VITAL EDUCATION PTY LTD

Software Design Document

Version <1.0>

Created By

Raja Sekhar Masina

Anil Rai

Bobby Sri Harsha Sajja

Ashok Kumar Najani

CONTENTS

INTRODUCTION 4

PURPOSE 4

DOCUMENT SCOPE 4

PRODUCT SCOPE 4

GOAL 5

OBJECTIVES 5

STAKEHOLDERS 5

SYSTEM DESCRIPTION 6

SYSTEM FOR USERS 6

DEVELOPMENT METHODOLOGY 8

BENEFITS 9

LANGUAGE USED FOR SYSTEM DEVELOPMENT 9

SYSTEM DESIGN OVERVIEW 10

HIGH LEVEL DESIGN MODEL 11

LOW LEVEL DESIGN MODEL 12

USE CASE DESCRIPTIONS 13

DATA FLOW DIAGRAMS 15

DFD LEVEL 1 (LOGIN PROCESS) 15

DFD LEVEL 1 (BOOK COURSE) 16

DFD LEVEL 2 (PAYMENTS) 17

ARCHITECTURAL MODEL VIEW 18

ARCHITECTURAL DESIGN OVERVIEW 19

LIMITATION IN DESIGN: 20

STATIC/DYNAMIC ASPECTS: 20

FUNCTIONAL AND NONFUNCTIONAL ASPECTS: 20

GUEST USER 20

ENROLLED USER 21

ADMINISTRATOR 22

DATA DICTIONARY 22

FILE FORMATS 23

DATA FLOW DIAGRAM LEVEL 0 24
ER DIAGRAM 25
ACTIVITY DIAGRAM 26
SEQUENCE DIAGRAM 27
DETAILED DECOMPOSITION OF THE SYSTEM 28
CODE-SPECIFIC DESIGN SPECIFICATIONS 30
GUI DESIGN 32
USER WEB SYSTEM 32
HOME SCREEN 32
COURSE PAGE 33
ADMIN SYSTEM 34
COURSE LIST 34
UPDATE COURSE 35
DESIGN SPECIFICS 36
REFERENCES 37

INTRODUCTION

This Software Design Document is created to provide an in-depth information about a web application that is being developed for Vital Education Pty Ltd. This document contains descriptive as well as graphical representation of the software models such as use case models, sequence diagrams, collaborative models, object behavior models, ER diagrams and other supporting documents for the software development processes.

PURPOSE

The purpose of this document is to clarify possible doubts and confusions about this software and give detail understanding about the system for users. This document aims to satisfy users' questions and queries about the software and its usability. It further tries to explain software architectural design and built in processes for future expansion and maintenance. It will serve as a guideline for new users as well as system maintenance engineer of Vital education Pty Ltd.

DOCUMENT SCOPE

This Software Design Document is for base level system which will be serving as conceptual document for users and system maintenance professionals. It emphasis on system building models and the functionalities of the system. It also focuses on structural design architecture of the application which will be useful for further extension of the software.

About the software scope itself, it will be serving as an excellent software tool for vital education Pty Ltd to manage their students and their courses. It can be used for managing most of problems related to students, staffs and administration of Vital Education Pty Ltd.

PRODUCT SCOPE

The following bullet points show the scope of the product:

- A dynamic website for users to manage their courses.
- Interactive software for admin to facilitate their daily tasks in office.
- Database support for data storage and retrieval.
- Moodle system for course contents.

- Online payment system for users.
- Online course booking system for guest users.
- Timetable management for admin and tutors for their concurrent courses.
- Online article blogs for registered and guest users.

GOAL

- To develop easy, efficient and interactive web application for Vital Education Pty Ltd which will be managing all the problems related to students, courses offered staffs and admin of the institute.

OBJECTIVES

- Design a suitable Entity relationship (E-R) diagram for developing back end of the system.
- Construct a database with all the tables represented by the ER diagram.
- Design front end of the system for user interaction. For example home page.
- Create an interactive front end page that can be connected to our database.
- Implement short functional requirements to the system.
- Test the system.
- Implement huge functional requirements like payment system, integrate Moodle system etc.
- Test the system once again and improve the quality of code (refactoring).
- Make documentation of the project.

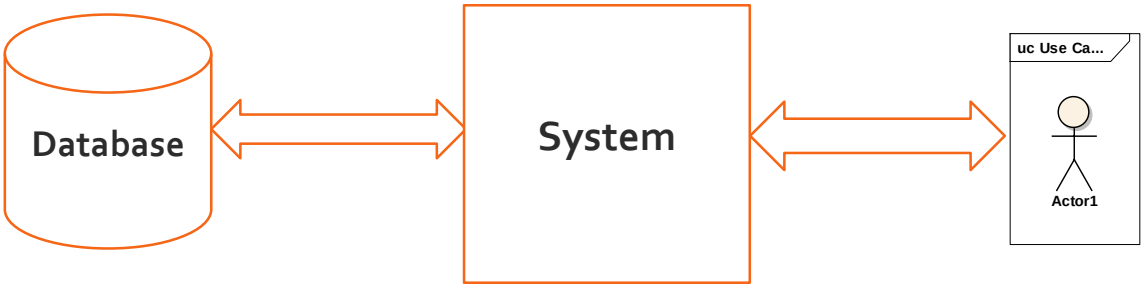
STAKEHOLDERS

There are various stake holders in this project, to name few of them:

- Vital education Pty Ltd
- XP consulting Pty Ltd on behalf of our client.
- Project supervisor.
- Project team members.
- Student of Vital education Pty Ltd.
- Guest users of software.

SYSTEM DESCRIPTION

This web system allows users to interact with its components for manipulation of the queries. It is a 3-tiered application which consists of system database of back end support, web page for front end interacting interface and middle ware to process the information given by users. The middleware will be located in system server which will be looking up for users' request and as user push their request to the server, our program will access system database and extract the required information to server the user. Following diagram shows our system diagrammatically.



SYSTEM FOR USERS

Vital Education is an institute which offers courses in various locations so users (students) faced many difficulties while studying in this institute. This software is here to solve this problems faced by students. Meanwhile, management of institute also has difficulties to manage courses and students. This system will solve all these problems.

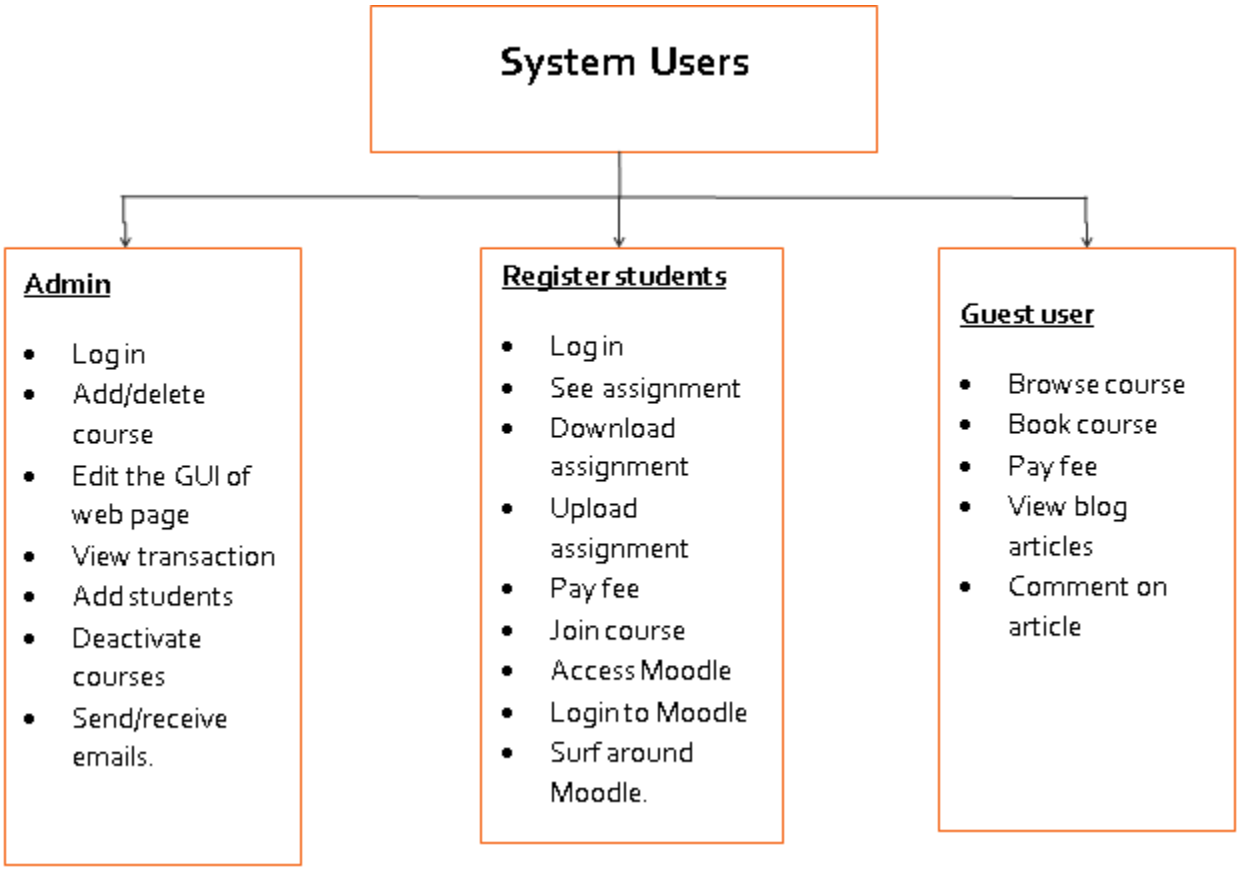
There are mainly three types of users to this system. They are categorized on the basis of user's characteristics for the system. They are admin, enrolled students and guest users.

Users of the system have their own privileges on the basis of their role in the institute. For example, Admin is the one who have highest level of privileges amongst all the users. Admin can log in to the system with secured user id and password. Once he/she is log in to the system can edit all the contents of the courses, students' details, and tutors details.

