Software Design Document

**Version 2.0**

**Prepared on**

**Date: (23/04/2013)**

**1. INTRODUCTION**

**1.1 Purpose**

The purpose of the Detailed Design Document (DDD) is to deﬁne the design, architecture, components and interfaces for all components of e-Complier system which are speciﬁed in the SRS.

**1.2 Scope**

Online compiler is to create a generic portal where codes of programming language can be compiled online conveniently. The compiler will be set up on the server. On submission of code by the client, it will be compiled and executed on the server. The output thus generated is displayed on the client browser.

**1.3 Overview**

This document is intended to be read by:

* Client
* Developer
* Tester
* Project manager

**Client**: To get the clear idea of this software and its specification.

**Develope**r: To understand the algorithms and methods to develop the software.

**Tester**: To write the test cases for various inputs and check for the errors.

**Project manager**: The project manager uses this document to verify whether the end product has matched with customer requirements and also for quality checking.

**1.4 Reference Material**

[*http://cakephp.org/*](http://cakephp.org/) *referring to the mvc model*

Software Engineering, Roger Pressman, 5th Edition

<http://www.gliffy.com/gliffy/>

<http://www.tutorialspoint.com/uml/uml_deployment_diagram.htm>

<http://www.tutorialspoint.com/uml/uml_component_diagram.htm>

**1.5 Definitions and Acronyms**

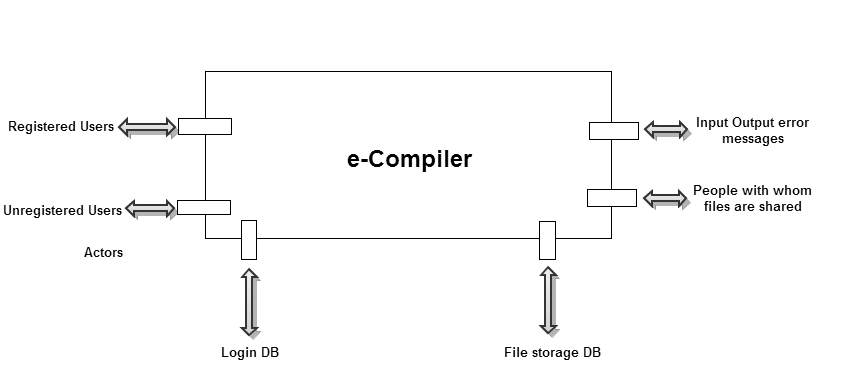
|  |  |
| --- | --- |
| GUI | **Graphical User Interface**  A type of user interface that allows users to interact with electronic devices using images rather than text commands. |
| Server | A computer system that selectively shares its resources |
| Web Application | A computer software application that is accessed by users over a network such as the Internet or an intranet. |
| Web Browser | A software application for retrieving, presenting and traversing information resources on the World Wide Web. |
| MVC | Model View Controller |
| DB | Database for storing information |

**2. SYSTEM OVERVIEW**

The product facilitates the user to choose from a given set of programming languages and compile and execute the code. The user can share the code with other users. The user’s data and preferences are stored in a database on the server.

The code can also be saved on the server for further references.

The numbers of users who can access the product simultaneously is dependent on the server capacity.

The design document gives a detailed overview of the architecture, components and interface of the system as described.

**Figure 2.1: Architecture Context Diagram**

Fig 2.1 shows the various actors and database interactions

The Actors for our application are the Registered Users and the Unregistered Users. These two actors have each an interface attached to it.

The Registered Users need to Login to use the application where they can enjoy the privilege of saving and sharing their code with others via e-mail.

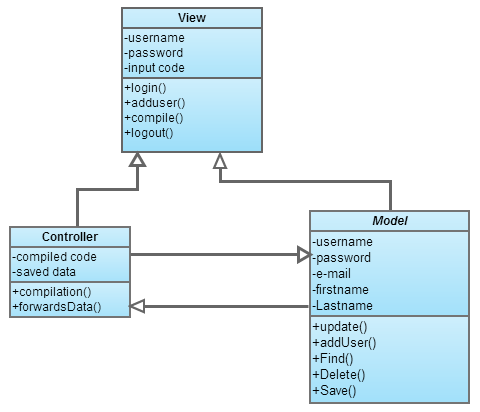
The unregistered users can use the application for writing, compiling and executing the code.

The e-compiler depends on the Login Database for storing the information of the Registered Users and the File Storage Database for storing the codes saved.

