System Requirements Specification for:
DBIS Course Project SRS

Sriram Bhargav,Varun Reddy,Deepankar Reddy,Hasan Reddy

December 1, 2011

Contents

Chapter 1

Introduction

1.1 Overall Description

Purpose of the project is to introduce new features into moodle. The project aims at introducing *Inline commenting* feature in Moodle forums and to embed *code highlighter* feature into moodle forums and to embed the ability of *making groups* for a particular project or assignment In brief, Inline commenting feature is to comment on a part of section of code or forum, so as to enhance communication between Professor/TA's and student with the feature of Professor being able to moderate the thread. Code highlighter feature is to increase visual appeal and to make code more readable. We will be able to create groups of student for particular project or assignment so that they all can view same submission page and will be able to get marks and crib through same thread we are creating.

1.2 Definitions

Moodle:- Moodle (abbreviated as Modular Object-Oriented Dynamic Learning Environment) is a free and open-source e-learning software platform, also known as a Course Management System.Blocks are items which may be added to the left, right or centre column (depending upon your theme) of any page in Moodle.

Any block can be made 'sticky' so that it appears in all the contexts below, such as throughout a course or a particular activity.There are many standard blocks like: Activities, Blog menu, Calender.

1.3 References

http://docs.moodle.org/dev

1.4 Overview of Developer′s Responsibilities

Developer is responsible for loading features and ensure it′s functioning and also training people how to get used to that feature and their facets.

Chapter 2

General Description

2.1 Product Perspective

It is built on existing moodle schema. To the schema we add new tables. Interface is plain dialog boxes with user′s name, displayed bottom of the question w.r.t to phrase highlighted on clicking the button show comments. No interface for code highlighter as it is embedded into forum.we will be giving interface for creating groups and thus making them to see same submission page.

2.2 Product Functions Overview

1. *Comment*:- After selecting a phrase on then clicking on comment will open up a dialog to comment on that phrase and the phrase will be highlighted.
2. *Show comments*:- To see all the comments on a highlighted phrase.
3. *Edit comment*:- To edit a previous comment on a phrase.
4. *Delete comment*:- To delete your comment on a phrase.
5. *Phrase Highlighter*:- To highlight the phrase on which commenting is done
6. *Notify*:- To notify users posts from other users.
7. *Code Highlighter on/off*:- Enabling/disabling the code highlighter feature while writing.
8. *Conclude*:- To finally shut or moderate or delete the comment thread and conclude on a thing.

2.3 User Characteristics

Particular comment thread is specific to the student, prof., TA′s, course, assignment. Any person registered for a course can use this feature. Students, TA′s and professors are to be trained for using this.

