

Requirements Specification

IU Physics Gradebook System

Yu Feng¹

Rohit Chandran²

Naga Rekha Malae³

Rev 418, November 5, 2008

¹Physics Department, Indiana University, Bloomington IN 47401

²Department of Computer Science, Indiana University, Bloomington IN 47401

³Department of Computer Science, Indiana University, Bloomington IN 47401

Executive Summary

Rev 721, December 3, 2008.

The Physics Department at Indiana University is a non-profit education institution. It provides quality education in physics to the students on campus. The department also offers undergraduate and graduate level programs in the physics major.

The proposed system will be used by the Physics Department as a means of entering and storing grades based on the organizational structure. After studying the client environment, the proposed system's characteristics, the possible technical issues in the development of the system and the possible alternatives in the feasibility study, the team found that such a system can be successfully built.

The client currently uses OnCourse to achieve this task of managing grades for students. However, due to complexities in the organizational structure and shortcomings of OnCourse, the current system does not achieve this task very efficiently. It has, therefore, become necessary that a suitable system be developed to meet the needs of the client.

The team studied the client background, emphasizing on organizational structure and the problems that the client faces with the current system. Also, an understanding of daily activities of the client has been taken into account. At each level of the organizational structure, the problems are made evident.

Next, the team analyzed specific goals of the system in terms of the needs of the client with respect to setting up the organizational structure and assigning grades. Also, the client environment has been carefully understood, specifically with respect to roles of users in the system. These roles have been understood in terms of user activity and frequency of use. Also, the system's hardware and software environments have been studied along with the interaction of the system with other information systems, mainly the Central Authentication Service (CAS).

The team has highlighted the functioning of the current system with the help of Entity-Relationship (ER) diagrams and Data Flow Diagrams (DFD's). The availability of the source code and documents is minimal but a process of reverse engineering covers some of the more important aspects of the current system.

The ER diagram captures most of the relationships and constraints in the proposed system. Some of the constraints cannot be modeled and are, therefore, explicitly stated. The DFD's (Data Flow Diagrams) capture the essential data flowing in the system specific to the various users of the system. In addition, important functional, qualitative and maintenance requirements are considered with special focus on client needs. The functional requirements have been prioritized and stated clearly.

A user interface that meets the client requirements has been proposed. Additionally, other factors such as training after the installation and limitations of the proposed system have been analyzed.

Also, a Quality Assurance Plan (QAP) is proposed to ensure the quality of the proposed system. A Project Plan is also attached to this document. The roles and tasks of each team member are also stated.

Client questions, considerations and concerns regarding different aspects of the document have been noted at the end.

Contents

1	Introduction	4
2	Client Background	5
3	Goals of the Project	7
4	Environment	8
4.1	Users and Roles	8
4.1.1	Primary Client and Contact	8
4.1.2	Information Systems Staff	8
4.1.3	Professors	8
4.1.4	Instructors	9
4.1.5	Administrator	9
4.1.6	Students	9
4.2	Hardware Platforms	9
4.3	Software Platforms	10
4.4	Interaction with Other Information Systems	10
4.5	Impact on Operations	10
5	Current System	11
5.1	Current Entity-Relationship Model	11
5.2	Data Flow Diagrams for Current System	12
5.3	Other Available Documents	15
6	Proposed System	16
6.1	Overview	16
6.2	Proposed Entity-Relationship Model	16
6.3	Data Flow Diagrams	18
6.4	Interaction of ER and DF Models	36
6.5	Functionality Requirements	39
6.6	Business Rules	42
6.7	Maintenance Requirements	43
6.8	Qualitative Requirements	43
6.9	User Interface	44
6.9.1	Conventions	44
6.9.2	Login	45
6.9.3	Gateway	46
6.9.4	Offering - Student	47
6.9.5	Offering - Professor	48
6.9.6	Component - Professor	49
6.9.7	Section - Instructor	50
6.9.8	Administrator Gateway	51
6.9.9	Offering - Administrator	52
6.9.10	Component - Administrator	53
6.10	Installation	54

6.11	Limitations	54
7	Quality Assurance Plan	55
7.1	Purpose and Scope	55
7.2	Organization, Tasks, and Responsibilities	55
7.3	Documentation Required	55
7.4	Standards, Practices, and Conventions	56
7.4.1	Program Standards	57
7.5	Reviews and Audits	57
7.6	Configuration Management	58
7.7	Problem Reporting and Corrective Action	58
7.8	Tools, Techniques, and Methodologies	58
7.9	Code Control	58
7.10	Media Control	59
7.11	Supplier Control	59
7.12	Records Collection, Maintenance, and Retention	59
8	Appendices	60
9	Client Comments	61

List of Figures

1	The Client's Business Model	6
2	ER Current System	11
3	ContextLevel Current System	12
4	L0 Current System	12
5	L1-1 Current System	13
6	L1-2 Current System	13
7	L2-2.1 Current System	14
8	L2-2.3 Current System	14
9	L2-2.4 Current System	15
10	ER Proposed System	17
11	VennDiagram Proposed System	18
12	ContextLevel Proposed System	19
13	L0 Proposed System	19
14	L1-1 Proposed System	20
15	L1-2 Proposed System	21
16	L1-3 Proposed System	22
17	L2-3.2 Proposed System	23
18	L3-3.2.5 Proposed System	24
19	L2-3.3 Proposed System	25
20	L1-4 Proposed System	26
21	L1-5 Proposed System	27
22	L1-6 Proposed System	28
23	L2-6.4 Proposed System	29

24	L3-6.4.2 Proposed System	30
25	L3-6.4.5 Proposed System	31
26	L4-6.4.5.2 Proposed System	32
27	L4-6.4.5.4 Proposed System	32
28	L4-6.4.5.5 Proposed System	33
29	L4-6.4.5.8 Proposed System	34
30	L4-6.4.5.9 Proposed System	34
31	L3-6.4.6 Proposed System	35
32	L3-6.4.7 Proposed System	35
33	L2-6.5 Proposed System	36
34	User Interface for Login	45
35	Gateway Screen for Regular Users	46
36	Offering Screen for a Student	47
37	Offering Screen for Professor	48
38	Component Screen for Professor	49
39	Section Screen for Instructor	50
40	Administrator Gateway Screen	51
41	Offering Screen for Administrator	52
42	Component Screen for Administrator	53

List of Tables

1	Login Screen	45
2	Gateway Screen for Regular Users	46
3	Offering Screen for a Student	47
4	Offering Screen for Professor	48
5	Component Screen for Professor	49
6	Section Screen for Instructor	50
7	Administrator Gateway Screen	51
8	Offering Screen for Administrator	52
9	Component Screen for Administrator	53

1 Introduction

The Physics Department at Indiana University is a non-profit education institution. It provides quality education in physics to students on campus. The department also offers undergraduate and graduate level programs in the physics major.

The Physics Department currently uses OnCourse as its gradebook system and has trouble storing and managing these grades. This is due to its complex organizational structure. This inefficient method of management of grades has necessitated the development of a new system. The proposed system will be used by the Physics Department as a means of entering and storing grades based on the organizational structure mentioned in Section 2. After studying the client environment, the proposed system's characteristics, the possible technical issues in the development of the system and evaluating possible alternatives in the feasibility study[1], the team found that such a system can be successfully built.

This document begins with the client background in Section 2, where the client organizational structure and the problems that the client faces with the current system are analyzed. The goals of the proposed system are then stated in Section 3. In Section 4, the team has studied the client environment, including the users, the hardware and software environments, and the interaction of the system with other information systems. In Section 5, the current system is analyzed with the help of Entity-Relationship (ER) diagrams and Data Flow Diagrams (DFD's).

The requirements of the proposed system are finally examined in details in Section 6. The ER diagram, DFD's and their interactions are modeled in Section 6.2, Section 6.3 and Section 6.4, respectively. The functionality requirements, business rules, maintenance requirements and qualitative requirements are specified through Section 6.5 to 6.8. A user interface design that can achieve these requirements is proposed in Section 6.9. Other factors, including training required after installation of the proposed system and the limitations of the proposed system are stated in Section 6.10, and 6.11.

A draft Quality Assurance Plan (QAP) is covered in Section 7.

The comments and feedbacks from the primary client are summarized in Section 9.

A Project Plan is supplied in the appendix of the document.

2 Client Background

The primary client of the project is the Physics Department at Indiana University, Bloomington. The Physics Department provides quality education to students on campus in broad topics of physics in undergraduate and graduate level programs for physics majors and minors. The client organization is divided into two primary divisions, the Research Division and the Teaching Division. The Teaching Division is the main target of this project. The Research Division is not of concern for the proposed system.

The Teaching Division consists of professors and associate instructors. A professor and several associate instructors form a Teaching Unit, that is responsible for offering courses to the students. Each individual member in the Teaching Division can participate in multiple units. On the course hierarchy side, a course consists of various components, including but not limited to lecture component, lab component, discussion component and recitation component. Students enrolled in each component are organized by sections. Also, the set of sections in one component need not be the same as the set of sections in another component.

It must be noted that a professor is responsible for the entire course, whereas each instructor is responsible for one or maybe more sections.

The primary client, Dan Beeker, co-ordinates the teaching units. The current system, OnCourse, does not allow for this to be done efficiently. The problem with using OnCourse is that it only allows for the creation of courses and then for the assignment of professors, instructors and students to each of the courses. It does not allow for the hierarchy shown in Figure 1. Therefore, the client organization currently creates a new “course” for each section that is needed within an actual course. Students are also assigned to each such new “course”.

Also, each teaching unit is required to assign grades to students for each assignment within a section of a component. Currently, the client organization uses various workarounds to accomplish the above task. Many grades from previous semesters have been lost and assigning grades for the semester in progress has been a cumbersome task. This is because the number of grade entries to be made for each student is very large and ambiguous.

All of these complexities, in turn, become a very difficult management and storage issue for the administrator at the end of the semester with respect to the grades of each student.