Similarly, tutors have second priority among other users. He/she can add course contents, upload/download assignments while users like students can download/upload assignment. In the same way, tutors can mark the assignment and upload the marks so that students can see.

Guest users can see all the contents of webpage like courses, web articles. They can add comments to the blog articles. This will be updated in the system as uploaded by users.

Following chart summarizes the user's characteristics and their privileges to the system.



The following table shows use case scenarios.

Use case name	Register to the system
Primary actor	User
Supporting actor(s)	Vital Education System
Summary	Any guest user can register to the system by selecting the registration option available in the webpage
Pre-Conditions	Navigate to the registration page Provide necessary details
Flow of events	Select the register now button Fill all the necessary fields Confirm password Answer the human identification question Hit the register button Open email and confirm the account by clicking the link in the received mail
Exceptions	User email id already given Username already taken Server issues
Post-Conditions	Create a new user in the database Send an automated email

DEVELOPMENT METHODOLOGY

This system is developed based upon agile methodology. This methodology is more adoptive than traditional Software development life cycle (SDLC) as the changes in the requirements can be adapted to the system development process at any development stages. We were required to develop this system with in limited amount of time as per specified in our course description, the selection of agile methodology enables us to develop the system with this period of time.

To refer the properties of agility, we have chosen Extreme programming. The reason behind this is as follows.

- XP is light weight and fast development approach for small projects. We had short duration of time for developing whole application and make documentation of it and then implement it to our client.

- XP provide short term goals and meet that goal in short duration of time. It is again very much suitable for our project because we can set short term goals for our project and meet our client every week because we could meet our client on weekly basis.
- Pair programming is another important feature. Yes, we implemented this as well. Since in our team there were two students who wanted to code, so we implemented this feature as well.

BENEFITS

By the use of stated methodologies, we can jot down following benefits:

- **Requirement Fulfilment:** This methodology helps us to fulfill all the requirements of software development easily. It satisfies the client's expectations.

- **User friendly UI:** Bootstrap framework gives a platform to develop user friendly UI and it is easy to implement to our project.

- **Dynamic Design:** By using PHP, we can create a dynamic website which can solve the required problem to our client. So this language is one of the most useful tools for our project.

- **Reusable:** we can reuse all the components developed in this project because of agile methodology's feature. It is helpful for further extension of the project in future.

LANGUAGE USED FOR SYSTEM DEVELOPMENT

For the development of this software, we have used HTML 5, for front end markup language, JavaScript, PHP for server side language, MySQL for database, and bootstrap as the development frame work.

Bootstrap is a framework for web application development. It emphasizes on front-end development, integration HTML, JavaScript, PHP, and others. The encapsulating property of Bootstrap enables to maintain the system in the future. Independent and integration environment served by bootstrap made us feasible to manage varieties of web application into one umbrella structure.

This framework enables developers to code in fast pace basis with minimum errors. Moreover, it focuses on problem solving approach for beginners so that there will be lots easier ways to find out paths to solve a problem.

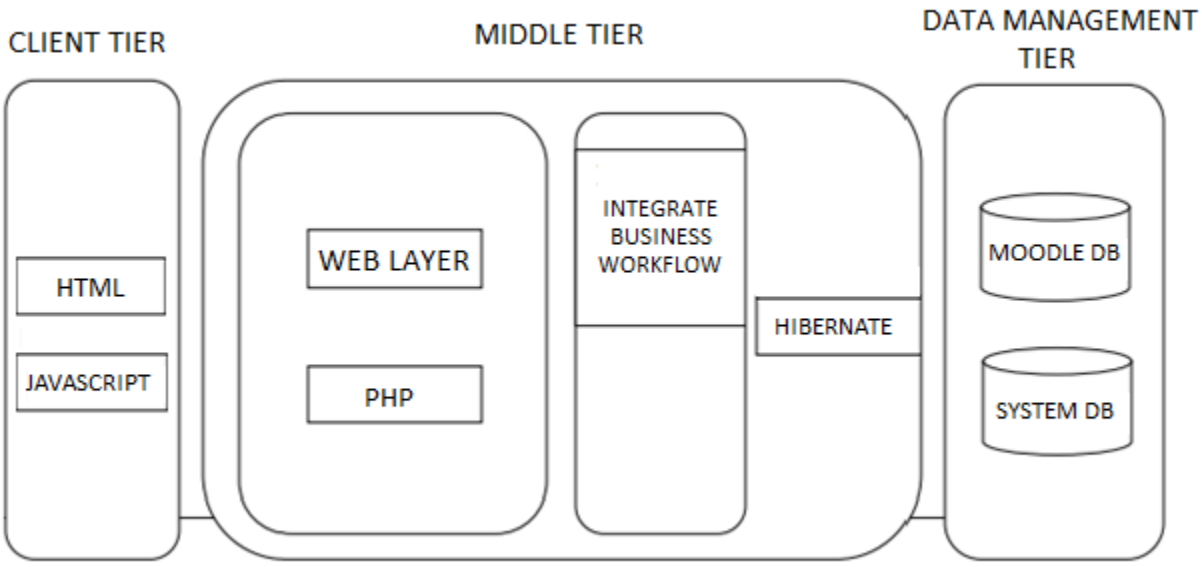
Since it inherits the principle of simple design, we have privilege to construct software with clear concept and implement them easily. It provides facilities for software extendibility and code reuse in future. We can manage to fit much further expanded functionality in future expansion of the system. The following bullet points notes some benefits of this framework.

- Easy to use.
- Highly flexible for developer.
- It has responsive grids.
- It has comprehensive list of components.
- It leverages JavaScript libraries.
- Frequent Updates can be found easily.
- Detail documentation and vast community so easy to explore in this new programming paradigm.

SYSTEM DESIGN OVERVIEW

This application is developed in three tiered architecture. It has front end tier or client tier for communicating with clients, middle tier for internal business/data processes and back end tier or data management tier to store/retrieve data from database.

HIGH LEVEL DESIGN MODEL



In this diagram, client can only see the front end tier. In middle tier, one module will handle webpage contents and style since everything is webpage should directly come from database. And the other module will integrate BPM (business process workflow). Data management tier looks for data storage and data retrieval to the middle tier.

This software has to perform certain tasks in repetitive way in order to give clients requests in successful manner. This can be enlisted in the user story board for more precise understanding.

In this system, we can visualize that client tier is functioning as a display screen and back-end tier as store room to store data in manageable manner. So, whole lots of functions are done in middle tier which consist of many different layers in reality. This tier will be discussed in more detail below.

Middle tier should perform the following functionalities in general.

- Response to the request from client with appropriate result taken from database.
- Response to admin and allow admin to change some contents stored in system database.
- Manage data flows from users to admin and vice versa.
- Handle some exceptions while the system is running.

To handle all these tasks, middle tier should perform all the users (student, admin, and guest) requests in more specific manner. This can be stated as user stories as shown in following table.

Iteration	User stories
1	Visit website as guest users can see all the contents of site.
2	Registered users can login, see their courses in Moodle, pay fee, view marks/results etc.
3	Admin can login edit course contents, view students' details, add/delete students/tutors, activate/deactivate accounts etc.
4	Tutor can see course contents, marks the assignments/exam papers etc.
5	New users can create account and pay fee to be registered students.

LOW LEVEL DESIGN MODEL

The below diagram depicts low level design model which is detail operation and communication from the application layer to the server side presentation layer. This is actually an overall flow of the application from server to client, server to database and database to client in browser. It is further explained with supportive diagrams like context diagram and class diagram, sequence diagram, use case diagram etc.

USE CASE DESCRIPTIONS

Registration: Actor provides all his details and provides a verified user id to login and a password with specified characters and letters. Actor then gets the authentication for the login to the site and use the web for the accessing.

Login: Student and Admin as an actor provides the id and the password to get logged in. The login process also includes the verification of the id and password. Only after the login student can buy the courses whichever needed.

Browse Courses: Student here in this can browse to the available courses present in the web and have glance on the course and then move on to the further stage.

Book Course: In this actor after accessing/browsing the course they can next book the course. Each course has its own availability and the specified price to that particular course where actor can choose the availability.

Add/Delete Course: In this the admin has the official authority in the adding or deleting of the courses in the web according to the flexibility.

Add/Delete Article: In this same as of the course admin also have the same way of adding or deleting of the articles present in the web according to the need of the article on certain course.

Payment: In this after selecting the course and the moving on to the payment. Payment is processed through the PayPal providing the correct details and then admin confirms the payment to provide the further access.

Timetable: Admin provides the timetable to the courses and then student can watch the timetable and even with the change of the timetable from admin.