Additionally, there is the I/O error messages interface and the interface where code can be shared with other users via emails.

**3. SYSTEM ARCHITECTURE**

**3.1 Architectural Design**



**Figure 3.1: Archetype**

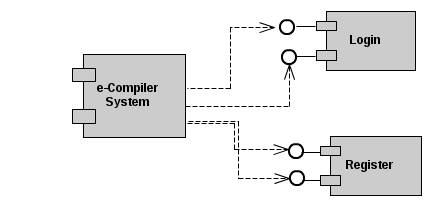
The archetype separates the logic from the implementation. Cakephp framework utilizes MVC model. Hence fig 3.1 depicts the various interactions of model, view and controller.

The model interacts with the database for updating, adding, finding and saving information.

This information is then passed on the controller accordingly to present different views to the client.

.

**3.2 Decomposition Description**

****

**Fig 3.2 shows the detailed flow of the application.**

One component of the DFD is elaborated in fig 3.2. This depicts the authentication component of the system.

The authentication component can be divided into two categories

Login component takes the username and password as input, finds for a match in the database and appropriately provides success or failure message.

The register component is for a new unregistered user. This lets the user fill a form and submit it to the DB .This user is registered by adding the user information and can hence access the save and sharing facilities of the system.

**3.3 Design Rationale**

e-Compiler being a web based application ,MVC model is the most apt model. Here the business logic and user views can be separated conveniently from each other as well as the DB.

Critical issues:

● Database interaction

● User interaction

● User Experience

● Load on server

The DB interaction is achieved using the model which interacts with the controller to information to the view. The user interaction is separated from the business logic in the controller.

Hence the following advantages of can be listed out

Advantages of MVC:

Separation of concerns

Complete abstraction

Further extension is easy

Reusable

A functional approach is also taken as it’s a web application and is not data centric but

more towards the flow of data through the various states.

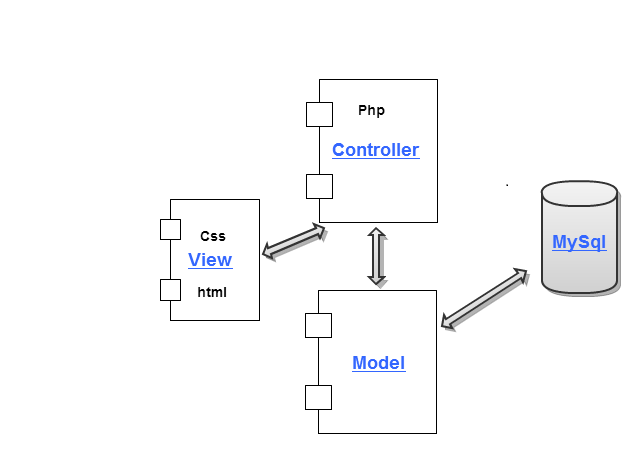
**4. DATA DESIGN**

**4.1 Data Description**

MySql database is used for maintaining and organizing the data present in application

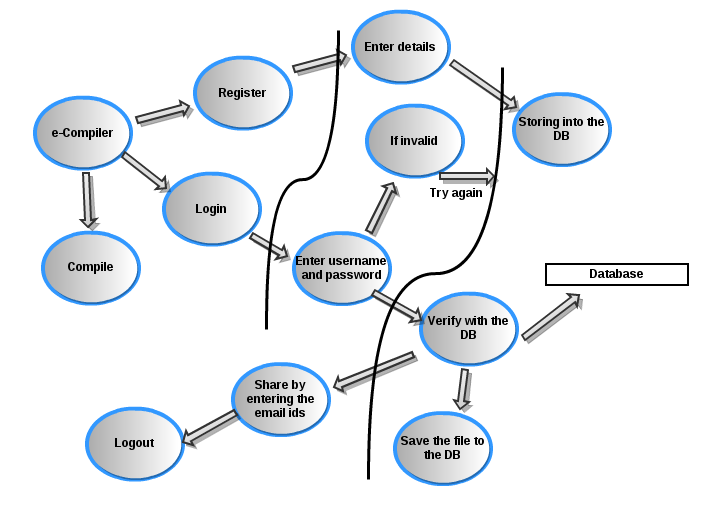
* 1. Since the major application of the project is to compile the program submitted, the database consists of fields for storing the programs which holds the information such as the file name, date and time of creation of file and date and time of modification of files.
  2. For creating a new user, a registration form is utilized which holds necessary data about the user entering the application. The fields include First name, Last name, User name, password and e-mail id .This is stored into the database along with an id which is unique to the user. This id serves as a foreign key for maintaining the programs saved into the database.
  3. Login page takes username and password for validation which are derived from the previous table.