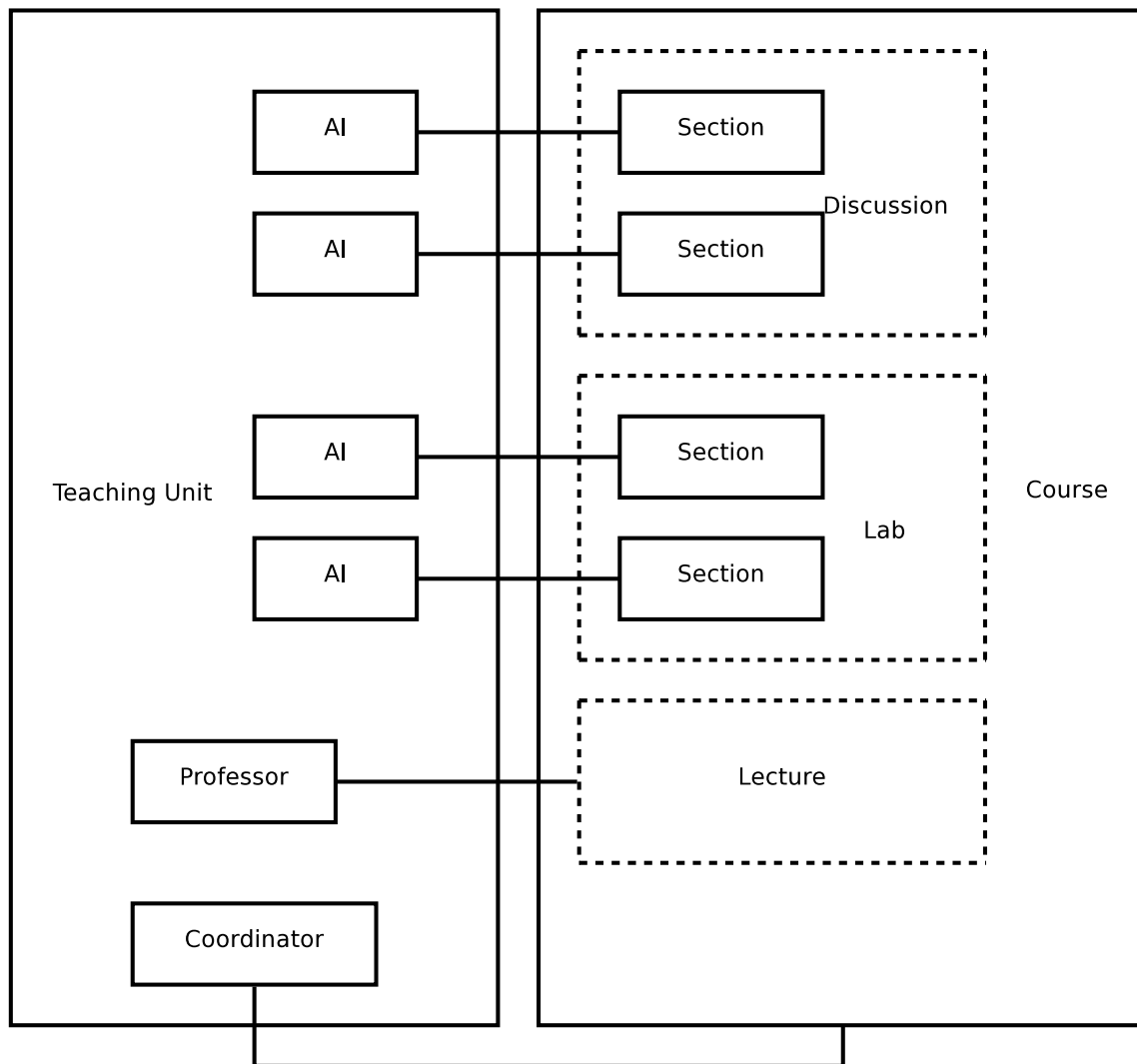


Figure 1: The Client's Business Model

3 Goals of the Project

The goal of the project is to precisely represent the unique course structure and efficiently manage the grades assigned to the students over years.

Handle Complexity The Physics Department has a fairly unique course structure as explained in Section 2. This unique structure increases the complexities in the allocation of entities for the administrator. Assigning grades for students also becomes a more cumbersome task. The new system should help the client handle this complexity.

Provide Efficiency The primary client specifically requested the system to be direct and efficient. The user interface should be simple; the model should be concise, and the interaction between the system and the users should contain as few steps as possible.

Ensure Privacy The system deals with the students' grades. A student is assigned a grade for each assignment and these grades are private to the individual students. Therefore, ensuring privacy is also an implied goal of the system.

4 Environment

4.1 Users and Roles

Every semester, the system is expected to interact with approximately 1000 students, 20 professors, 40 instructors and the primary client (administrator).

The administrator, professors and instructors form the internal users of the system and the students form the external users of the system.

4.1.1 Primary Client and Contact

The Primary Client for the proposed system is Daniel Beeker, the Undergraduate Laboratory Coordinator. Daniel Beeker is the head staff for undergraduate education in the teaching unit. He has requested a new gradebook system with very clear objectives to the team.

The primary client has sufficient resources for ongoing operation and maintenance of the proposed system. He is generally available in his office. It is unlikely that he will be absent for any extended periods during the course of the project.

The Primary Client can be reached at:

Daniel Beeker
Undergraduate Physics Laboratory Coordinator
Swain West Room 115
727 E. 3rd St. Bloomington, IN 47401
Phone Number: 812-855-5903
Email: debeeker@indiana.edu

4.1.2 Information Systems Staff

The client already has an IT staff to maintain servers and computers and does not intend to hire new staff. The current staff will be responsible for maintaining the proposed system.

4.1.3 Professors

Professors in the system are in charge of each of the courses that they teach. The professors are fully responsible for all such courses.

The professor provides assignments to students within each of the components of the course. This task is generally done as and when a new assignment is to be given to the students during the course of the semester. It is also possible that a few assignments are provided to the students in a bulk by the professor. An error in the assignment name would mean that the professor might change the name (modify) of the assignment. The professor can also, possibly, revoke an assignment that is deemed not required.

The professor may view the student roster at both the course and component levels.

The professor may also assign grades to any student for any assignment in any section of the course. This is generally done after the completion of each assignment. In practice, the professor does not assign grades for the lab, discussion and recitation components. He could, however, assign grades for students in the lab component if required. Such cases are likely to be rare because a professor does not directly conduct any lab session.

The final grades of all students for each component and also the final course grade can be obtained by the professor.

4.1.4 Instructors

Instructors can be assigned to multiple sections across components in a course. For each such section, they may assign grades to students. They may not assign grades to students belonging to any other section. This task is generally done after the completion of each assignment. This means that they can overwrite grades previously assigned by the professor.

4.1.5 Administrator

The primary client is the administrator. He has complete access to the system. He accomplishes the following:

- Setting up the course structure for the current course offering. This is generally done just before the beginning of the semester.
- Assigning professor(s) to a course offering. This is done just before the beginning of the semester.
- Assigning instructors to various sections. This may be done just before the beginning of the semester or just after the semester begins or even during the semester.
- Adding students to the roster for both course and components. This is generally done just before the beginning of the semester or sometimes during the semester.
- Reviewing gradebooks for all components and courses and generating reports. This is generally done at the end of the semester.
- Making backups of the system. This may be done at regular intervals of time.

The administrator is familiar with the above mentioned tasks.

4.1.6 Students

Students are external users of the system. A students in a course simply view the list of assignments and the grades assigned to them for each of the assignments. Student can only view their own grades. The grades would be available for viewing as soon as the professor or instructor puts them up. The final grades are generally available in the week following the end of the semester.

It must be noted that a student can also be an undergraduate instructor.

4.2 Hardware Platforms

The client currently has Personal Computers(PC's) with Pentium 4 Central Processing Units (CPU's) that are suitable for running Windows 2000 Server. These machines can be used for development purposes. The computing capacity of the PCs is sufficient for the proposed system. The final hardware environment might differ from that of the development machine, and therefore an installation stage is expected.

4.3 Software Platforms

The software platform for the proposed system will be built upon productive versions of several pieces of OpenSource Software.

- For the Database management system, MySQL will be deployed.
- For the HTTP service daemon, Apache will be deployed.
- For the Web applications, PHP runtime (with development runtime) will be deployed.

A software environment mentioned above is easy to set up and can be accessed from any personal or commercial web hosting service.

4.4 Interaction with Other Information Systems

The proposed system interacts with the Central Authentication Service (CAS) for authenticating regular users. [5]

4.5 Impact on Operations

After the deployment of the proposed system, the team will prepare the primary client on the usage of the system. The primary client in turn will be responsible for training the other users of the system. The team will also provide a user manual for the proposed system.

Initial familiarization with the system may slow down operations a little, but is not expected to last too long.

Also, the Physics Department is expected to stop using OnCourse for updating and keeping track of grades. Final grades are sent to the professor in charge of the course. Along with the proposed system, OnCourse will also be used to put up final grades.

5 Current System

The client currently exploits the Gradebook module of the OnCourse system. The gradebook system is simple, and contains most functionalities, especially

- course/section management,
- roster management,
- instructor¹ allocation,

are cascaded to other subsystems of OnCourse.

Direct document for OnCourse is not available; however, documentation of the Sakai Project[4], which is the source for the creation of OnCourse, is available. Since there are no substantial differences between OnCourse and Sakai, the team has analyzed Sakai for this section.

5.1 Current Entity-Relationship Model

In the current system, there are three types of users: Student, Instructor and Associate Instructor(AI)/Undergraduate Instructor(UI). The modification history of grades is not stored. An offered course is called Section.

Although Sakai has its own user authentication system, OnCourse utilizes Indiana University's computing account (itaccount) for identifying users. This fact is also captured in the ER Model.

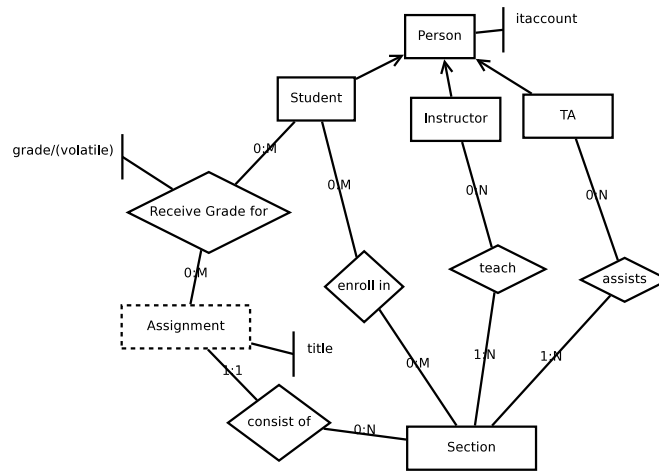


Figure 2: ER Current System

¹Here the notation of the current system is used. Instructor in the current system corresponds to Professor in the proposed system; TA in the current system corresponds to Instructor in the proposed system.

5.2 Data Flow Diagrams for Current System

This part consists of two sections. The definitions of the data flow are defined in the first section. The data flow for the current system is described by Data Flow Diagrams in the second section. The data flow in the Current system is simple.