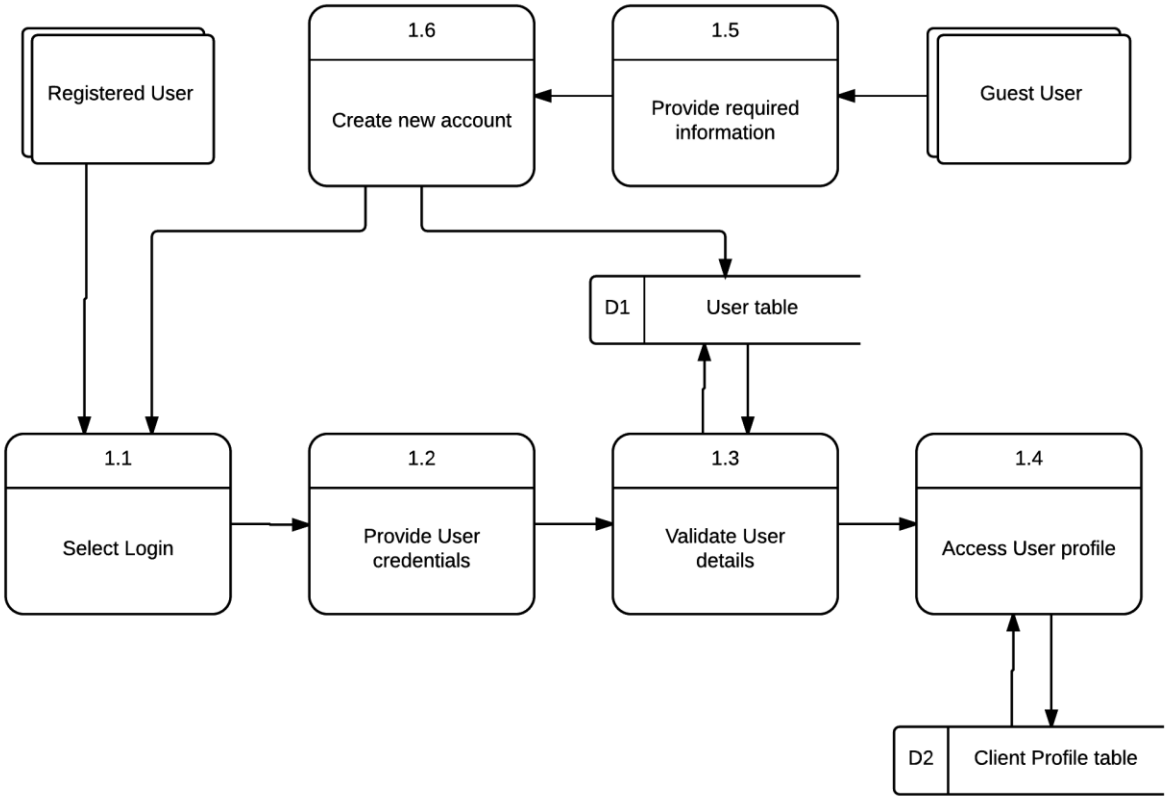
Moodle: After the payment admin provides the access to the Moodle with an id and password to access Moodle. With that password and id one can login to the Moodle.

Moodle courses: After the login to Moodle student can access the available courses in the Moodle. Admin also can add some courses and also delete some.

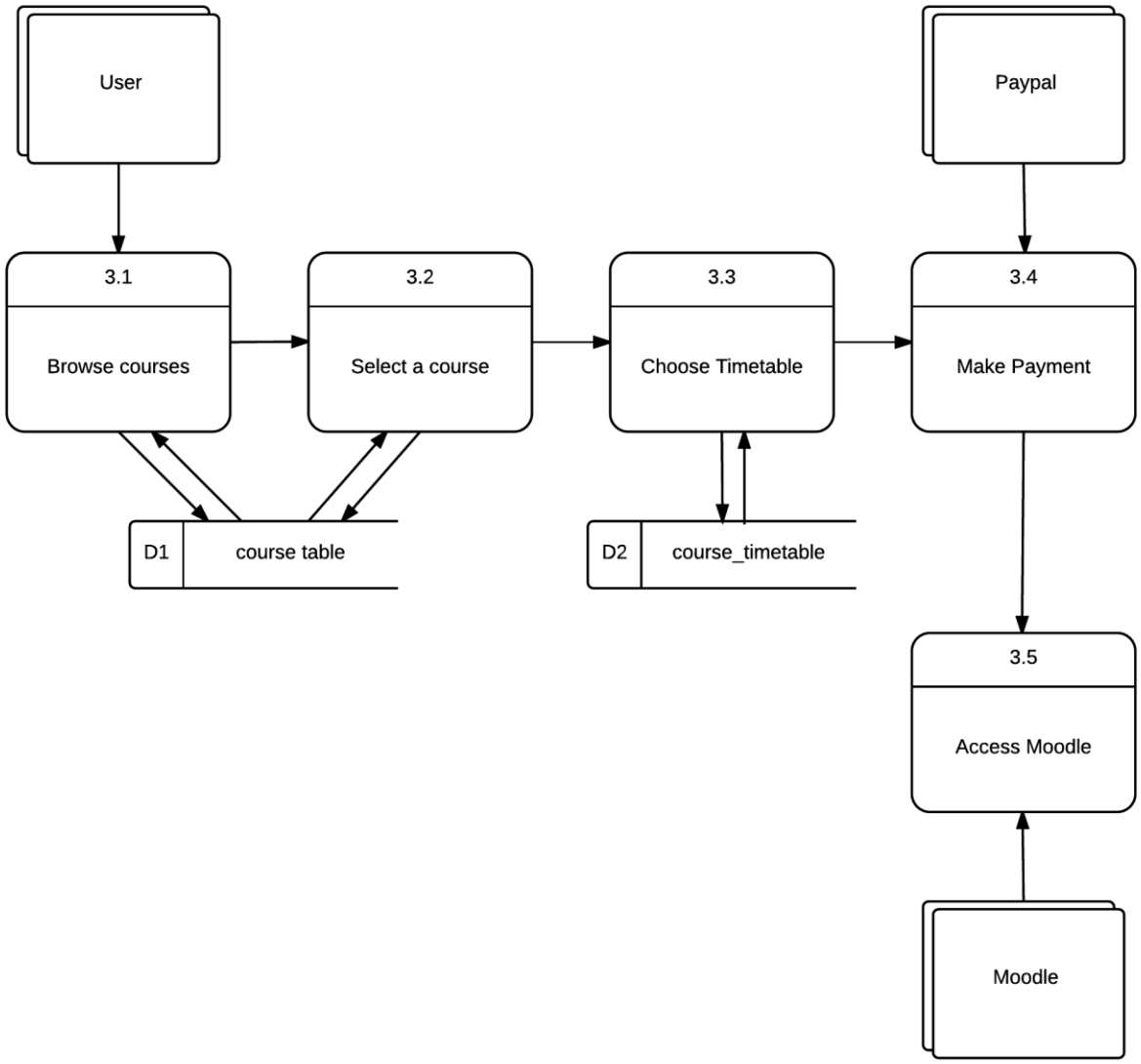
Mails: Student can send mails to the admin in the verification of the courses and any queries and receive mails from the admin regarding several aspects of the courses. Admin can send mails to the students regarding any changes and announcements.

DATA FLOW DIAGRAMS

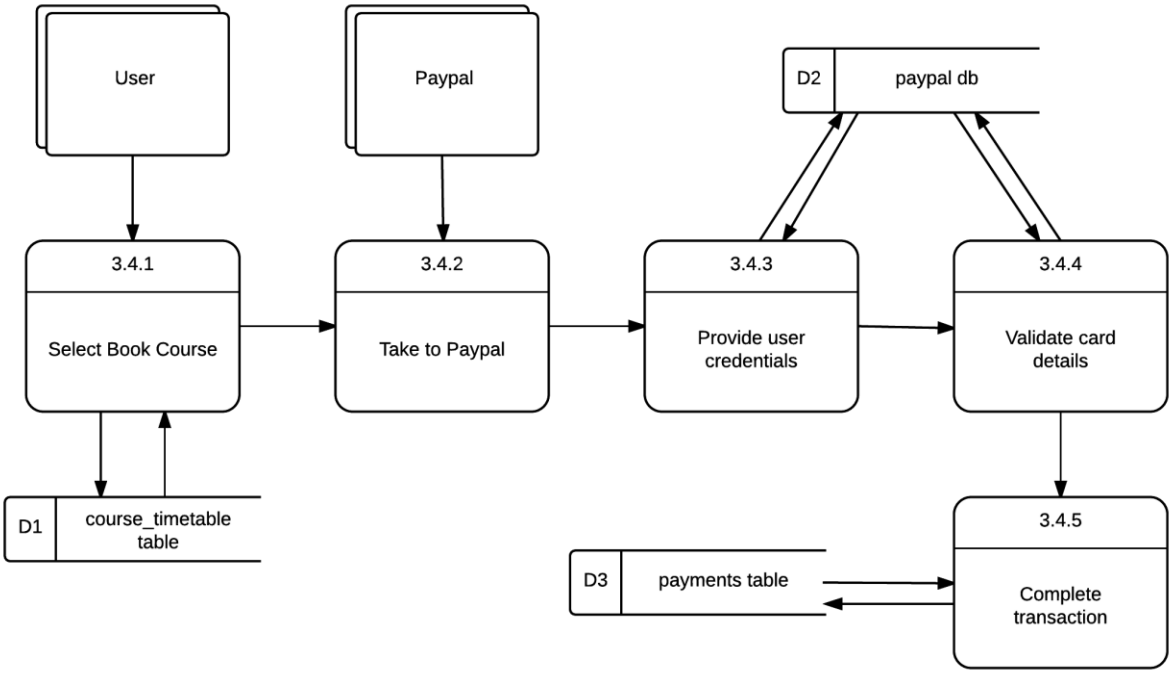
DFD LEVEL 1 (LOGIN PROCESS)