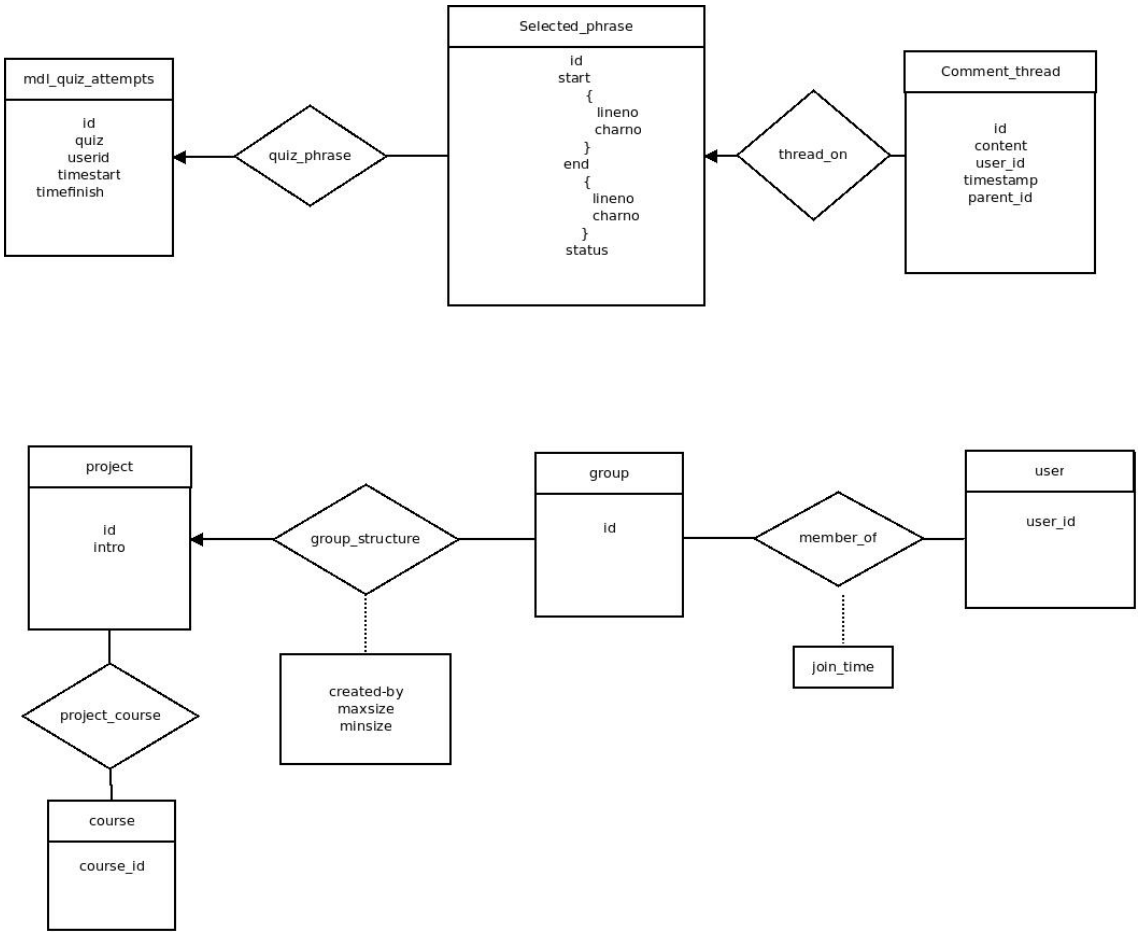
2.4 General Constraints

Main task underlying before us is to understand the Moodle open source project. Once we are done with that these features can be embedded in a couple of weeks.

Chapter 3

Information Description

3.1 Entities and Relationships



3.2 Data Dictionary

1. mdl_quiz_attempts

id	int	primary key	Quiz id - unique number quiz attempted
quiz	int	not null	Quiz number
userid	int	not null	User id of student
timestart	int	not null	Time quiz started
timefinish	int	not null	Time quiz finished

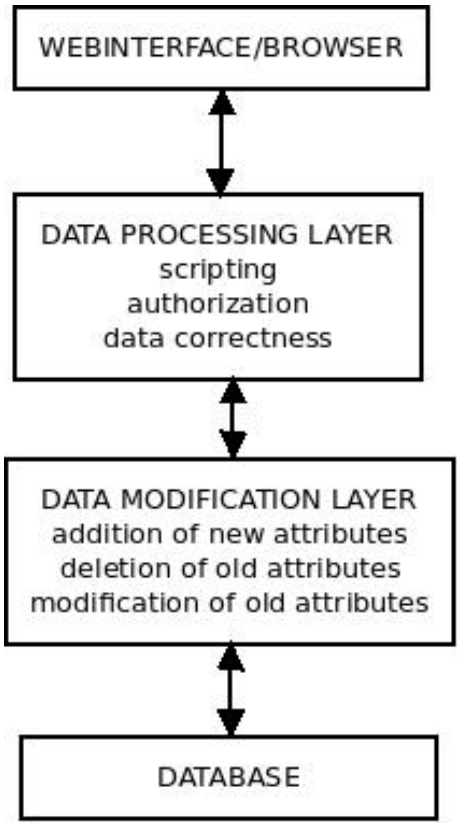
2. phrasecomments

id	int	primary key	unique id of phrase(high-lighted)
startnum	int	not null	Selection start char no
endnum	int	not null	Selection ending char no
contextid	int	not null	Context id is like pagenum
questionid_charno	int	not null	questionid for which the crib is made
attemptid	int	not null	Status of discussion(resolved or not)
phraseid	int	foreign key to mdl_quiz_attempts	
postcap	bool	not null	resolve or not resolve
instanceid	int	foreign key to mdl_block_instances	

3. phrase

id	int	primary key	
content	varchar(100)	not null	Content of the post
user_id	int	not null	It's the id person who made comment
timestamp	timestamp	not null	Time at which the post is made
phrase_id	int	not null and foriegn key to comment_thread	id of the comment to which it is the reply
contextid	int	not null	Context id is like pagenum

3.3 Data Flow



When you select a phrase to comment it, relation is formed between them. Only the first comment is connected to the phrase all the rest will stay as child i.e maintaining parent hierarchy. Phrase is inturn connected to assignment as we can comment on multiple phrases. For creating a project group, Course can have multiple projects and each project have multiple groups and each group have many numbers with max.size and min.size restrictions imposed. When a group of students form a project group, it is connected to project and which inturn is connected to course. The main data flow is as follows for inline commenting: the data flows from user for selecting the tag to the db(selected phrase) relation.Then after starting the comment we have the data flowing to db(comments relation).When the instructor is requesting the status of a thread then the data flows from the db to the computer.
for creating groups mainly the difference in the flow will be in the when the user requests for a submission page the interface will look it up in the db and thus generate the common ui belonging to all members of the specific.