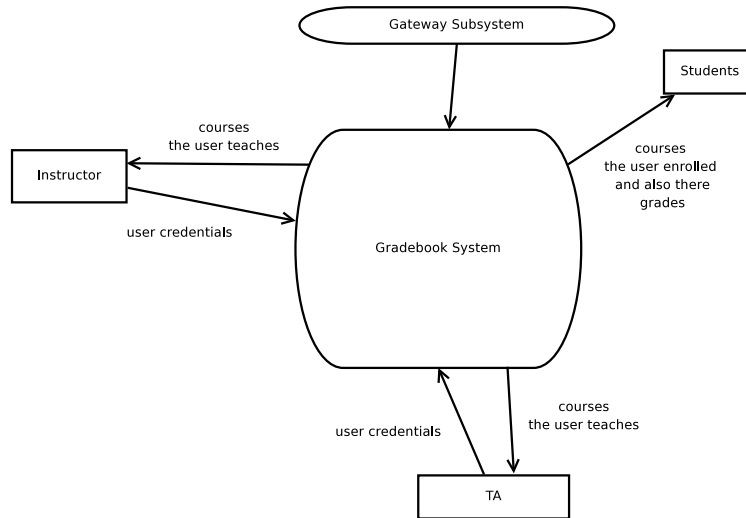


Figure 3: ContextLevel Current System

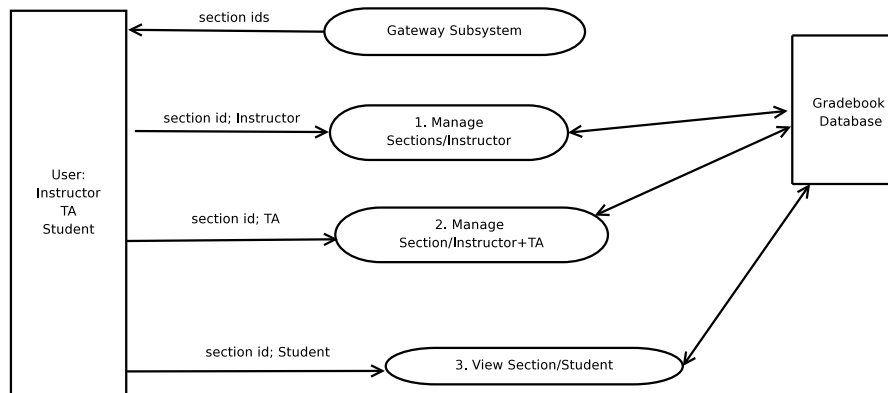


Figure 4: L0 Current System

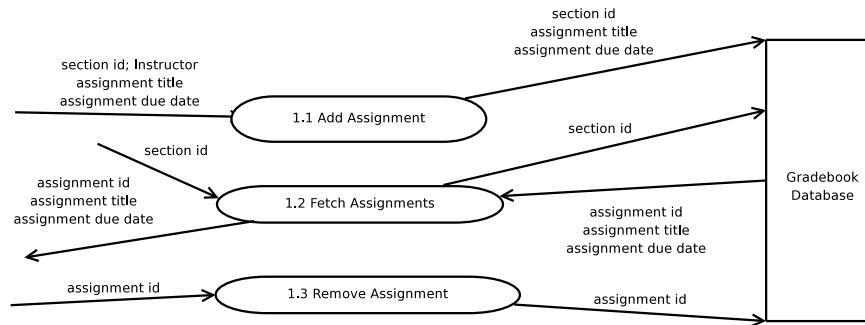


Figure 5: L1-1 Current System
In this diagram, the instructor manages the assignments in a section.

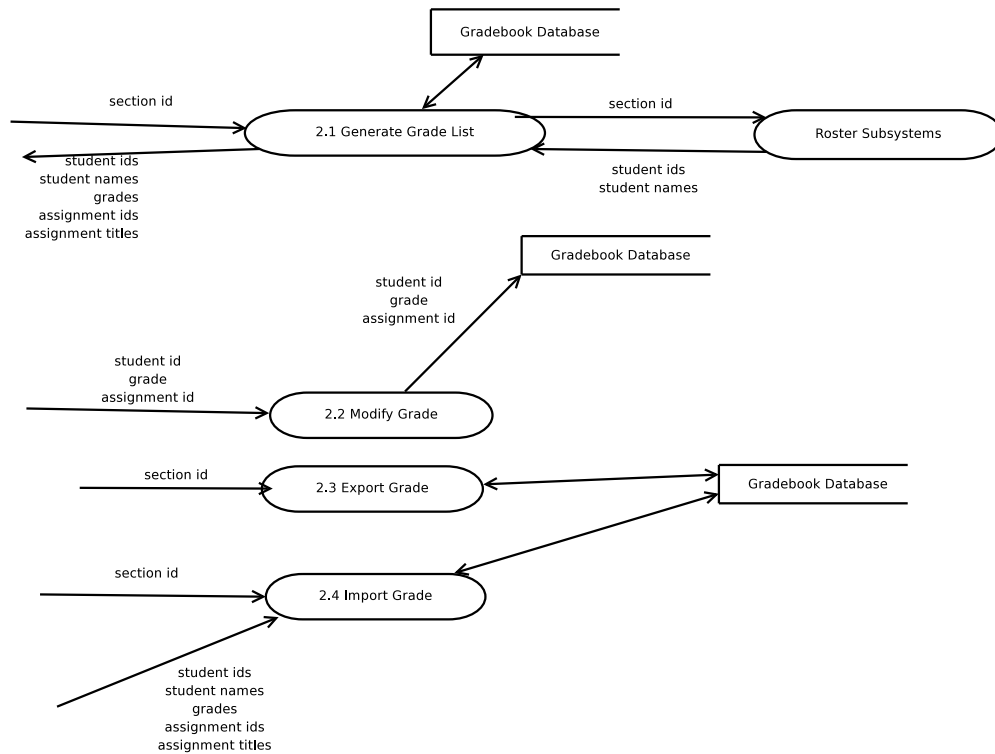


Figure 6: L1-2 Current System
In this diagram, the instructor or the TA manages the grades for a section.

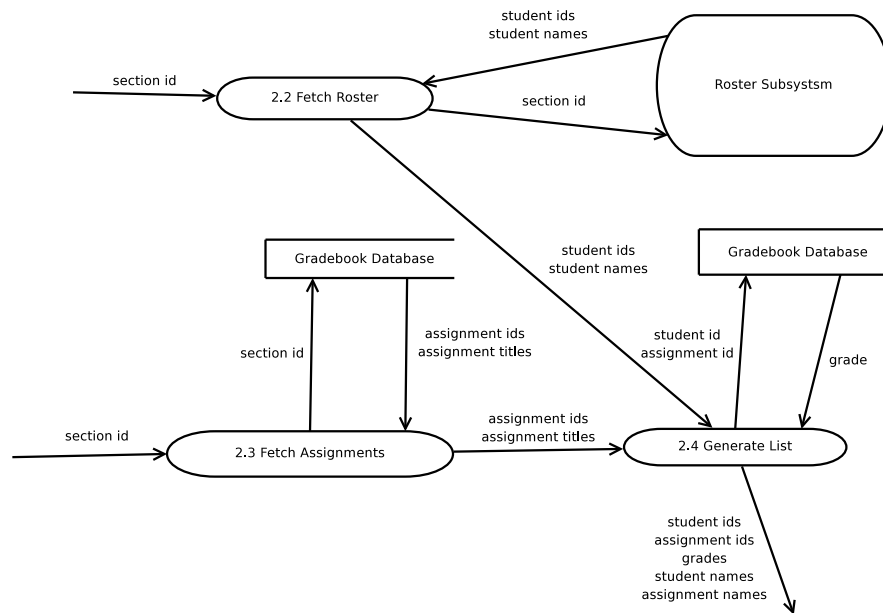


Figure 7: L2-2.1 Current System
In this diagram the process for generating the grade report is modeled.

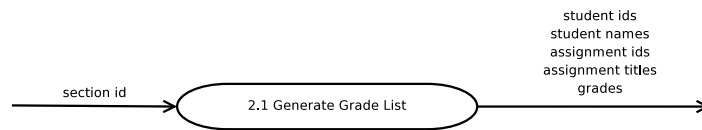


Figure 8: L2-2.3 Current System
In this diagram the process L2-2.1 is reused to export the grade report to an external format.

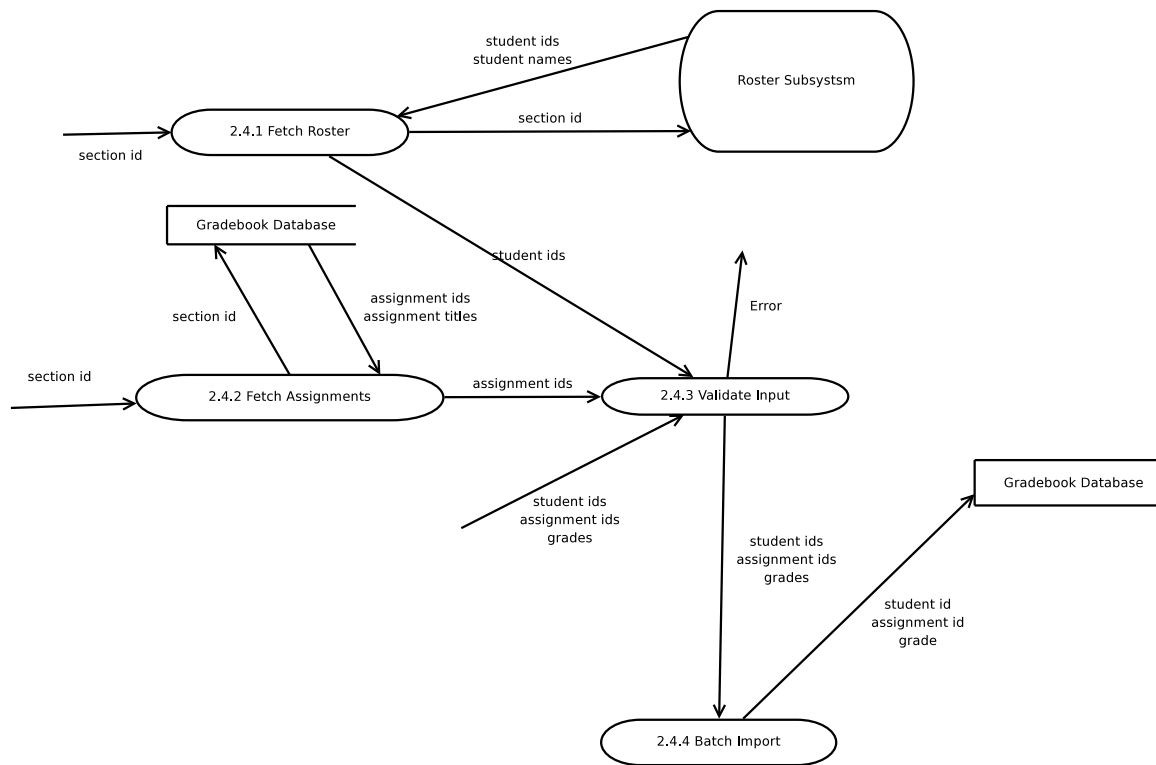


Figure 9: L2-2.4 Current System

In this diagram, the batch importing for grades is described. Note that the list of students, which is used to validate the importing data, is obtained from an external system.

5.3 Other Available Documents

The Sakai² Project provides a description of the functional requirements of their gradebook subsystem[4].

²OnCourse is a derivative of Sakai.

6 Proposed System

6.1 Overview

The goal of the project is to precisely represent the unique course structure and efficiently manage the grades assigned to the students over years. In order to handle the complexity of the course structure and to provide efficiency in the client's relevant activities, the system should meet specific requirements which are obtained by analyzing the client organization with Entity-Relationship and Data Flow modeling techniques. The Functionality Requirements in this section is the most important part of the document because it specifically describe how the proposed system can achieve the project goals. The privacy of the grade data is guaranteed in the functionality requirements.