The above mentioned fields are maintained and organized into two tables in a database namely ‘users’ and ‘files’. The two tables are connected using a foreign key in the ‘files’ table. Hence any addition of programs by a particular user will be respectively updated in the database.

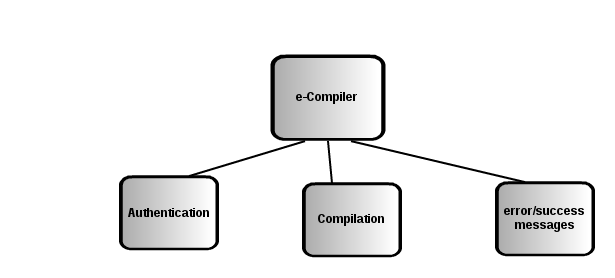


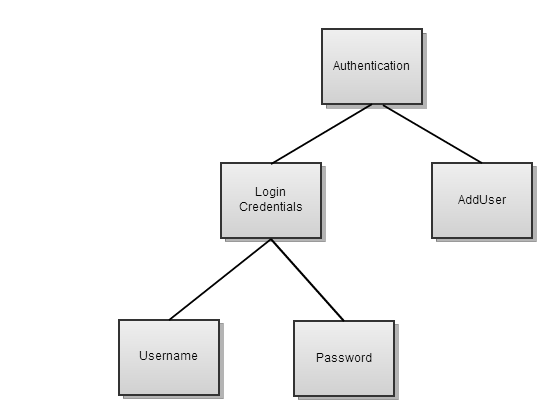
**Fig 4.1 Component Diagram**

The diagram 4.1 gives a very high level structure of the implementation of the application. Cakephp is used as a framework for developing the application, hence MVC model is used. The implementation is divided into Model, View and Controller. Model interacts with the database with retrieval and updation of information. Controller collaborates with view and the model to present to the user the information retrieved by the model.



**Fig 4.2 Data Flow Diagram**



**Fig 4.3 First level factoring**

**Fig 4.4 Second level factoring**

**User registration and Login**

The facility provides users with a privilege to log into the application for sharing and saving the code to the repository.

* **Editor for the user to write code**

Editor is a space where the users can write their code and compile it immediately to produce the output, compile and debug it.

* **Code compilation and execution**

The code submitted by the editor can be compiled and executed online providing output.

* **Code repository**

Any registered users can save their compiled codes online on the repository that can be retrieved for editing or future compilation.

* **Code sharing**

The application provides a unique feature of sharing the compiled code through e-mails.

**4.2 Data Dictionary**

**Created** : Datatype is datetime. This saves the date and time the file was created.

**e-mail id** : This field is validated for e-mail id format.

**File**: This is the text document saved into the database.

**Filename**: characters with maximum of 25 characters.

**FirstName**: characters with a maximum of 25 characters and this field is necessary and cannot be empty

**id** : datatype - Integer with maximum of 10 digits. It is auto incremented assigned a new id to each user.

**LastName**: characters with a maximum of 25 characters and this field is optional.

**Modified**: Datatype is again datetime. This saves the date and time of the most recent modification done to the file.

**Password**: This is stored as entered by user and no restriction is mentioned on the type of the data. It is encrypted and stored in the database for security issues.

**Username**: This field must contain only alphabets and numbers and it must be unique.

**User\_id**: Integer with maximum 10 digits .This is used to link the ‘users’ table to the ‘files’ table.

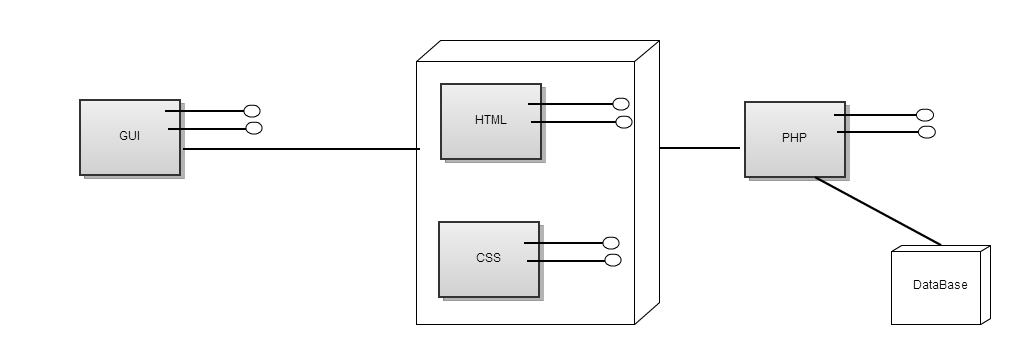
Functions mentioned in 3.2 are described below,

1. **Login**: This function takes user name and password as parameters and compares it with DB and appropriately takes action of entering the session or asking the user to try again.