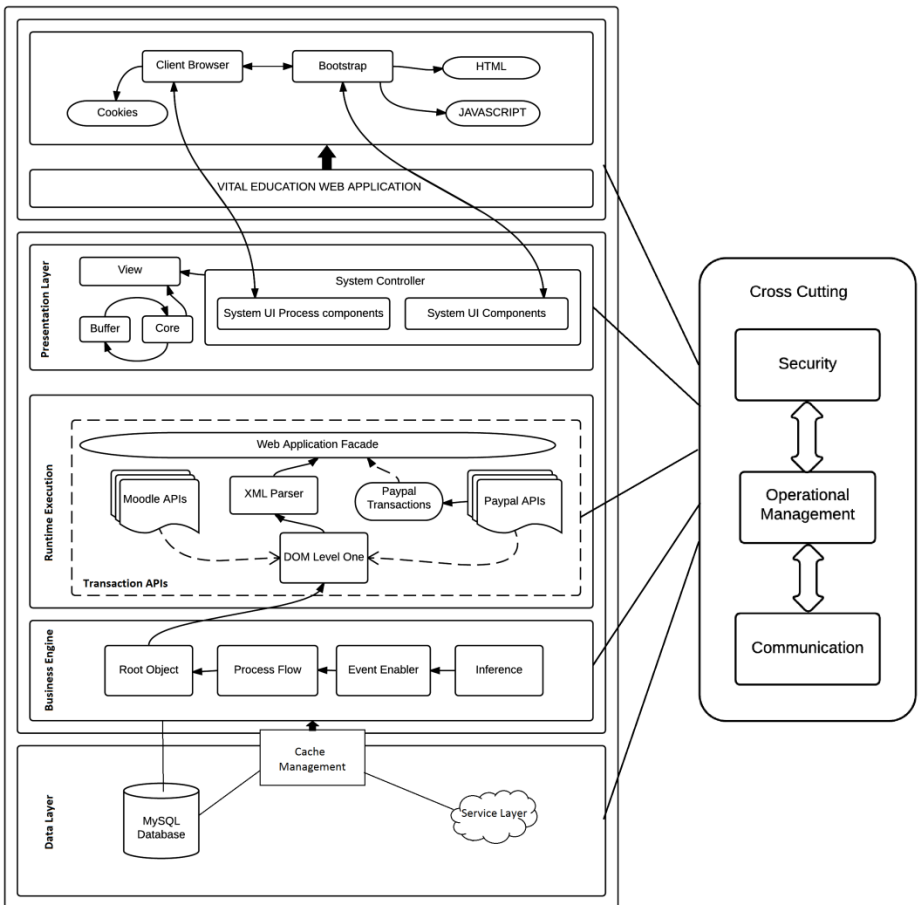
DFD LEVEL 1 (BOOK COURSE)



DFD LEVEL 2 (PAYMENTS)



ARCHITECTURAL MODEL VIEW

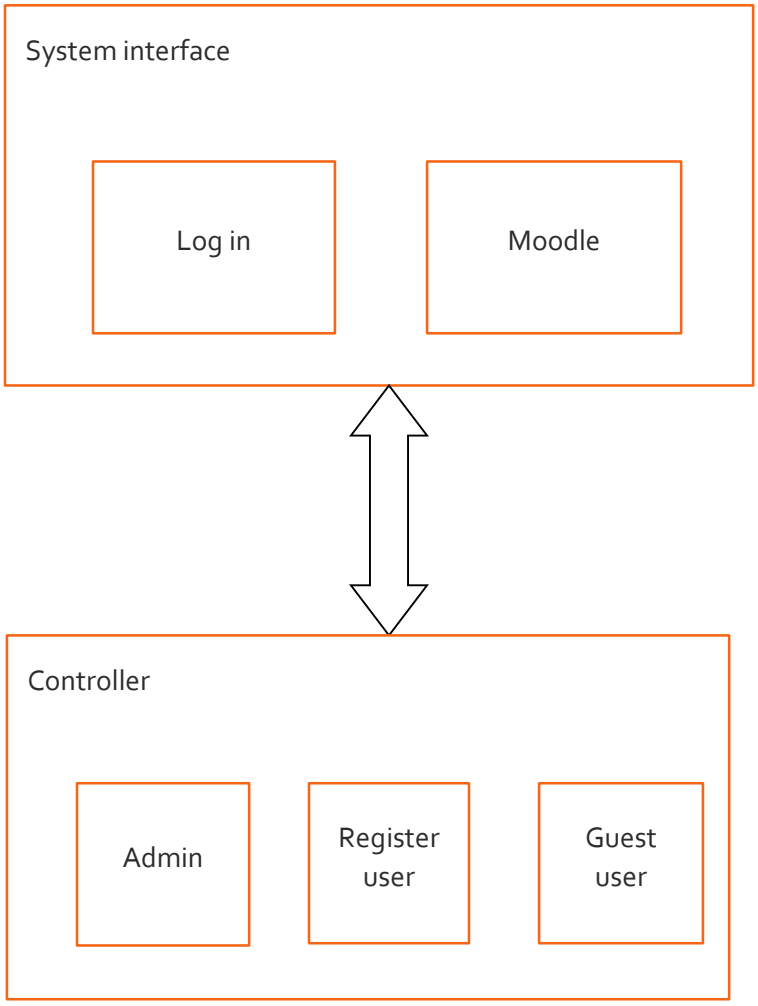


ARCHITECTURAL DESIGN OVERVIEW

Architectural styles and element: The proceeding part will deal with system architecture and style and elements.

Components, connector, and interfaces:

There are components which are interlinked to the software in different phase. The components will be providing support to other components. For example login components will be helping to check valid users to the system. Similarly, PayPal is for payment system. It will help to pay the fee.



Connectors:

For system component connection there are different modules like PayPal system, Moodle system. These components will connect to the system to do specific tasks for the system. Moodle is for course contents and PayPal is for paying course fee.

LIMITATION IN DESIGN:

This system is no the perfect version of the application. We can further improve this system in many aspects to make it appropriate for client. Some of the limitations are given below.

- The web application should be light as much as possible; if this application contains heavy contents then it will be slower while operating. This can make the users irritated by slow uploading or opening of the application.
- Similarly, this system need to be handled by admin with proper care, if he/she delete anything by mistake then the component that is deleted will not be retrieved again because he/she delete the content from the system database directly.
- The application is vulnerable to different kind of malware attacks like Trojan horse, viruses so to protect all the data from these malware, the system should be protected from malware attacks.

STATIC/DYNAMIC ASPECTS:

- This web site is dynamic website; it can interact with user by providing information that user request. It is supported by backend that is database and server lookup for data to the system database.
- Effective GUI to the website gives the look and feel of the site. It is designed in appropriate format what can be operational to all modern web browsers.
- It accepts various file formats as an input to the site, like text, pictures, videos and also give similar output to the users.

FUNCTIONAL AND NONFUNCTIONAL ASPECTS:

This website should support the following functional aspects during its operations:

GUEST USER

Register to the web system: The user can register to the Vital Education system by providing the required information like username, email, password etc.

Access Blog articles: Any guest user can access the blog and articles included in it.

Post Comment in Blog article: To post a comment, like or share the articles, the user should log in to the system

Make an enquiry: If the guest needs any additional information about the course, they can make an enquiry by sending an email to the corresponding staff.

Get the location map: An embedded google map should be provided to check the location and directions to the locations where the courses are being held.

ENROLLED USER

An enrolled user should be able to perform all the tasks that a guest user can do as well some additional functions which were listed below.

Login to the web system: The registered user can log in into the system by providing the valid username and password.

Reset Password: A forgot password option should be included in the login page to rest the password and the email will be sent to the address given at the time of registration.

Browse the webpage: Any user (Guest or Enrolled) can access the webpage and navigate through the system using the provided user interface.

Book a course: User can book a course by choosing one from the list of available courses and selecting the time table. To book a course, the user should be logged in to the system. If not, the system will redirect to the login page.

Choose Availability: Once the course is selected, the user can choose their preferable timing from the list of available timings.

Make PayPal Transaction: After choosing the course and availability, system should redirect the user to PayPal website with parameters like user id, course id and course price.

Auto generate Moodle account: The system should generate a Moodle account for the user as soon as the payment is successful.

Send account verification mail: The user should receive an email to verify the account and reset the Moodle password.

Access Moodle courses: The user should be able to access only the course that was enrolled in the Vital Education web system.

Send and receive emails: All the users can send as well as receive emails after logging in to the Moodle system.

Upload assignments: Moodle users can upload various file formats into Moodle system for their assignments.

ADMINISTRATOR

As per the client requirements, an admin panel is to be designed through which the admin can add, remove, change or update almost the whole GUI of the system as well as the system database which will reflect the main system with which the users interact.

Login to Admin Panel: Admin can login to the admin panel using the admin credentials. As per the client specifications, only one admin account will be created and anyone in the management panel will use the same credentials to login to the system.

Add/Delete courses: Admin can add and delete the courses into the database through admin panel. Adding a course involves providing all the details of the course in the given fields.

Update courses: Admin can modify the available courses according to the management requirements. A frequent updating of time table is required since the availability of the trainer changes.

Activate/Deactivate courses: Courses can be either activated or deactivated by changing the status of the course which will reflect the changes in the live system. A deactivated course will not be visible in the live system.

Reset Password: If a user cannot reset the password for any reason, the admin should be able to access the user table and reset the password for the user.

GUI modification: Admin will be provided with a GUI that is similar to the live system where he/she can change the web elements like course description, pictures etc.

Delete/Block users: Admin should be able to block or delete a user from the database.

Send and receive emails: Admin can send and receive emails through the admin panel.

Assign tutor: Assigning tutors to the course can be done in the course updating page where the admin can get the list of tutors from which he/she can choose one.

Change user access permission: User access in the Moodle depends upon the level of permissions they have. Admin can change the access level of the users in the Moodle admin panel.