The user interface has to be simple and efficient, which is also studied in this requirement specification.

However this section is not limited to the requirements of the proposed system. In order for the system to succeed after installation, requirements for the client to maintain the system are also investigated.

6.2 Proposed Entity-Relationship Model

The Entity-Relationship diagram of the proposed system is shown in this subsection. In order to overcome the difficulties that the client faces with OnCourse, an alternative model for Course/Offering/Component/Section is captured in the ER diagram. In contrast, the ER diagram of the current system does not contain a corresponding part because the functionality requirements are resolved in other modules of OnCourse.

Also, notice that the Roles (Student, Instructor, Professor) are weak³; Person establishes the role only if it participates in specific relations. As a compromise, the Person entity is used to precisely capture these volatile roles. An alternative model would be not to specialize the Person entity to three sub-types but we lose a lot of constraints in this process. Another requirement was to model the fact that a student can be an undergraduate instructor. However, modeling this makes the diagram very messy and therefore we have chosen to explain this by means of a Venn Diagram.

³This "weak" is different from the word 'weak' used in weak entity.

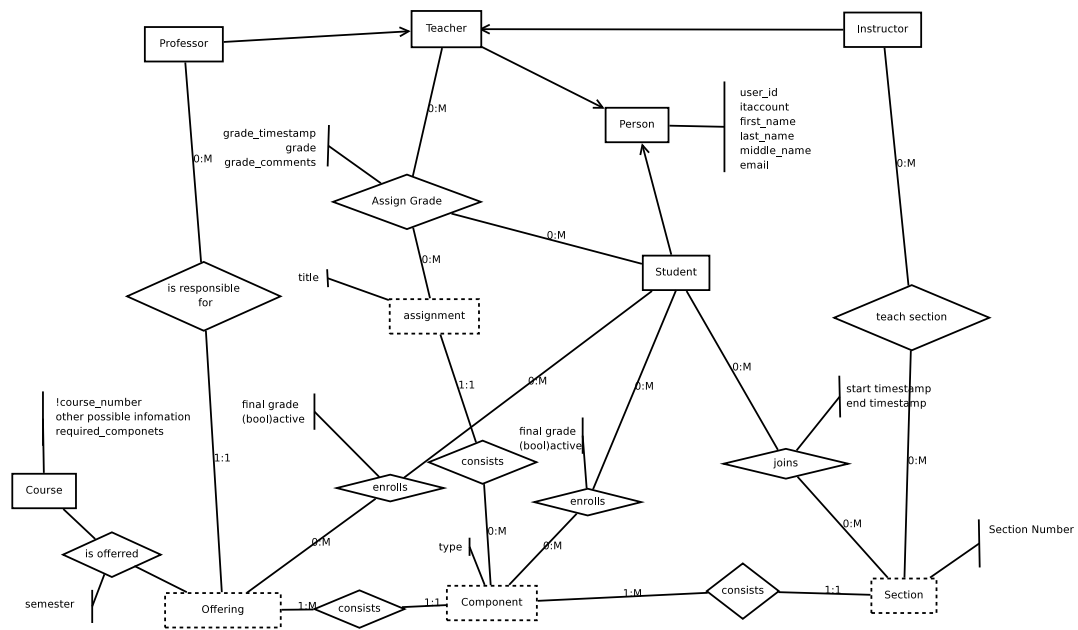


Figure 10: ER Proposed System

Figure 11: VennDiagram Proposed System
The overlapping illustrates that a person can be a student and an instructor at the same time.

6.3 Data Flow Diagrams

Convention: Double sided arrows that are not labeled indicate that corresponding process(es) are drilled down to the next level.

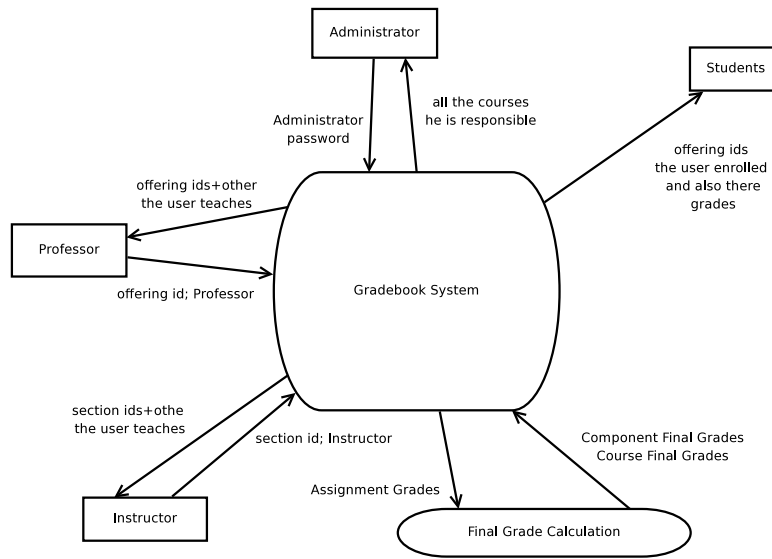


Figure 12: ContextLevel Proposed System

The Context Level diagram shows the overall data flow between the system and the real world.

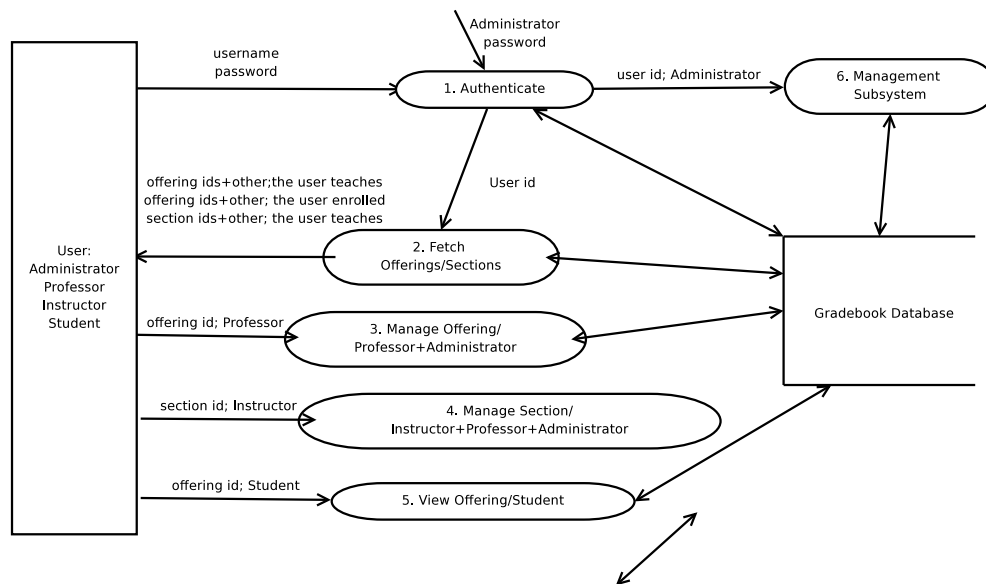


Figure 13: L0 Proposed System

In this diagram, the major components of the system are illustrated. User authentication is in L1-1. The role of the user is then established in L2-2.

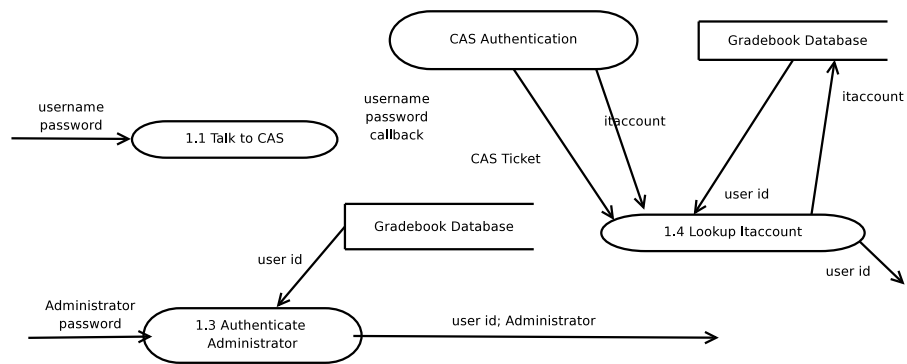


Figure 14: L1-1 Proposed System
L1-1

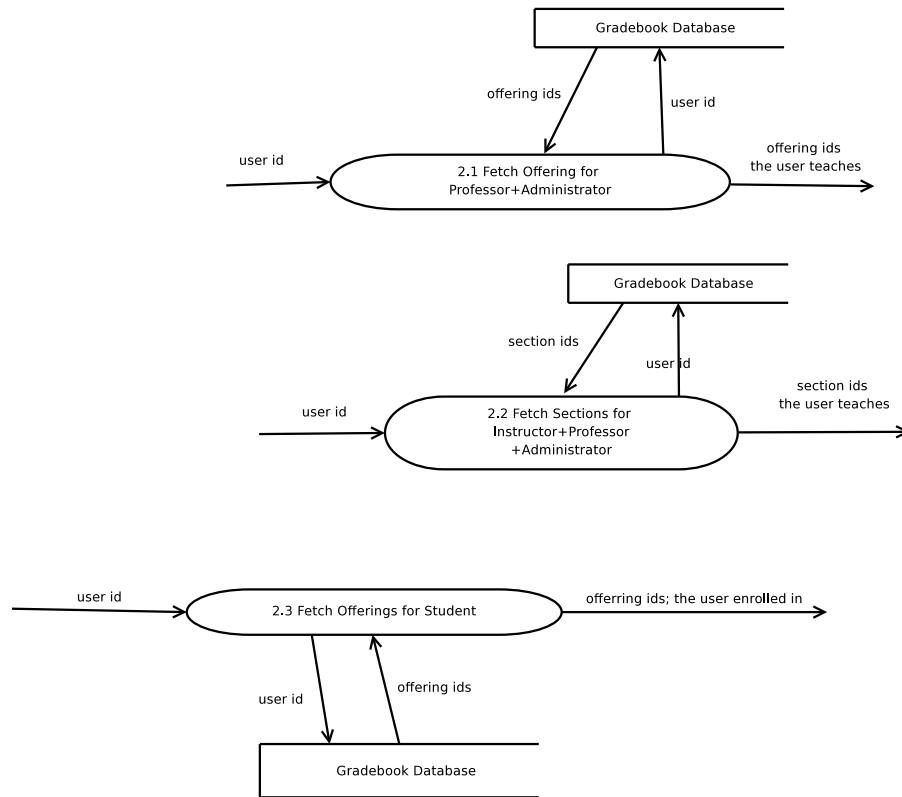


Figure 15: L1-2 Proposed System

In this diagram, the user obtains the related courses and sections in the system, and makes a selection. The determination of the role of the user is deferred to this process after the login stage.