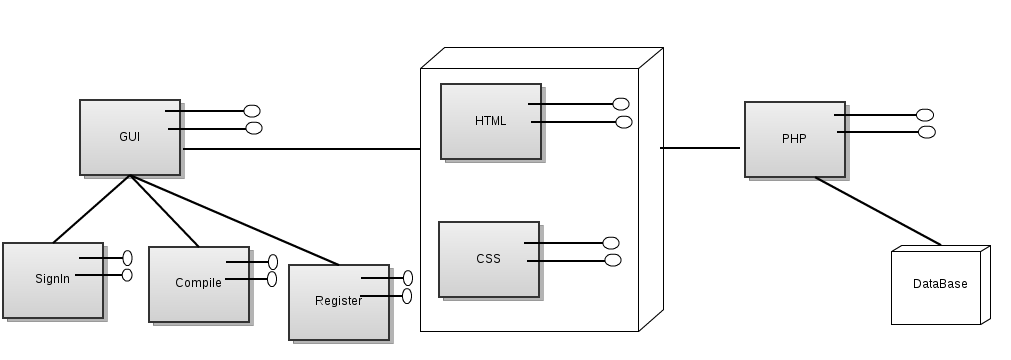
2. **Registration**: This function takes a form with various user details as input and updates it to the DB .It validates the form to check if the necessary inputs are given, i.e. First name is not empty, user name is unique, password is at least 8 characters long. After validation the information is updated to the DB with a new id.

3. **Compilation**: This is the most important function in the application. It takes filename and the program file as parameters and outputs the compiled messages.

**5. COMPONENT DESIGN**



**Fig 5.1 High level component diagram**



**Fig 5.2 High level component diagram**

The diagram 5.2 contains the detailed description of one of the components used in the application.

Cakephp framework uses MVC model for development and deployment.

1. **Session**: A session includes login and logout and all the pages that can be accessed within the session. As mentioned earlier, the compile page can be accessed by a non-registered user too. Hence only the save and share pages are privileged for registered users. When a user logs into the system, the function login is called by the controller and directs the model to check if a match for the entered username and password exists. The encrypted password is decrypted and matched. Once a match is found, the control is returned back to the controller which redirects the page using the view. Logout works by closing the session.

2. **Registration**: This function presents the user with a form with necessary fields. This form is validated by model to check that the necessary fields are not empty, the username stays unique, password at least 8 characters long .Validation is also done for email field. Once validated, the controller calls the function add which adds the new user to the DB.

3. **Compilation**: The code that is input through the view is taken in by the controller. It is compiled on the server and success/error messages are redirected to the view. Simultaneously, if the user intends to save the program, the filename, program and the created and modified dates are entered into the DB for the particular user id which is unique.

4. **Save** : Saving a code is simply done by extracting the user id for the logged in user and the fields are loaded into the DB.

5. **Share**: Share is an option where the user can share the desired code with anyone by entering the email address.

**6. HUMAN INTERFACE DESIGN**

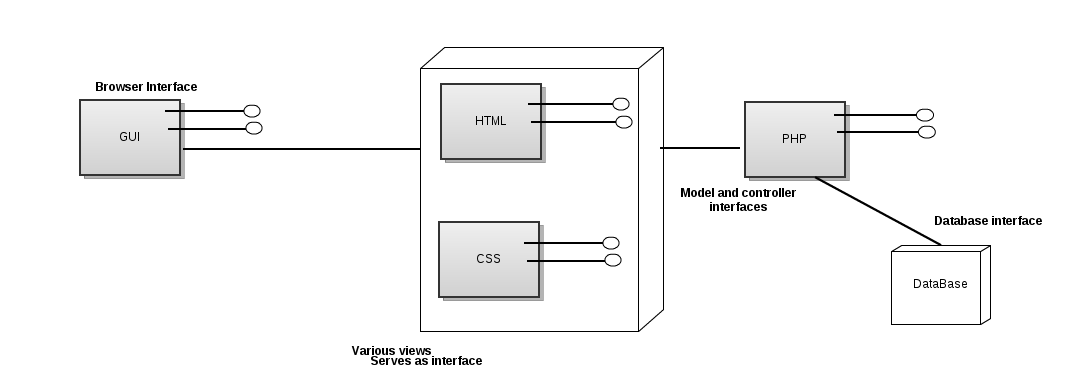
**6.1 Overview of User Interface**

The home page of the application allows the user to compile the code C.