DATA DICTIONARY

Entity Name	Entity Description	Column Name	Column Description	Data Type	Length	Primary Key	Nullable	Unique
Registered students	Students Who registered	id	Unique	int(11)	11.-1	true	false	false
		name	Upcoming Event Name	varchar	150	false	false	false
		address	Name of the upcoming event	varchar	800	false	false	false
		User id	User id	Int(11)	200	false	false	false
		User	Encrypted	varchar	255	false	false	false

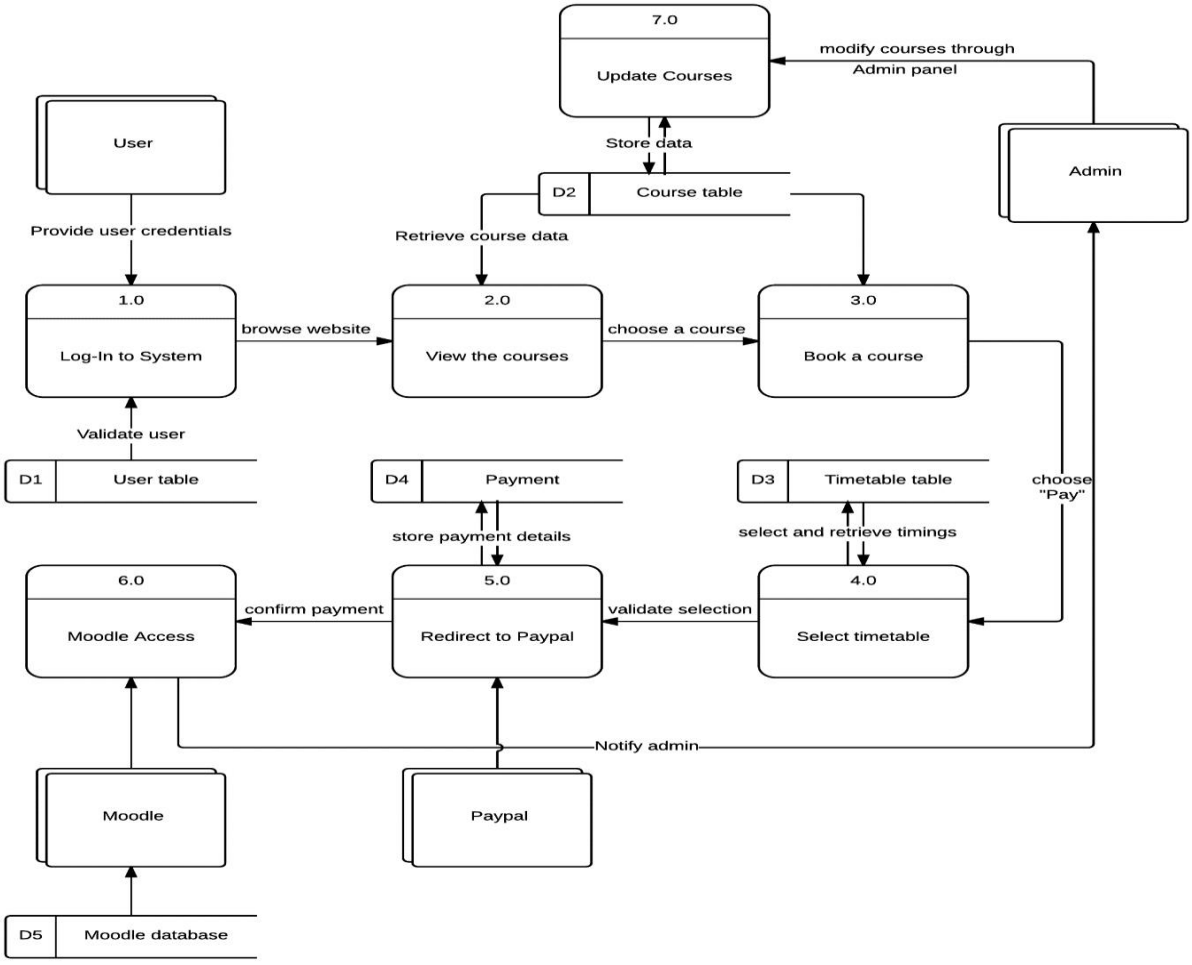
		password	password					
		DOB	Event Date	date	15	false	false	false
		Course id	Enrolled course	Int(11)	10	false	false	false
Admin	Administrator of vital education	id		int(11)	11.-1	true	false	false
		name	type name	varchar	150	false	false	false
		password	Encrypted password	varchar	100	false	false	false
		Users	User type	varchar	150	false	false	false
Guest user	Guest users	name		varchar	120	true	false	false
		address	User's address	varchar	200	false	false	false
		email	User's email id	varchar	110	false	false	false
		Phone no	Contact number	Int(11)	120	false	false	false
User id new		New id	Id	Int(11)	120	false	false	false

FILE FORMATS

The file formats we have used in this project are listed in the table below.

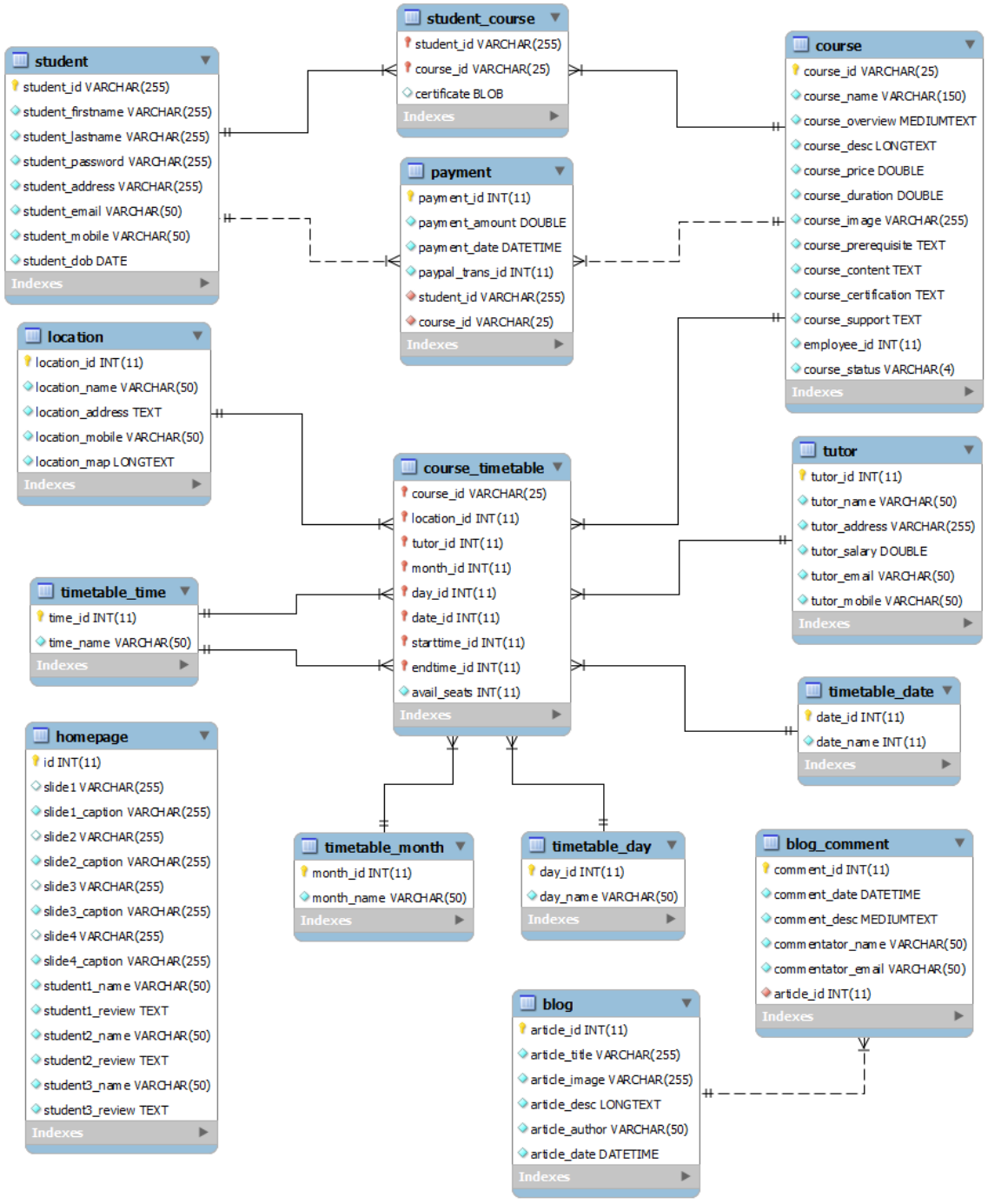
File type	Description
.JPG	Images used in projects
.php	Php files are used to run server side scripts
.css	This is used in style sheets of front end site
.sql	Storing data to our database

DATA FLOW DIAGRAM LEVEL 0



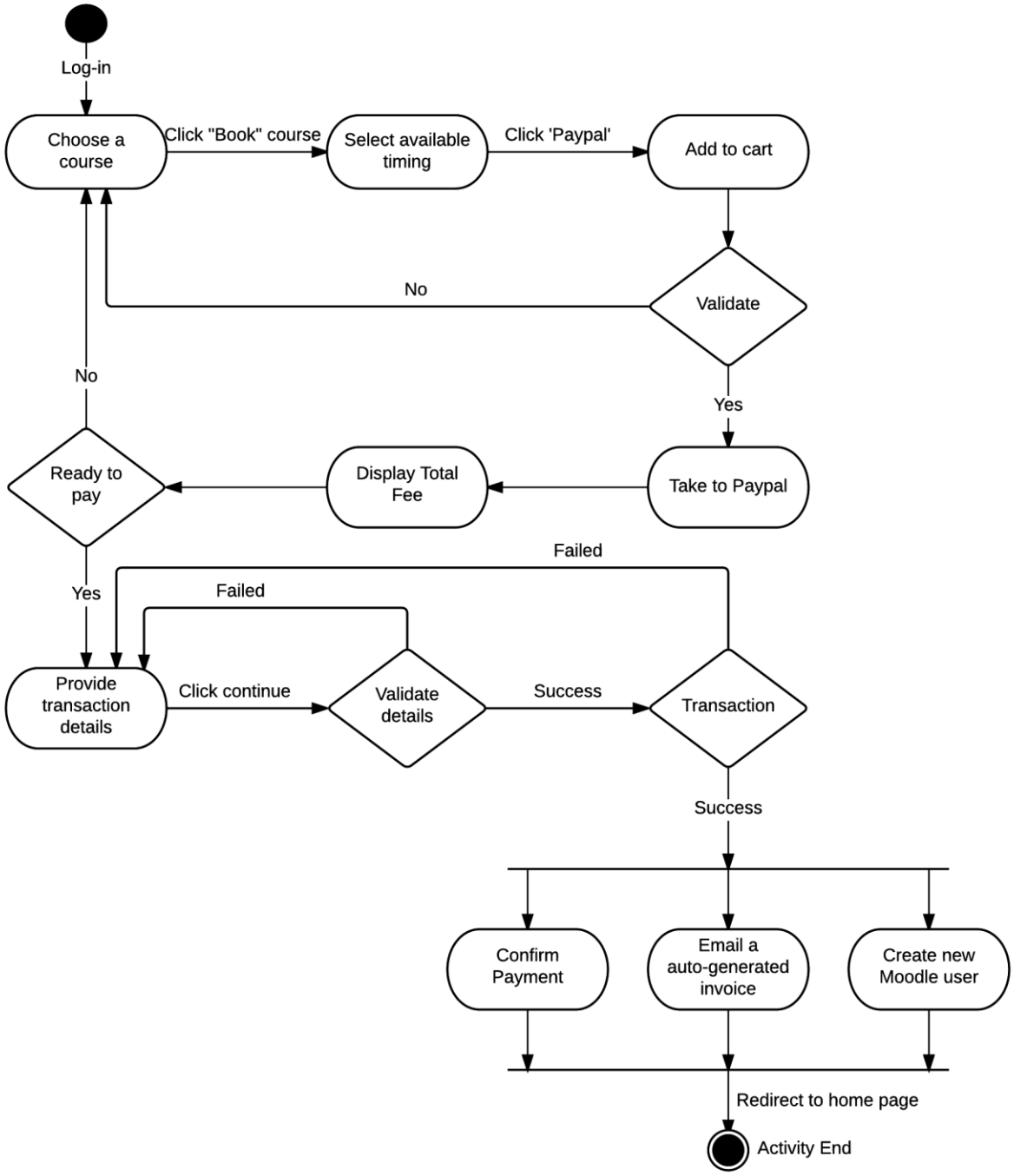
Data flow diagram is used to show the process that is happening in the system. It is used to make a trace of flow in the system. There are 6 processes in this diagram which are show in the above diagram. It is used to show when and where the data can be stored. It is very useful for making note of our data being kept in the system. It is easy to understand.