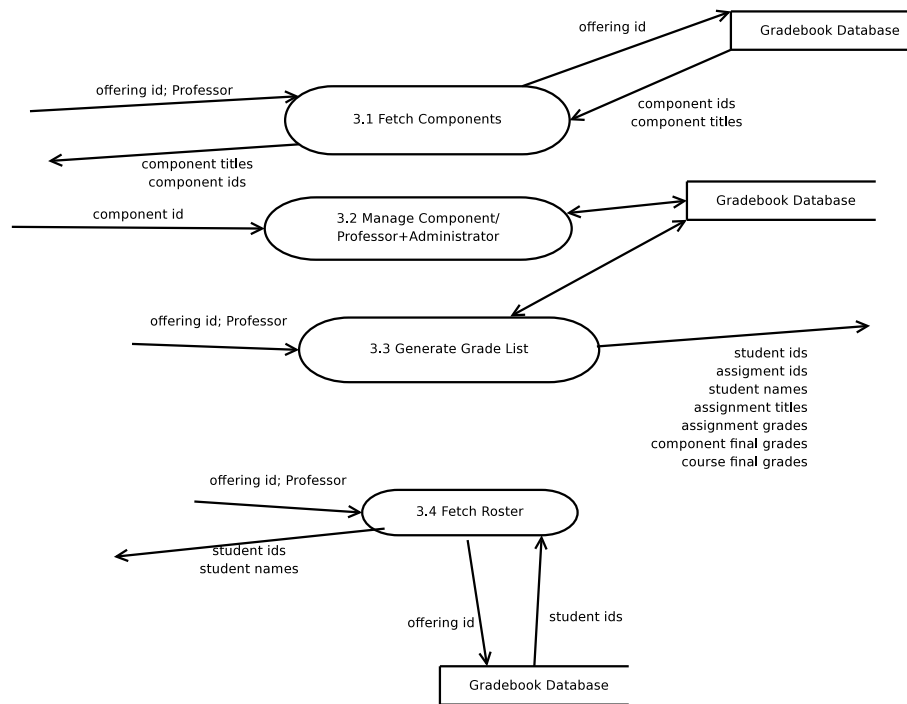


Figure 16: L1-3 Proposed System

In this diagram, the professor performs management tasks on an offering. The management for a component is drilled down to next level. The untermiated flows are from and to the user.

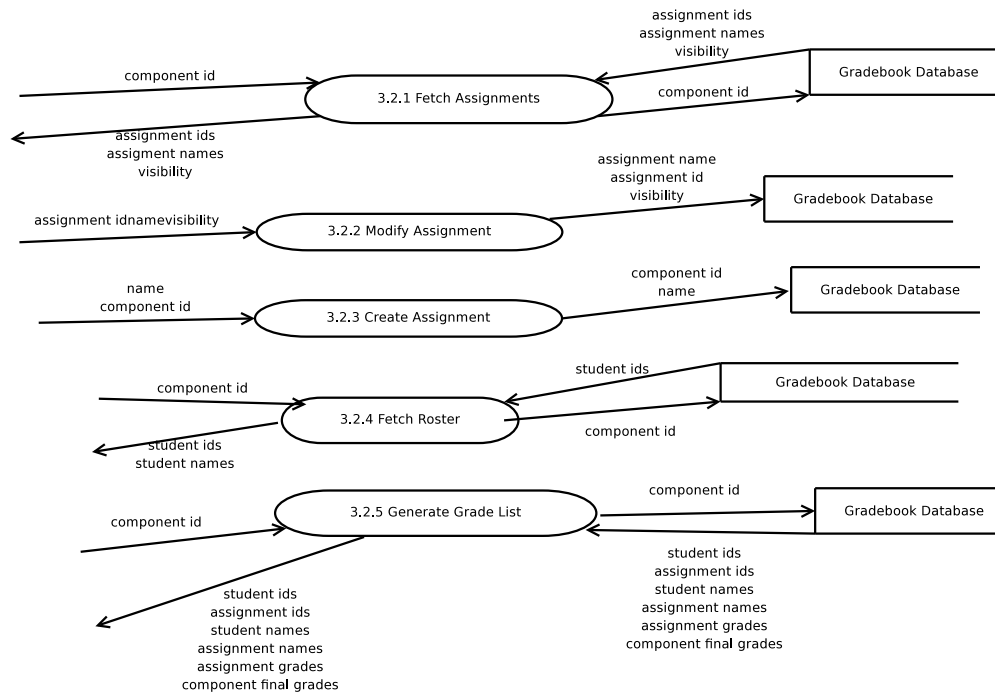


Figure 17: L2-3.2 Proposed System

In this diagram, the professor performs management tasks on a component.

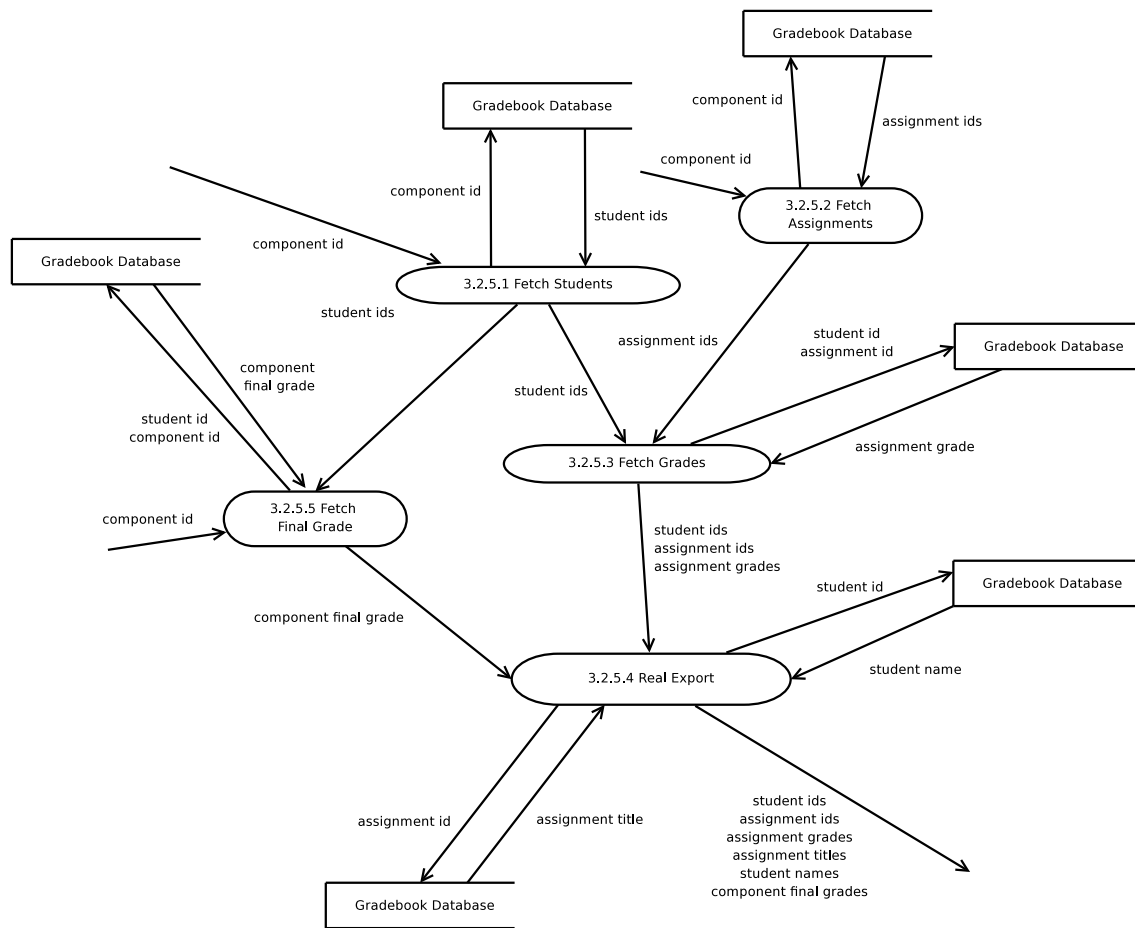


Figure 18: L3-3.2.5 Proposed System

In this diagram, the professor obtains a grade report for the component. A grade report contains the students, their grades for each assignment, and the final grades if there are any.

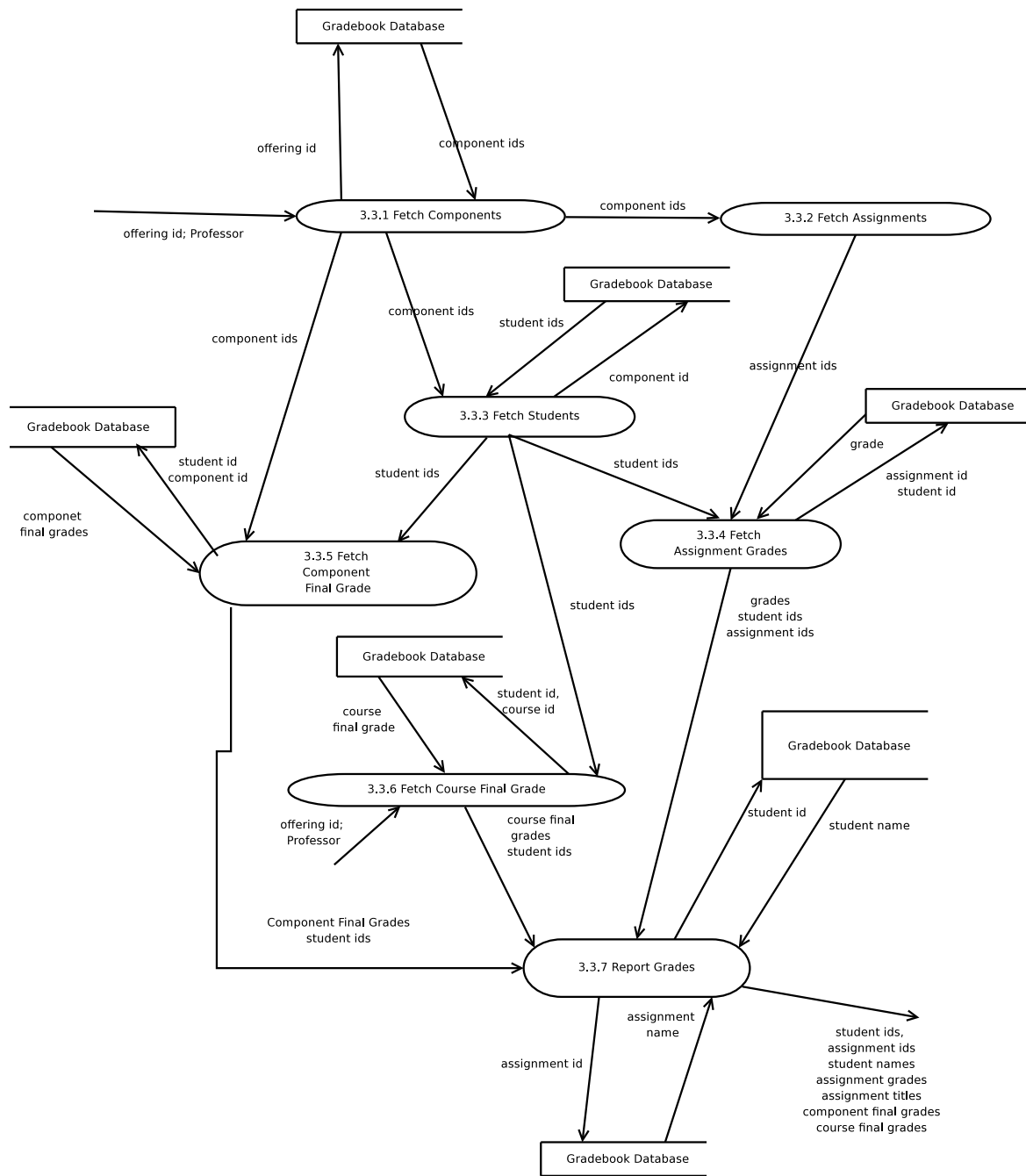


Figure 19: L2-3.3 Proposed System
In this diagram, the professor obtains a course grade report of the offering.