Chapter 4

Functional Requirements

4.1 Functional Requirement of Inline Commenting

When the grading of assignement is done the feature of the Inline commenting gets activated.Ta/student will be able to tag the particular part/segment of the assignment and start commenting in comment box.Instructor will be able to moderate it and finally say whether it is resolved or not. Creating new comment-boxes and using them: It's done by adding a new type of blocks (called phrasecomments different from predefined blocks) in moodle.

4.2 Functional Requirement of Making Groups

This function will be giving ability to make a group of students for a particular project or assignment.Then this group of students will be seeing the same submission page.Same marking interface and give an option of giving the same marks for all the students.

4.3 Functional Requirement of Code Highlighter

This is just a visual function.Using a Api of mozilla we will be able to detect and the language and highlight and format the code using a rich text editor

Chapter 5

Implementation of functions

5.1 For Commenting

5.1.1 phrasecomments block

We have created a new block by name phrasecomments. It uses instance of phrasecomment class and creates a comment dialogue. It is initiated by a module.js file on clicking on "write comments" options. We allowed multiple instances of the block on the page since it can be called on different phrases of the same page.

5.1.2 blocksphrasecomments js file

This javascript handles writing comments on selected texts. On selecting option write comments, it adds a phrasecomments block on the page corresponding to that question's phrase. It will pass the arguments question id, context id, startnum and endnum. Startnum and endnum are starting and ending positions of the phrase in the question.

5.1.3 Class for Commenting

We have implemented a entirely new class for implementing the comments called PhraseComments.This class handles all the functions Starting from creating a comment corresponding to user ids,quizid and other details from the input to inserting them in the database and also outputting them in required format. It also furthers handles adding and creation of new comments to the comment block. It has two versions one NonJS and other JS. It uses JS file phrasecomment.js for it's smooth operation.
this class also has the private variables corresponding to db and will initialized whenever a new comment is made or when a new block of comments pertaining to a particular question are to made.

5.1.4 Implementing of resolve for instructor

In addition to save there will also be a resolve button which will disable the post capabilties of all the instructors and students for the particular block of comments and thus resolving the thread whenever it is necessary. This resolve button is visible only to instructor, thus restricting students from resolving it.

Chapter 6

External Interface Requirements

6.1 User Interfaces

Web is the user interface

6.2 Hardware Interfaces

No special Hardware is needed

6.3 Software Interfaces

Supported by all browsers.

Chapter 7

Performance Requirements

The code will be used by all the users in a institution .But at a time no of users using it will be in few hundreds for the task of inline commenting the code should not put more load on the server used by institute and also the response will not be that a matter as we will be creating a attribute about the comment in the existing table the response time will be atmost 1 sec or so.As said above the no of transactions will be the number of users that will be atmost few hundreds. Database may grow big if all the threads are to remembered but if they are deprecated then it will be again in few hundreds.

Chapter 8

Design Constraints

8.1 Standards Compliance

not applicable

8.2 Hardware Limitations

as the databases are here small enough here this does not matter in our case.

Chapter 9

Validation Criteria

We will enrole ourselves one as student, other as TA and another as professor and we will create a sample course and a assignment in it. Student takes the exam, code highlighter can be checked parallely and later TA and professor will comment on on a part of assignment and this discussion goes on for some time and then they finally conclude.

We will also test by creating 10-15 threads and trying to manipulate them using the above said functions

Chapter 10

Other Requirements

None.But we are hoping that Moodle Api will be able to provide the possiblity of doing groups specific

Chapter 11

Future Work

if we are able to understand the schema and Api of moodle quickly then we are going to implement a few more of our ideas

1. linking of btp project course of students with git inorder to use the features of svn completely
2. Implementing groups in the assignment, if two students officially form a group they share same assignment page
3. by making forums for the students only, TA's only
4. reply to the discussion for directly through gpo account mail itself
5. Give a better interface of moodle calendar, option to make it public, adding course class timings also to it. additional classes (amail will be circulated if any event is added by others to your calendar)