The user can register or login to his/her account by giving the username and password.

Once the user is logged in, the editor space will be available to write the code then compile and execute.

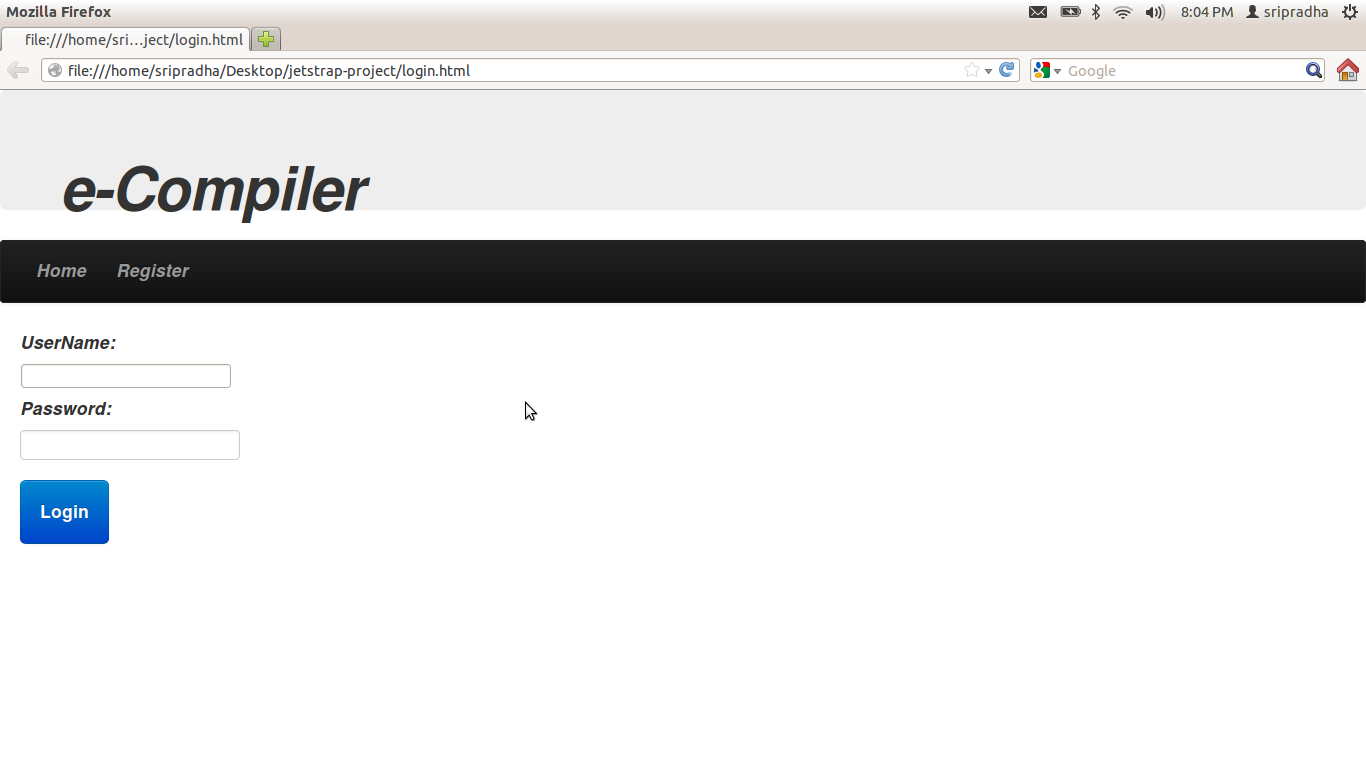
The registered users have the privilege to share and save their code on the database.