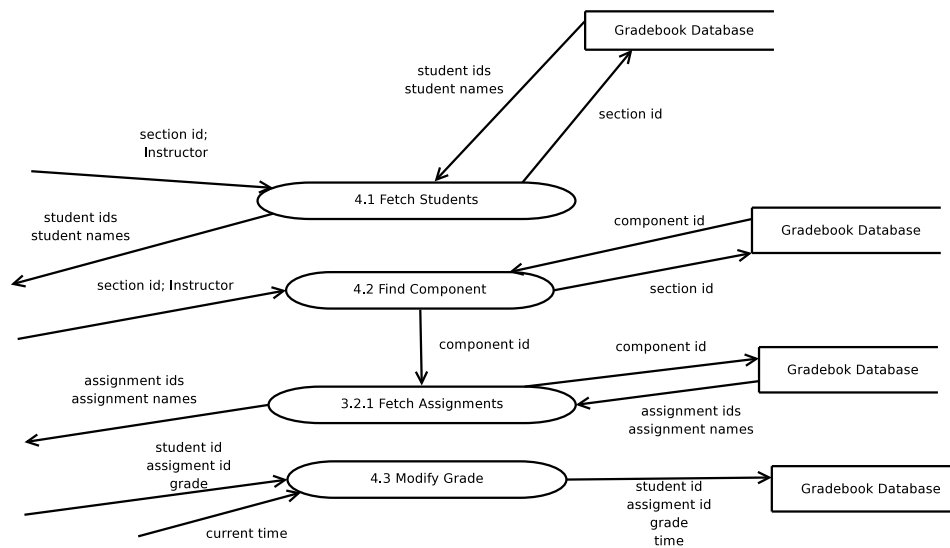


Figure 20: L1-4 Proposed System
In this diagram, the instructor performs grade entry on a section.

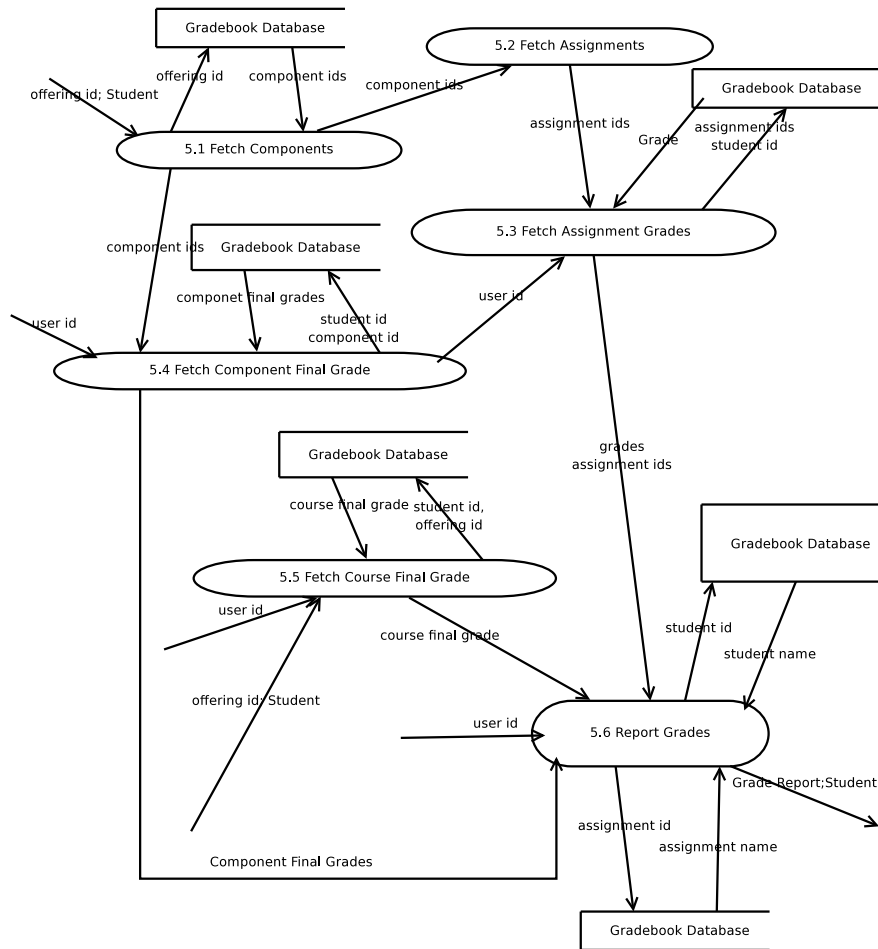


Figure 21: L1-5 Proposed System
In this diagram, the student obtains a grade report for an offering.

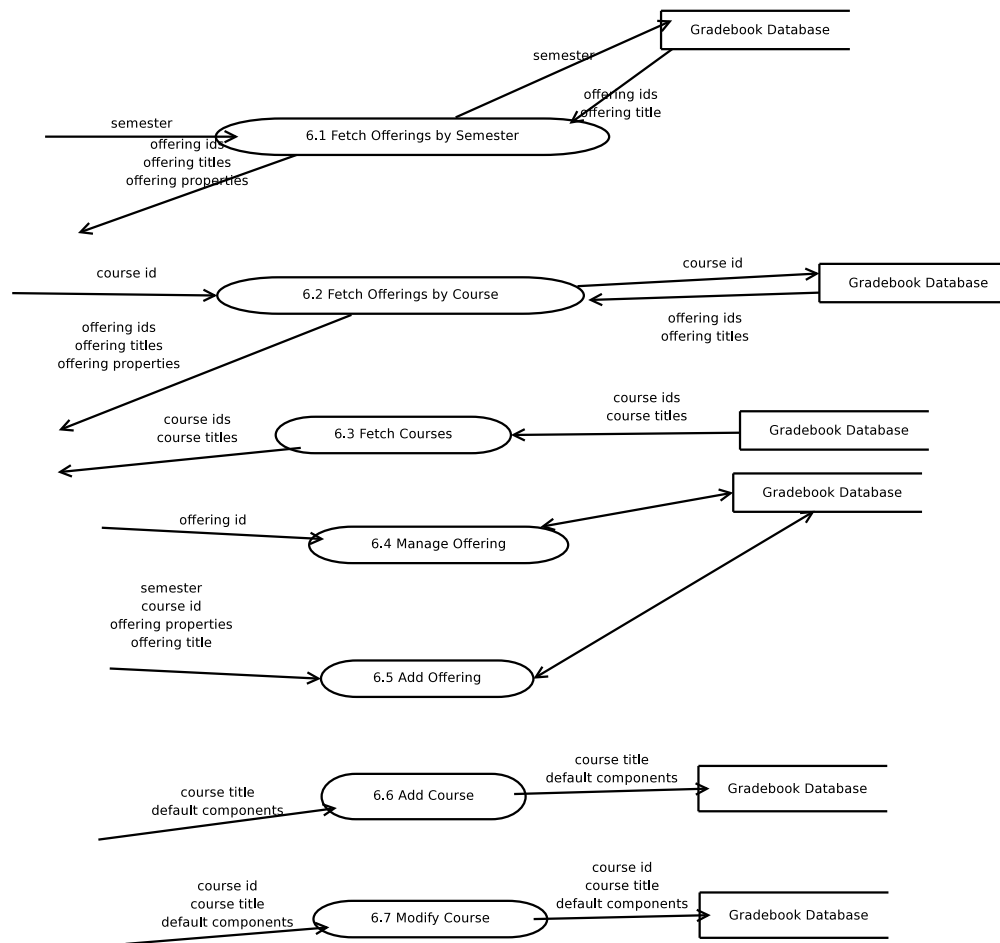


Figure 22: L1-6 Proposed System

In this diagram, the management subsystem - the necessary interaction between the administrator and the system to support the ER model is described.

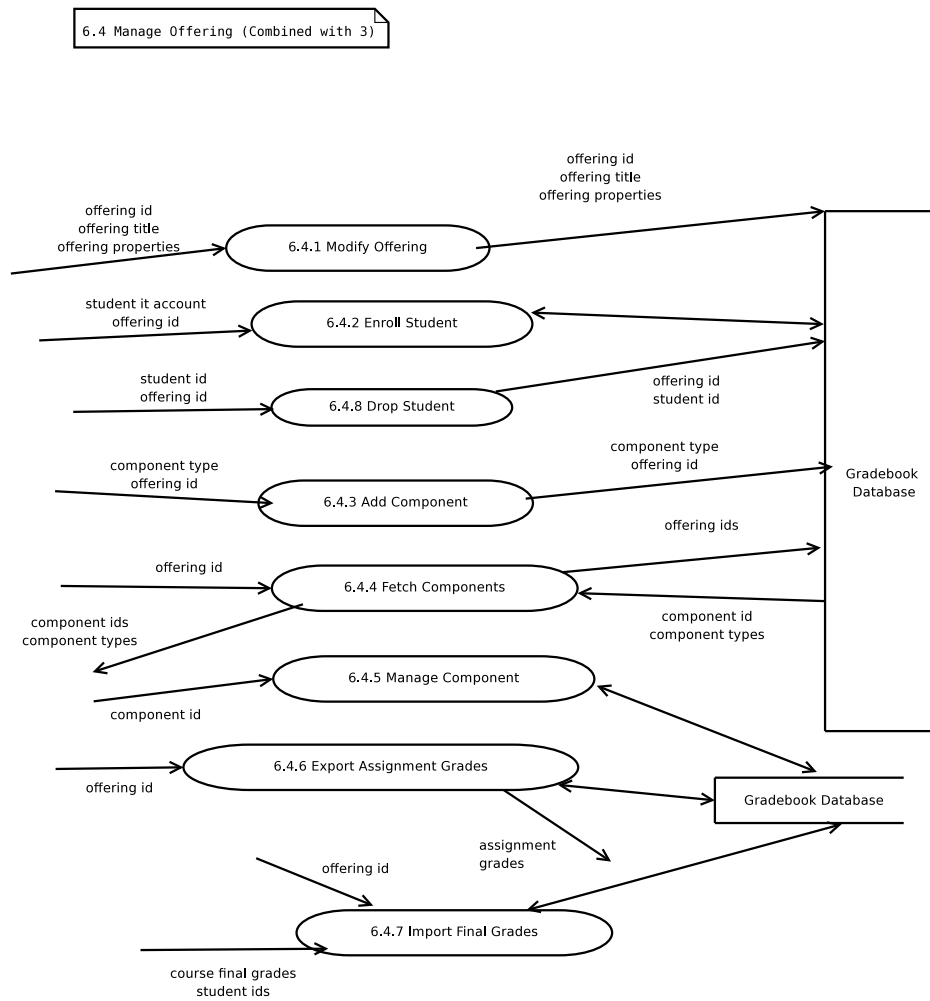


Figure 23: L2-6.4 Proposed System

In this diagram, the administrator performs management tasks on the offering. Note that this L2-3.3 should be attached to this diagram because the administrator can also perform management tasks that are performed by professors.