ER DIAGRAM



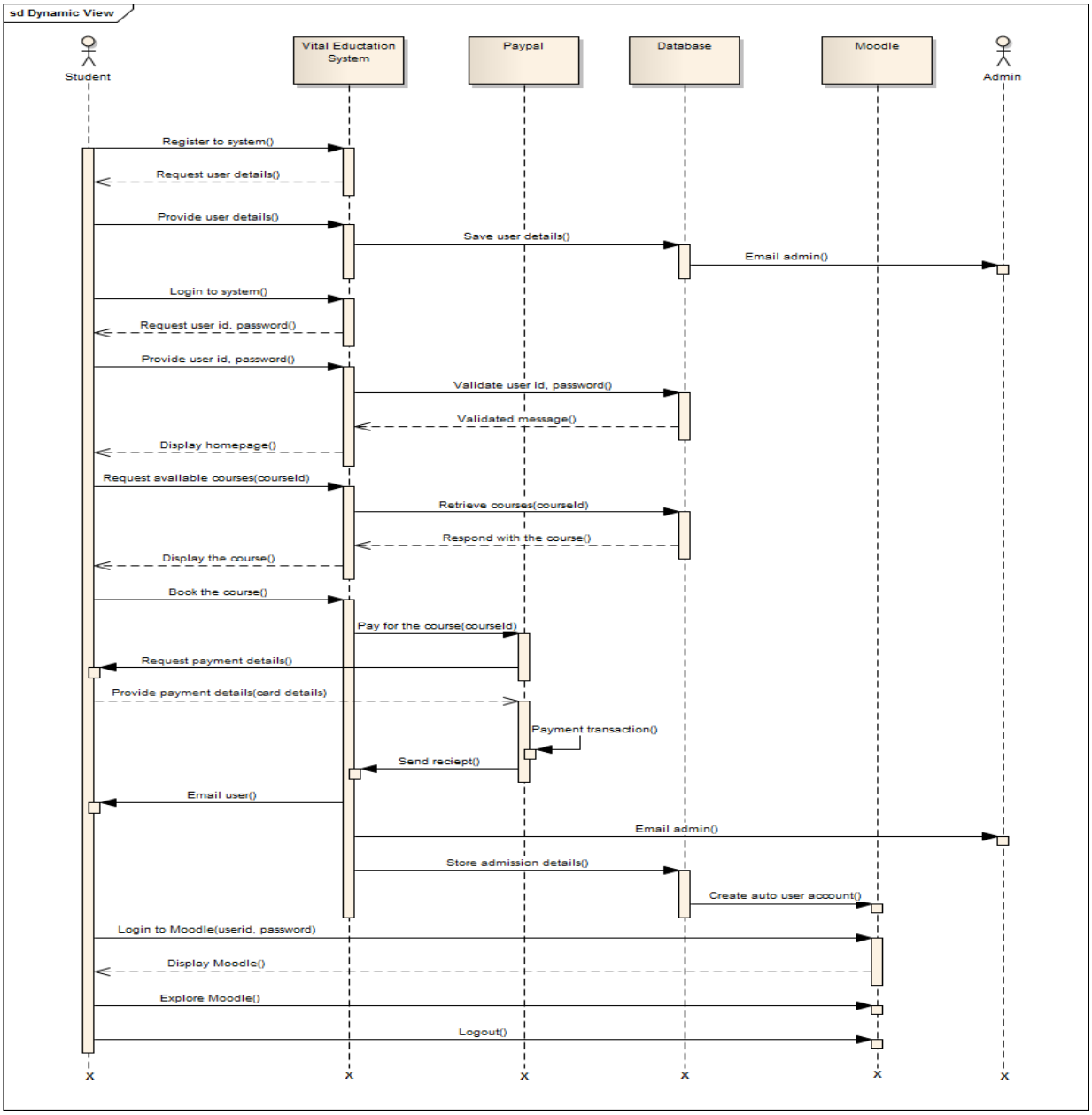
The above given diagram is our ER diagram. This diagram is basically used to create our database. This diagram shows all the table used in our database. It is very useful to map real database. There are eleven tables in the database which is shown in the ER diagram above. These tables are the entities where information of the system will be kept for future usability. It is retrieved using sql queries.

ACTIVITY DIAGRAM



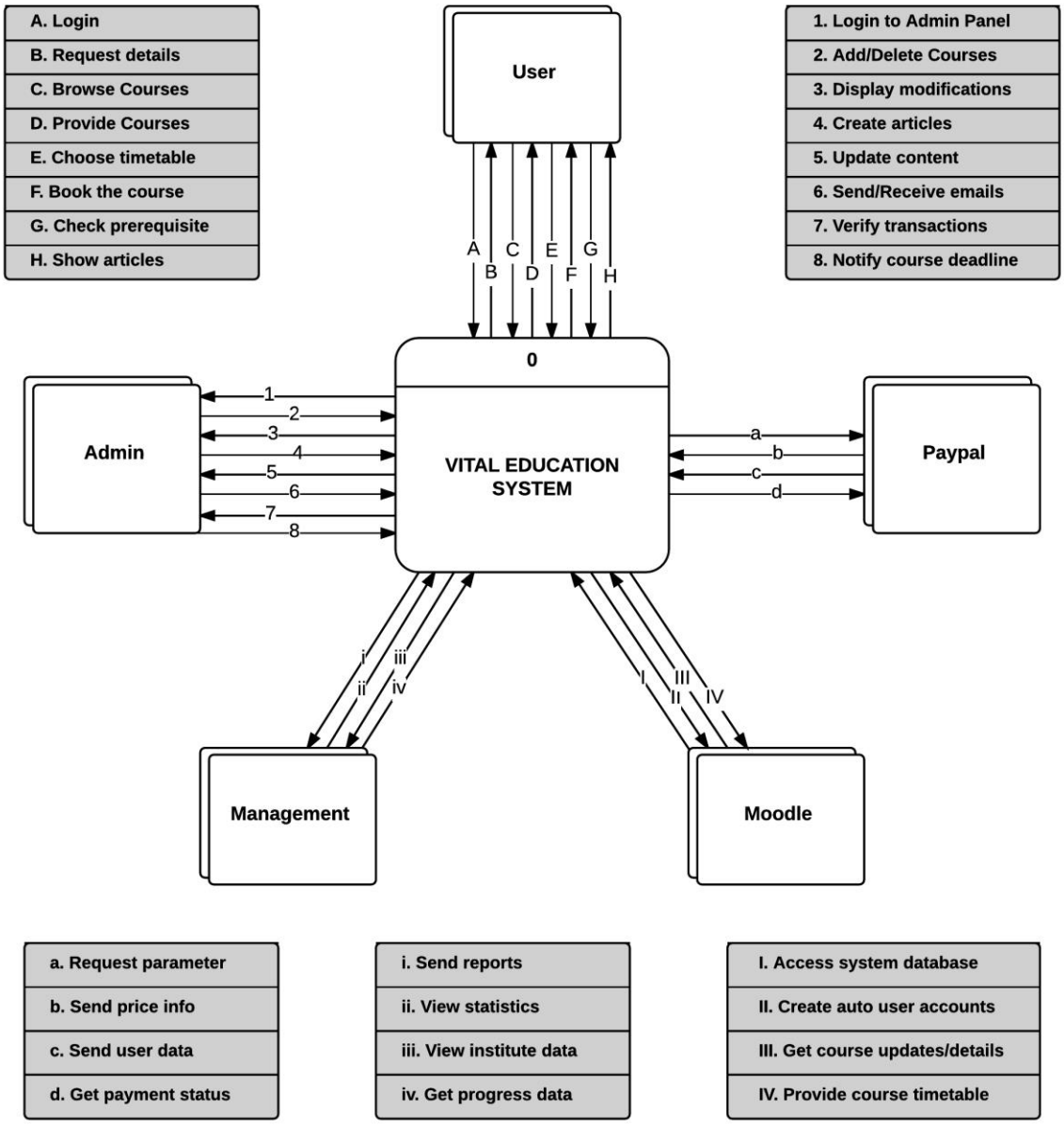
This diagram shows the events that are happening in the system step by step basis. This diagram is used to decide what the important events that are happening in the system flow.

SEQUENCE DIAGRAM



This is sequence diagram for our system. It will show the sequences of communication between actors and the sub system step by step. It is also a representation of sequences of communication happening in the system. It is very useful to know what kind of communication is happening.

