project folder, the code generator will then create an interface for the programmer to select the properties of the database to be created as shown below.

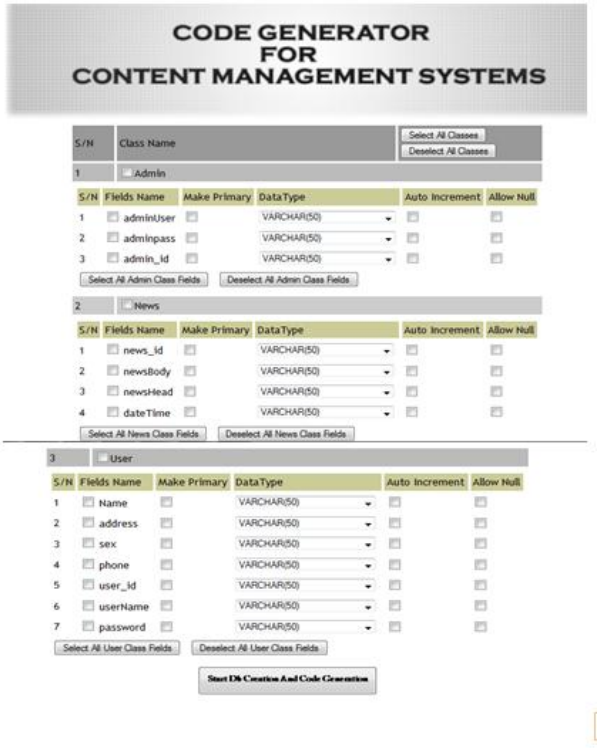

Fig 6: Selection of tables and fields to form database.

Through this process, the database will be created by the code generator as specified by the programmer or developer. Once the programmer click on the Start DB and Code Generation Button, The code generator will automatically create the

specified database and also generate the libraries for the CMS as specified by the programmer or developer.

4.2 TESTING USING BACKWARD APPROACH

This approach is synonymous to that of the forward approach. It involves the programmer or developer starting the creation of CMS from the database. Using this approach in accordance to the code generator, the programmer has to create the database for the CMS which must be well specified. The very first step here is to select the type of approach to use (see Fig 4), but we will select backward approach here.

For illustration, the database created by programmer is named project1. The database contains two tables named User and Parameter, each table has two fields each(N.B: The database is just created to implement the code generator).

After selecting the backward approach, the name that the programmer wants to give the application must also be specified for this will be name that the code generator will use to save the libraries of the CMS, the name of the database created must also be specified in this case we will use projectDb.

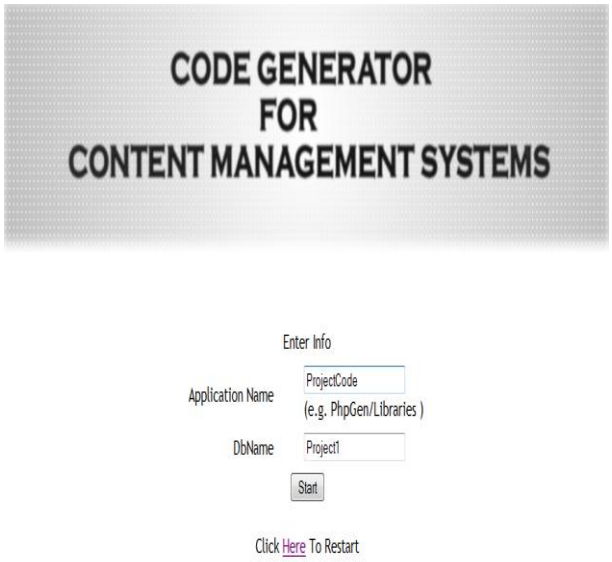

Fig 7: Specifying the name of the database and the name of the application.

By clicking the start button, an interface is presented that shows where the programmer can select the tables to generate code for. The code generator will generate libraries based on the database created. Each table in the database will be a class and each field in a table will be the class variables. The code generator also allows the programmer to have extender files that contain classes that inherit from the classes generated and hence allow programmers to extend the functionality of the code generated without modifying the generated codes.

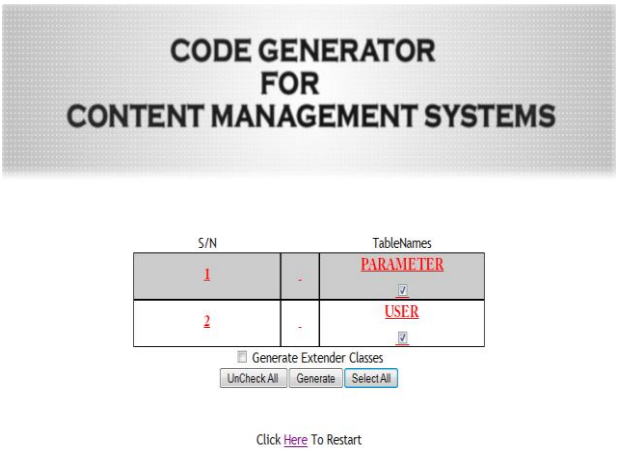

Fig 8: Generating the Libraries and the Extender Classes from the Database.