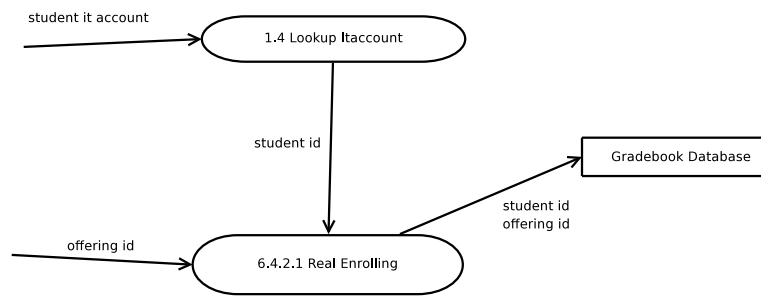


Figure 24: L3-6.4.2 Proposed System
In this diagram, the administrator adds a student to an offering.

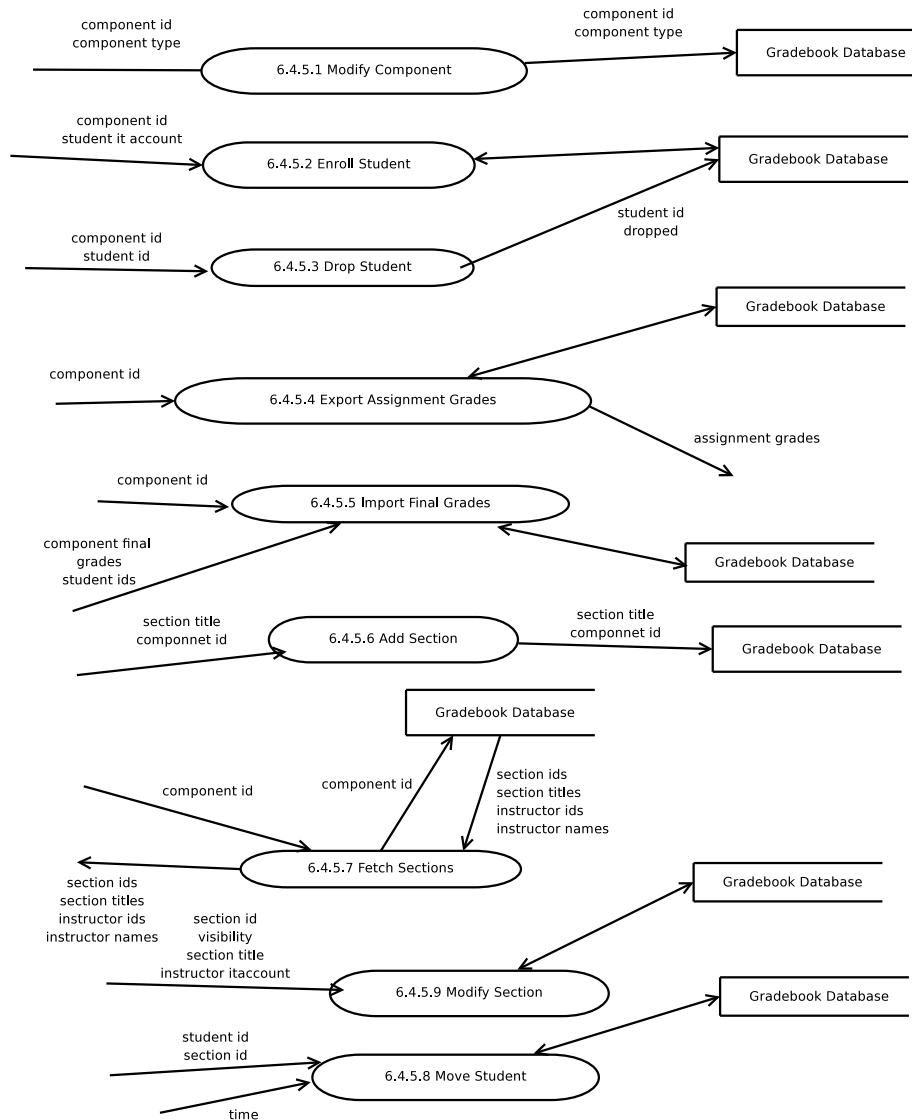


Figure 25: L3-6.4.5 Proposed System

In this diagram, the administrator performs management tasks on the component. Note that this diagram should be combined with L2-3.2, because the administrator can also perform the tasks that a professor can perform.

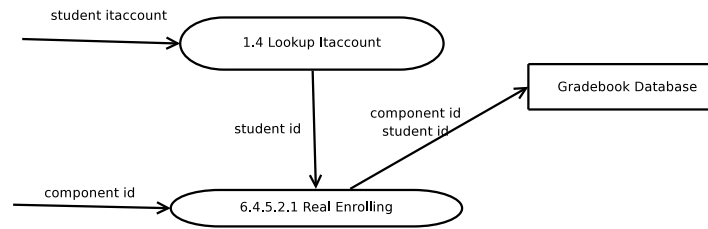


Figure 26: L4-6.4.5.2 Proposed System

In this diagram, the administrator enrolls the student to the component. A constraint is that the student has to be in the offering that offers the component.

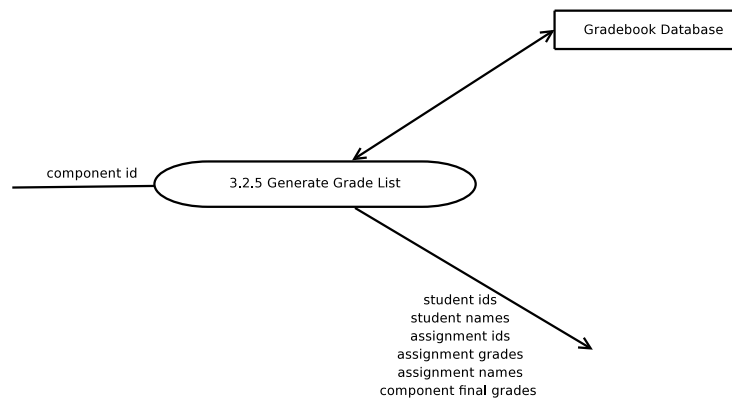


Figure 27: L4-6.4.5.4 Proposed System

In this diagram, the administrator exports the grades for a component. L3-3.2.5 is reused in this process.

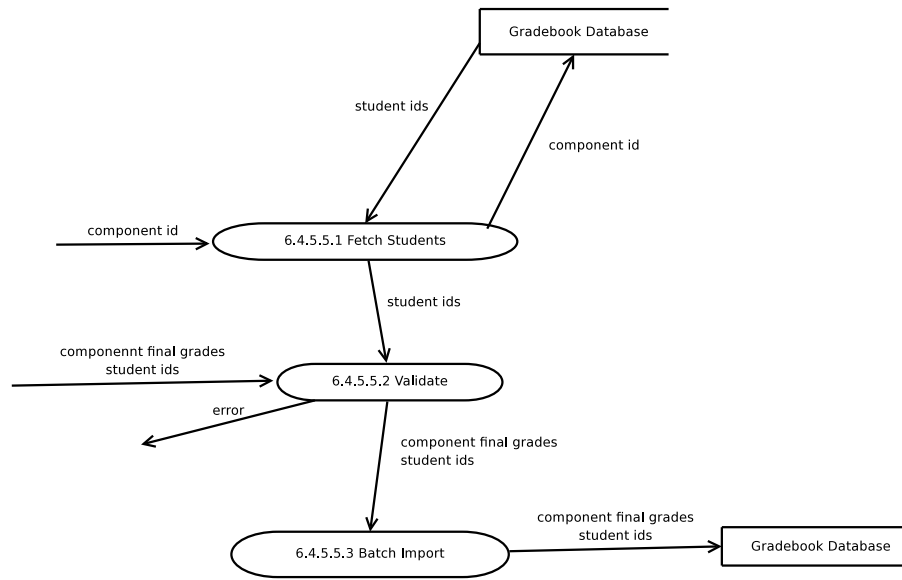


Figure 28: L4-6.4.5.5 Proposed System

In this diagram, the administrator imports the final grades for a component to the system. The assignment grades, if existing in the importing, are ignored.

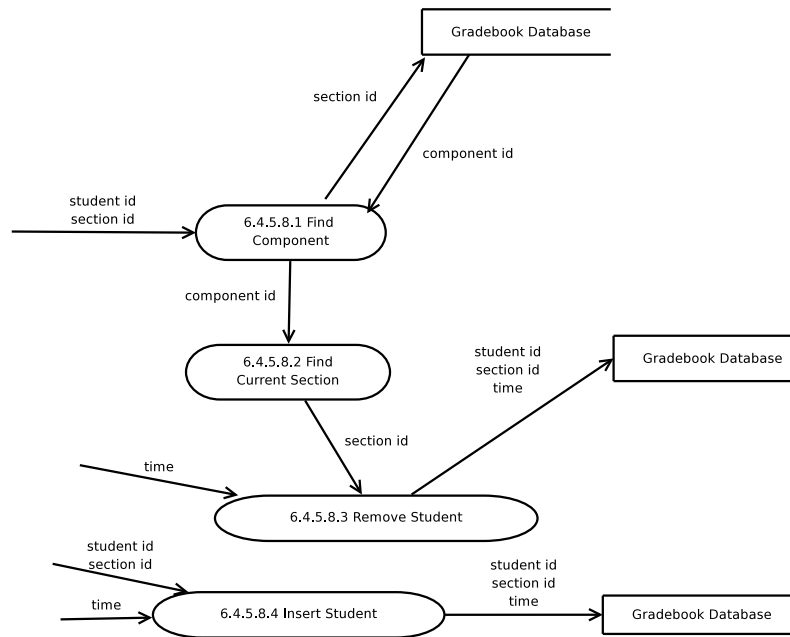


Figure 29: L4-6.4.5.8 Proposed System

In this diagram, the administrator moves the student from one section to another section. A constraint is that the two sections have to be within the same component. Another constraint is that the process has to be in one transaction to conserve the consistency.