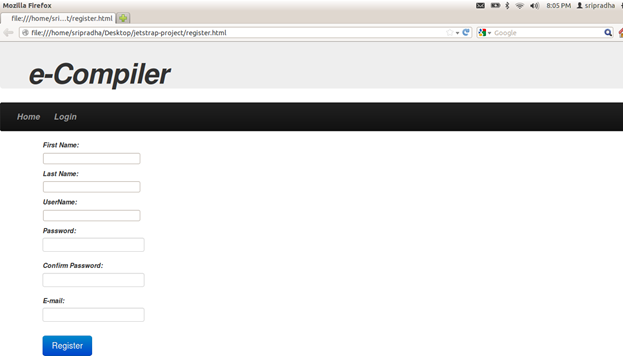
**Figure 5.3 Identification of human interfaces**

**6.2 Screen Images**

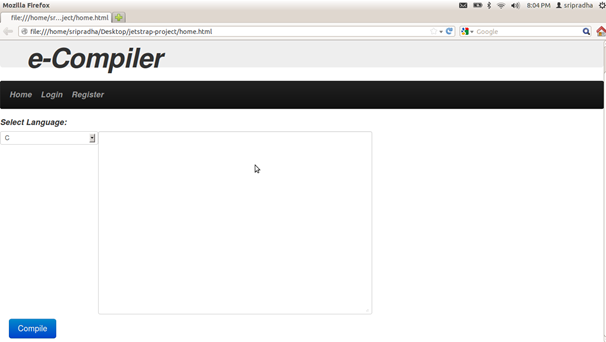
User login screen if the user account exists:



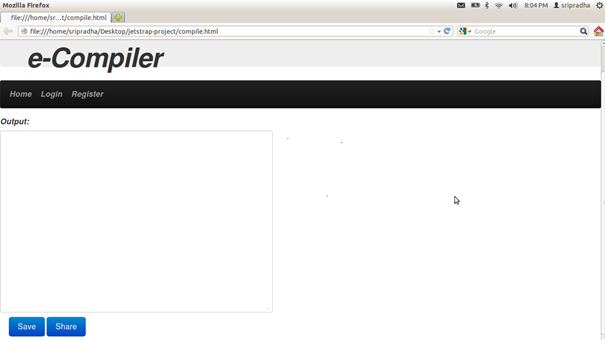
For new user to create an account:

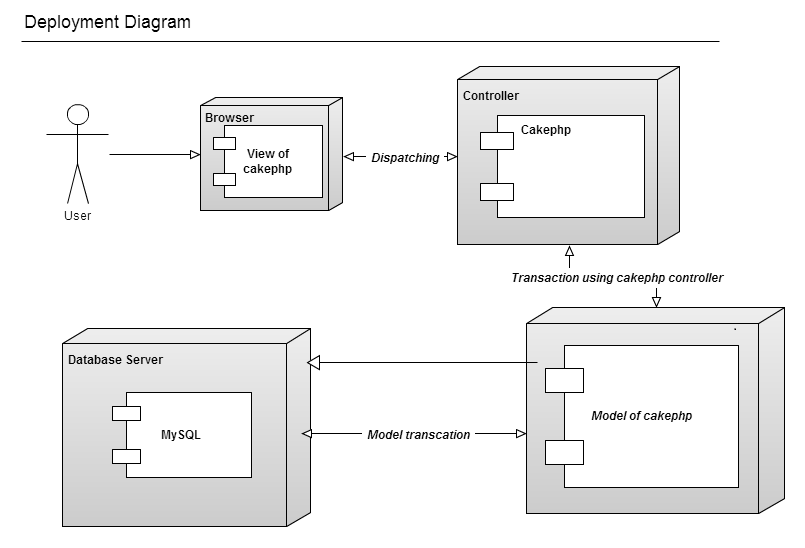


To compile:



Output screen:





**6.3 Screen Objects and Actions**

Login: This button helps the user to login into their account.

Register: This button helps the new user to register by providing the details of the user like first name, last name, user name, password, email address.

Compile: when the user clicks on the compile button, it complies the code and checks whether any error are there.

Share: This button is used to share the code to others by providing the email id.

Save: This button is used to save the code in the user’s account.

**7. REQUIREMENTS MATRIX**

|  |  |
| --- | --- |
| Functional Requirements in SRS | Functions used in code |
| Login | This function takes Username and password as parameters. |
| Register | This function takes a form with various user details as input and updates it to the DB .It validates the form to check if the necessary inputs are given, |
| Compile | It takes filename and the program file as parameters and outputs the compiled messages |
| Save | It takes the code that was entered and saves it to the DB in the appropriate user table |
| Share | Facility to share the code with other users via e-mail |