4.3 THE OUTPUT OF THE SYSEM

The libraries generated by the code generator contain the data access methods that will be used by the programmer or developer in creating a CMS.

The data access methods created are as follow:

INSERT METHOD: This method is used for inserting records into the tables in the database. It will insert values of the instance variables to their corresponding fields in the corresponding database table. If there is a table with primary key (Auto Increment), the insert method will return the primary key to the record just inserted but if otherwise nothing will be returned.

UPDATE METHOD: This method is used for updating an object of a class which in turn updates the database. This method is not used by all tables. It's only table with primary key that will have this method generated otherwise a space will be provided for the manual implementation by the programmer.

DELETE METHOD: This method is used for deleting an object of a class in the database. Deleting an object from a class uses another special method (RunScalar Method), due to the fact that if an object is being deleted there will be no return value. A table without a primary key will have a different delete method where the sql statement will be run directly.

SORT METHOD: This method is used for sorting an array of objects of a class by a specified property of the object.. This will allow the programmer to sort arrays of objects or of records.

SELECTALL METHOD: This method is used for selecting all records in a particular table. This method is used with another special method (RunSelect Method). Since this method selects all records from a particular table, then it must return a value. The value(s) returned are not in order, it only selects it the way it is in the database.

SELECT ALL BY ORDERING METHOD: This method is quite similar to the SelectAll .method but the difference between the two is that the Select All By Ordering returns the value in a particular order, based on the condition specified by the programmer or developer.

SELECT WITH CONDITION: This method is used for selecting records based on "where clause" and then construct an array of objects of the class from it but the SQL statement

will be provided by the programmer or developer. Usually useful in the place of using Select with where clause. It is also used with the RunSelect Method since a value is going to be returned.

SELECT SINGLE METHOD: This method makes it possible for the programmer or developer to select a single record based on the value of the primary key supplied and it returns a single object of the class.

SELECT ANYHOW METHOD: This method select records from database even if it is not all the fields of the database table that is selected, it will still loads the selected field for the object and construct an array of objects of this class.

SELECT BY COLUMNS: Instead of the programmer or developer writing sql statement his/herself, the programmer can just supply the columns names, the value of the columns and the connectors that he/she wants to use in querying the database as an arrays and the method will generate the sql statement and select the records to construct an array of objects of this class.

COMPARE METHOD: This method is used for comparing two objects; it compares the supplied object with the current object calling a particular method. It returns true if the object are equal by attributes and value and they are instances of that particular class otherwise false

STRICT COMPARE METHOD: This method is similar to the compare method, this method compare the supplied object with the current object calling this method. it returns true if and only if they refer to the same instance of the this class otherwise false.

All these Data Access Method generated by the code generator make it easy for the developer or programmer to access data or program properly.

The code generator also gives room for user defined methods also, which can be added to the data access method through an **EXTENDER CLASSES**. The user defined methods should be placed into the extender file which comes together with the main library since the extender Classes inherited the generated Classes.

5. CONCLUSION

With the emergent and continuous growth of internet, fundamental content management has also grown. No longer can information be published in a manual process and left unattended. Online information must be continually reviewed and updated by content editors. The internet forced subject matter experts to more rapidly maintain and update information for their constituents. Content management was born out of these increased published needs [5].

This paper analyzes the processes involved in creating or building a content management system. The existing or conventional way that developers or programmers create or develop web applications or content management systems involve the programmers or developers creating databases and creating the libraries for the content management system or web application which is very difficult and tedious for developers. It is a generic problem which all developers or programmers have to solve for them to create a web application or a CMS. Though it is relative among developers or programmers but they are the major problems encountered by them. This paper provides a code generator system that solves the problems of creating libraries and databases for a content management system by the developer. The code generator automatically generates libraries for any application based on the database created by the programmer or developer and it automatically generate a database and connects it to the application based on the classes created by the developer.

6. REFERENCES

[1] Deshpande, Y., Murugesan, S., Ginige, A., Hansen, S., Schwabe, D., Gaedke, M. & White, B. Web Engineering. J. Web Eng., 1, 3-17 (2002).

[2] Souer, J., van de Weerd, I., Versendaal, J., & Brinkkemper, S. (2007). Situational requirements engineering for the development of content management system-based web applications. Int. J. Web Eng. Technol., 3(4), 420.

[3] Karlsson, F. & Agerfalk, P. (2004). Method configuration: adapting to situational characteristics while creating reusable assets. Information & Software Tech., 46(9), 619-633.

[4] http://en.wikipedia.org/wiki/Object-oriented_programming Visited on 2nd of August 2012, at 9:45pm

[5] http://en.wikipedia.org/wiki/Content_management_system Visited on 12th of August, 2012 at 3:27pm.