DETAILED DECOMPOSITION OF THE SYSTEM



We have five external agents where the main system interacts. The five agents it Interacts are User, Admin, Moodle,Paypal,Management.

Admin has the authority of controlling whole system such as adding, deleting or updating the courses, posting articles in the blog, managing the UI, changing the webpage content etc. Admin can also provide the deadlines for the course selections. Admin in the article section can add or delete or even modifying the blog at any instant of time. Admin can send the e-mails to the users regarding any information and receive the mails from their users regarding any queries.

Management recieves the updates weekly on the whole system and goes through the transactions between the user and the admin on payment,etc.Management manages the reports of the whole system and also have an overview of the statistics day by day. Management views the institute data of the number of students that are enrolled, payment data. It gets the progress data of the institution.

Users play the most important role as the external agents of the system. They have access of booking the courses, accessing the moodle, choosing the timetable, navigate through the webpages, etc. Users first needs to get registered for the site and get and id with password and using that can access the login page. Unless he gets registered he cannot buy a course. Selecting the course after logged in leads to the paypal page. Users also can view the articles in the site and also can add the comments to the articles. Users in the case of booking the courses they even can check the pre-requisit required for the course.

Paypal helps the payment process, once the course gets selected and is to be booked it leads to the paypal page. Next to the selection of the course the user will be directed to the paypal page to continue with the transaction providing his card details or login with the paypal account, in this stage they also can create a new paypal account. On the successful payment, user gets the confirmation mail and gets a moodle account created for the new user. Through this moodle account user can access the institution moodle.

Moodle system databse is there connected to the vital education database through which the medium users will access the courses that are present in the Moodle. Users in the moodle can there view the courses which they have been enrolled and the content of the course. In this moodle they also can view the data or any material related to that course. In this moodle admin can make the changes with the courses such activation or deactivation, adding or deleting the courses. Admin can the schedules for the courses in this moodle.

CODE-SPECIFIC DESIGN SPECIFICATIONS

The following pseudo code explains how the system is being implemented and all the major SQL queries constructed to retrieve or store the data in the database.

To start with the database implementation, we created a course_timetable table which was connected to most of the tables to get the timetable. This table includes course_id, tutor_id, location_id, date, start_time, end_time, and available_seats. To retrieve the data from the database, the query that was constructed is

```
SELECT course_name,
location_name,tutor_name,month_name,day_name,date_name,st.time_name AS
'starttime',et.time_name AS 'endtime',avail_seats FROM course_timetable
JOIN course ON course.course_id = course_timetable.course_id AND course.course_id = '$id'
JOIN location ON location.location_id = course_timetable.location_id
JOIN tutor ON tutor.tutor_id = course_timetable.tutor_id
JOIN timetable_month ON timetable_month.month_id = course_timetable.month_id AND
timetable_month.month_id >= MONTH(NOW())
JOIN timetable_day ON timetable_day.day_id = course_timetable.day_id
JOIN timetable_date ON timetable_date.date_id = course_timetable.date_id
JOIN timetable_time as st ON st.time_id = course_timetable.starttime_id
JOIN timetable_time as et ON et.time_id = course_timetable.endtime_id
ORDER BY month_name DESC, date_name ASC";
```

The server-side language used in the project is Php. To access any information from the database, Php statements had been implemented inside the html pages where ever necessary. To login into the system, the Php code implemented is

```
<?php
error_reporting(0);
        require ('dbconn.php');
        include ('nav.php');
        if(!isset($_POST['submit'])) {
                Some HTML code to display the login page
} else {
        require ('dbconn.php');
        $userid = mysqli_real_escape_string($conn, $_POST['userid']);
        $password = mysqli_real_escape_string($conn, md5($_POST['password']));
        $query = "SELECT student_id,student_firstname,student_password FROM student
        WHERE student_id='$userid' AND student_password='$password'";
        $run_query = mysqli_query($conn, $query);
```

```
        $row = mysqli_num_rows($run_query);
        if($row == 1){
                $_SESSION['userSession']=$userid;
                echo "<script>alert('Log In successful')
                window.location.href='index.php'
                </script>";
        }
        else {
                echo "<script>alert('Invalid username or password. Try again!')
                window.location.href='login.php'
                </script>";
        }
}
?>
```

To keep the whole website more consistent, we used bootstrap framework which comes with some basic predefined styling. The design style used in the website is material design where the web components come up as cards and with some border and shadows. The implemented pseudo code is

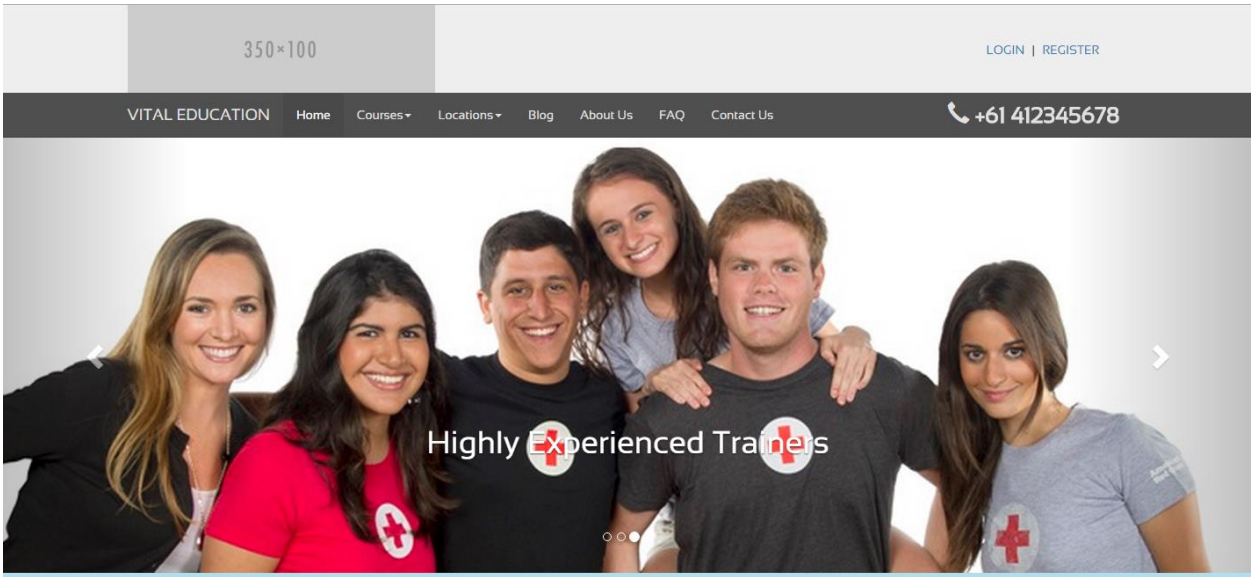
```
<div class = "panel panel-default">
        <div class = "panel-heading">
                //Some code that goes in heading
        </div>
        <div class = "panel-body">
                //Some code that goes to body
        </div>
        <div class = "panel-footer">
                //Some code that goes to footer
        </div>
</div>
```

GUI DESIGN

USER WEB SYSTEM

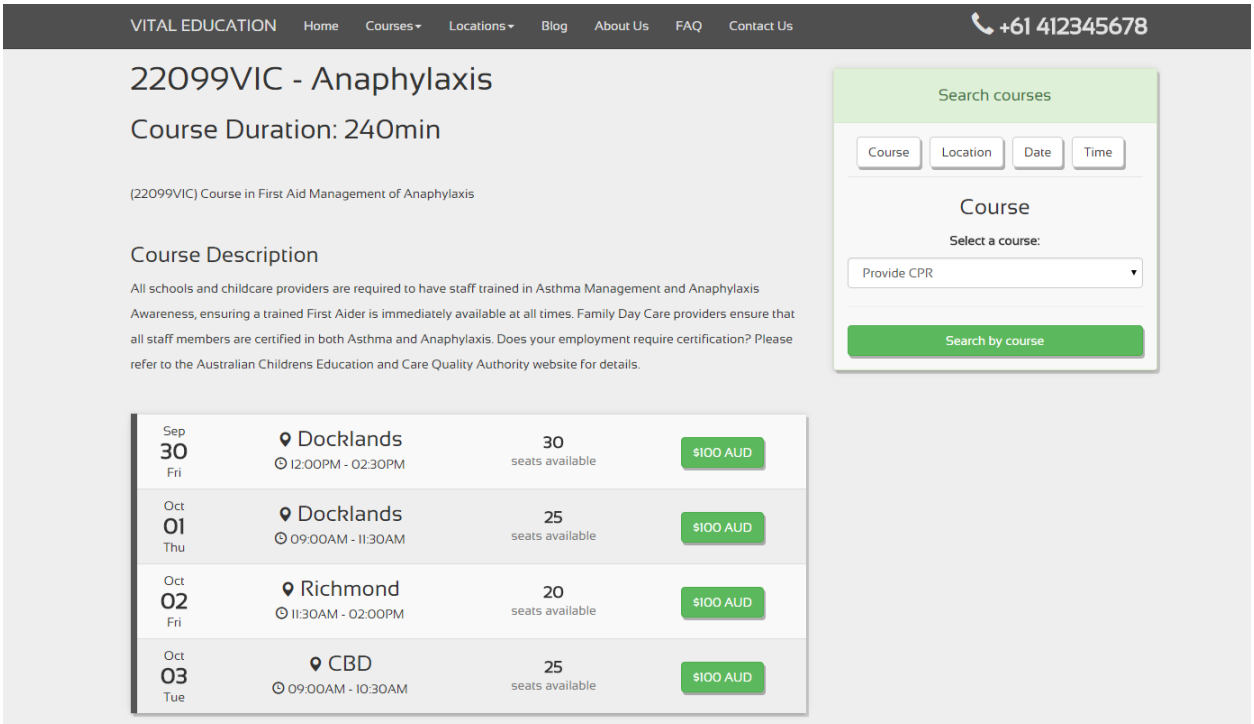
HOME SCREEN

Once the user open the webpage, the home page comes up with a carousel that includes some pictures that show the major updates or new in the institute. A navigation bar was implemented that sticks on to the screen helps the users to navigate through the pages. Under the carousel, some popular courses are listed which contains a book now button to directly book the course. Some information about the institute and the feedback about the institute will be available under the courses division. At the end of the page, some quick links are given. The users can login or register to the system using the links available at the top right corner of the page.



COURSE PAGE

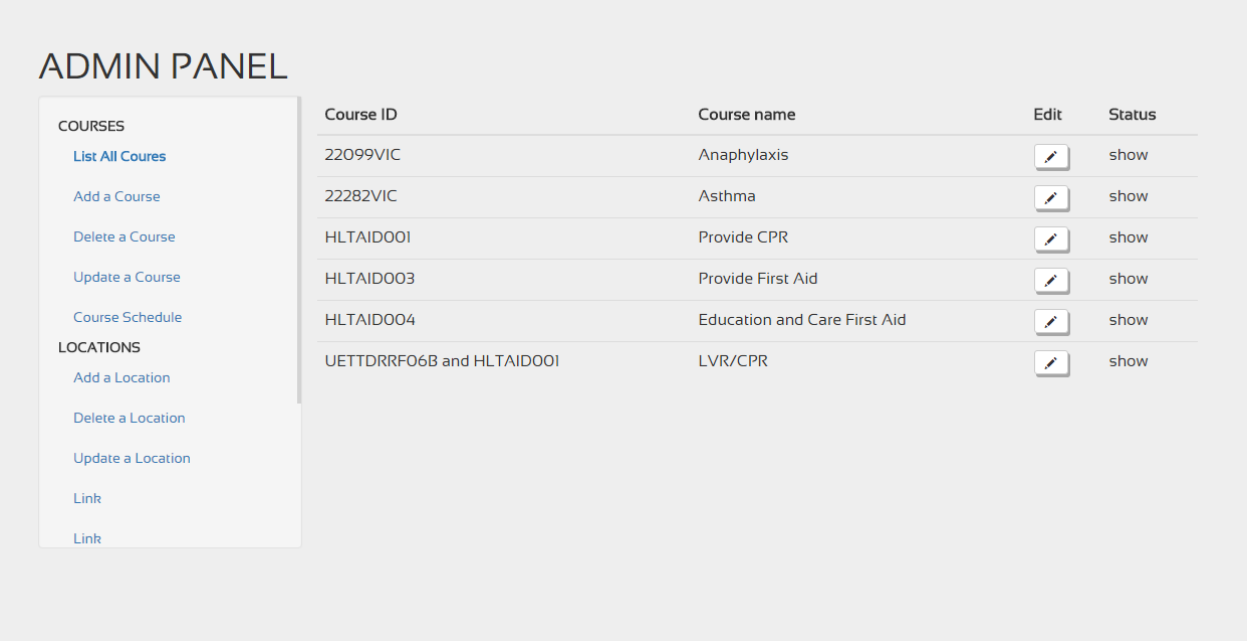
The user can access the courses through the navigation panel. Based on the user selection, the course page displays the content dynamically with the list of available timings. The course page contains the description and some overview of the course at the top. A list of available course timings will be displayed at the middle of the page from which the user can select the suitable timings if there are any seats left. A book now button is present for each schedule which checks whether the user is logged in or not and takes them to the PayPal page if they are logged in or to the login/registration page if no user session is found. At the bottom of the page, a tabbed pane is present to check the prerequisites, support and certification details. Apart from this, a search panel is implemented which is present at the top right corner which helps the users to search courses based on either course name or location or date or time. Once the user selects a particular option and hits the search button, it gives the results based on their search.



ADMIN SYSTEM

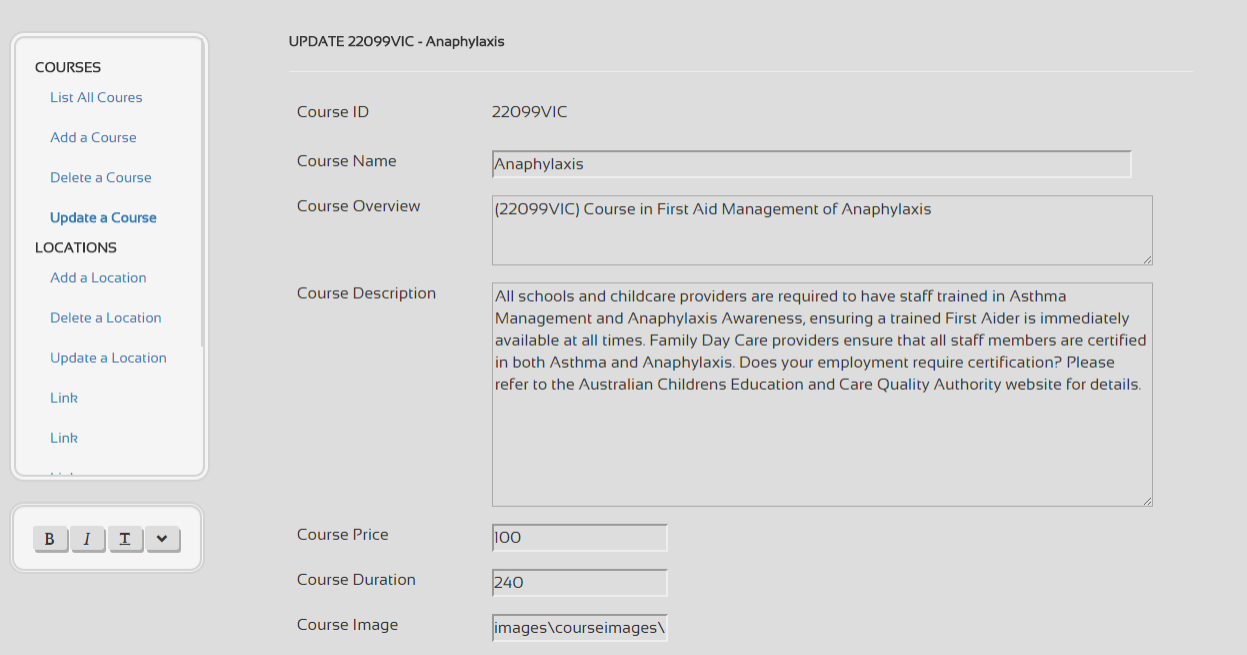
COURSE LIST

The admin has a separate system to manage the courses and web interface. Once the admin logs in to the admin panel, a list of courses will be displayed with an option to edit and the status of the course will be shown (either show or hide). The left hand side consists of a sidebar which acts as a navigation bar throughout the admin panel. Once the admin chooses a course to edit, it displays a page with all the course content which the user can edit. An admin session is required and maintained as long as the admin navigates or make changes to the data.



UPDATE COURSE

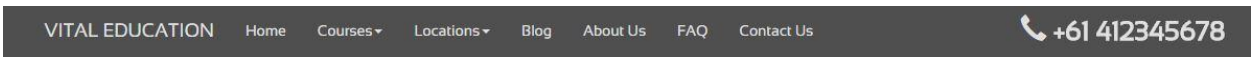
Admin can update already available courses through the admin panel. The update course page consists of all the content that is being displayed in the main system. The admin can modify the details by hitting the edit button at the top right corner. In order to add bold or italic text, toggle buttons are implemented which toggles the bold, italic, underlined or normal text. Once the admin updates the necessary fields, there is a button to preview the page before saving the information into the database. If all the changes are appropriate, the admin can save the data into the database by hitting the save button.



DESIGN SPECIFICS

We had chosen more flat and simple colors in the project which makes the website simple and more interactive. The whole website uses 'Sansation' web safe font which makes the text more clear and accessible. The buttons uses bootstrap theme with some custom cascading style sheet gives them a three dimensional effect and clickable experience to the user.

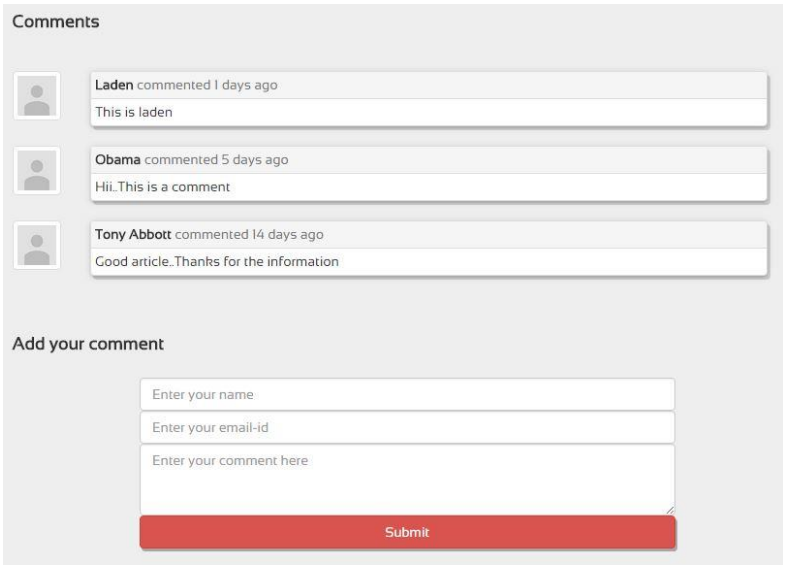
Navigation bar



Buttons

Panels

Comments



REFERENCES

Wiegers, K., & Beatty, J. (2013). *Software requirements*. Pearson Education.

Chung, L., Nixon, B. A., Yu, E., & Mylopoulos, J. (2012). *Non-functional requirements in software engineering* (Vol. 5). Springer Science & Business Media.

Hofmann, H. F., & Lehner, F. (2001). Requirements engineering as a success factor in software projects. *IEEE software*, (4), 58-66.

Laitenberger, O., Atkinson, C., Schlich, M., & El Emam, K. (2000). An experimental comparison of reading techniques for defect detection in UML design documents. *Journal of Systems and Software*, *53*(2), 183-204.