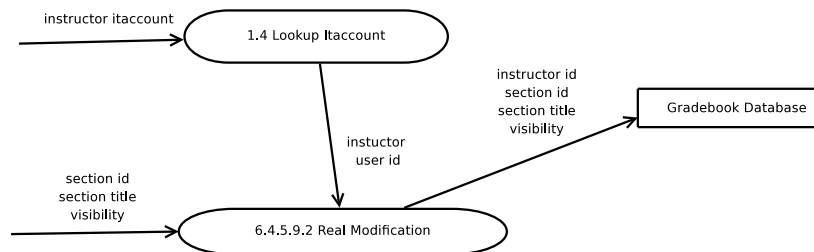


Figure 30: L4-6.4.5.9 Proposed System

In this diagram, the administrator modifies a section. The instructor of the section is also assigned in this process. Therefore a user id lookup process is required and L2-1.4 is reused.

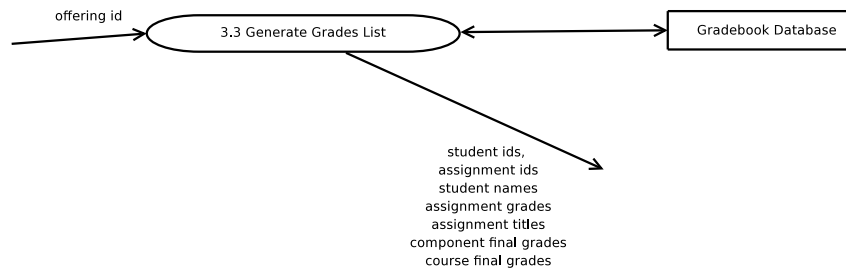


Figure 31: L3-6.4.6 Proposed System

In this diagram, the administrator exports the course grades for an offering. Note that L2-3.3 is reused to generate the report.

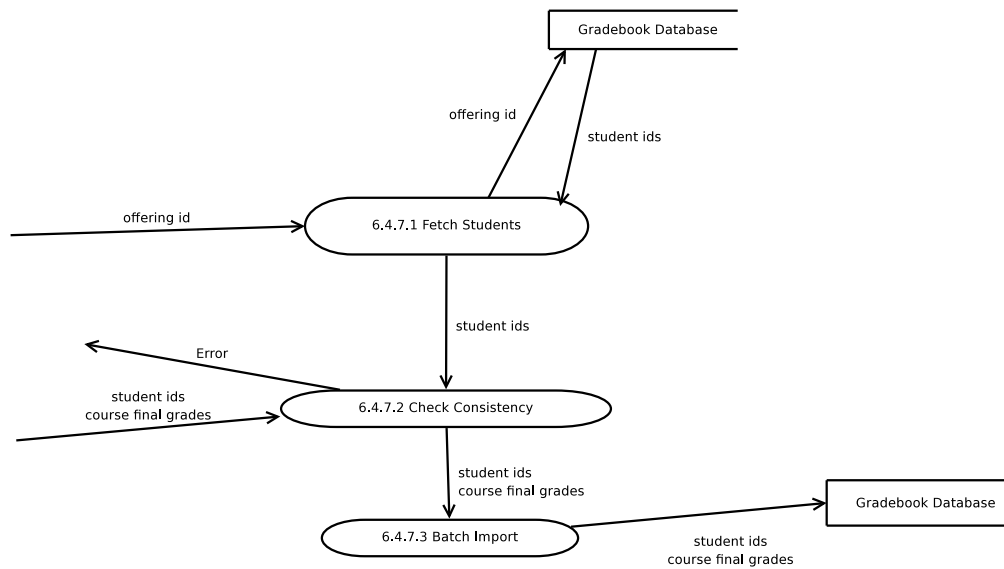


Figure 32: L3-6.4.7 Proposed System

In this diagram, the administrator imports final grades for an offering. Notice that the assignment grades are ignored even if they exist.

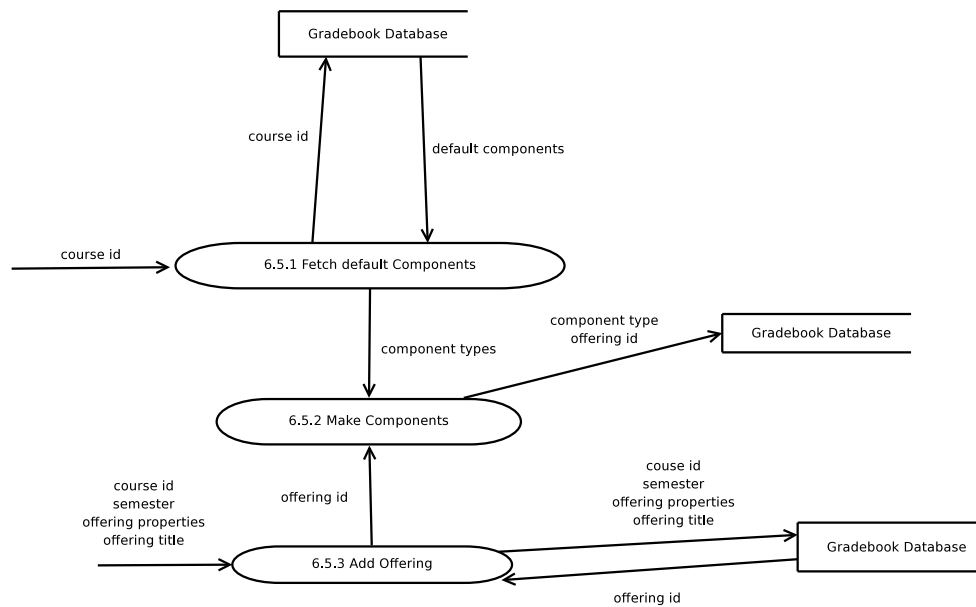


Figure 33: L2-6.5 Proposed System
In this diagram, the administrator adds a new offering for a course.

6.4 Interaction of ER and DF Models

In this section, interaction between the ER and DF models is explained in the form of the DFD process number and its corresponding set of entities in the ER diagram.

- | | |
|-----|---|
| 1 | Person |
| 1.1 | N/A |
| 1.2 | N/A |
| 1.3 | Person |
| 1.4 | Person |
| 2 | Offering, Section, Person |
| 2.1 | Offering, Person |
| 2.2 | Section, Person |
| 2.3 | Offering Person |
| 3 | Offering, Person, Component, Assignment |

- 3.1 Person, Component, Offering
- 3.2 Component, Assignment, Person
 - 3.2.1 Component, Assignment
 - 3.2.2 Assignment
 - 3.2.3 Component, Assignment
 - 3.2.4 Component, Person
 - 3.2.5 Component, Person, Assignment
 - 3.2.5.1 Component, Person
 - 3.2.5.2 Component, Assignment
 - 3.2.5.3 Person, Assignment
 - 3.2.5.4 Person, Assignment, Component
 - 3.2.5.5 Person, Assignment, Component
- 3.3 Person, Assignment, Component, Offering
 - 3.3.1 Offering, Component
 - 3.3.2 Component, Assignment
 - 3.3.3 Person, Component
 - 3.3.4 Person, Assignment
 - 3.3.5 Component, Person
 - 3.3.6 Offering, Person
 - 3.3.7 Person, Assignment, Component, Offering
- 3.4 Offering, Person
- 4 Assignment, Person, Component, Section
 - 4.1 Section, Person
 - 4.2 Section, Component
 - 4.3 Person, Assignment, Component
- 5 Offering, Component, Assignment, Person
 - 5.1 Offering, Component
 - 5.2 Component, Assignment

5.3	Person, Assignment
5.4	Component, Person
5.5	Offering, Person
5.6	Person, Assignment, Component, Offering
6	Offering, Course, Person, Component, Assignment, Section
6.1	Offering
6.2	Course, Offering
6.3	Course
6.4	Offering, Person, Component, Assignment, Section
6.4.1	Offering
6.4.2	Person, Offering
6.4.3	Offering, Component
6.4.4	Offering, Component
6.4.5	Component, Person, Assignment, Section
6.4.5.1	Component
6.4.5.2	Component, Person
6.4.5.2.1	Component, Person
6.4.5.3	Component, Person
6.4.5.4	Component, Person, Assignment
6.4.5.5	Component, Person
6.4.5.5.1	Component, Person
6.4.5.5.2	Component, Person
6.4.5.5.3	Component, Person
6.4.5.6	Component, Section
6.4.5.7	Component, Section, Person
6.4.5.8	Person, Section, Component
6.4.5.8.1	Section, Person, Component
6.4.5.8.2	Component, Section

6.4.5.8.3	Section, Person
6.4.5.8.4	Section, Person
6.4.5.9	Section, Person
6.4.5.9.2	Section, Person
6.4.6	Student, Assignment, Offering, Component
6.4.7	Offering, Course, Person
6.4.8	Student, Offering
6.5	Course, Component, Offering
6.5.1	Course
6.5.2	Component, Offering
6.5.3	Course, Offering
6.6	Course
6.7	Component, Section, Person

6.5 Functionality Requirements

The client requires the system to maintain the information that he is interested in but not to process them. Therefore all functionality requirements are related to the information content.

1. Record the ER model
The proposed system should be capable of recording the ER model of the client organization, as captured in 6.2.
2. Support and Maintain the ER model
 - (a) Student
 - i. View Student Grades
Information Source user ⋈ offering ⋈ component ⋈ assignment ⋈ assign_grade
Select Condition by user.id, by offering.id, max assign_grade.time
Sort Condition N/A
Other N/A
 - (b) Instructor:
 - i. Assign Section Grades
Insert grade, time, teacher_id, student_id, assignment_id
Information Source assignment_grade
 - (c) Professor:

- i. Assign Section Grades
Refer to Req.2(b)i
- ii. Add Assignments
Insert assignment_title, hidden=false
Information Source assignment
- iii. Modify Assignments
Update assignment_title, hidden=false
Select Condition by assignment_id
Information Source assignment
- iv. Hide⁴ Assignments
Update assignment_title, hidden=true
Select Condition by assignment_id
Information Source assignment
- v. View/Export Overall Course Grades⁵
Information Source offering ⋈ component ⋈ assignment ⋈ assign_grade (.receiver) ⋈ person
Select Condition by offering.id, max assign_grade.time
Sort Condition by component, by person, by assignment
Other N/A
- vi. View/Export Overall Component Grades
Information Source component ⋈ assignment ⋈ assign_grade (.student_id) ⋈ person
Select Condition by component.id, max assign_grade.time
Sort Condition by person, by assignment
Other N/A
- vii. View/Export Course Rosters
Information Source course ⋈ course_person ⋈ person
Select Condition by course.id
Sort Condition by person
Other N/A
- viii. View/Export Component Rosters
Information Source component ⋈ component_person ⋈ person
Select Condition by component.id
Sort Condition by person
Other N/A

(d) Administrator (Information Management)

- i. Add Courses

⁴To hide is to make inactive

⁵A Course Grade is actually an Offering Grade. The terminology is so widely used in the client's daily practice that the team decided to preserve it so as to avoid confusion in the client side.

- Insert** course_name,
 - Information Source** course
- ii. Modify Courses
 - Update** course_title, default_components
 - Select Condition** by course_id
 - Information Source** course
- iii. Add Course Offerings
 - Insert** course_id, offering_title, offering_properties
 - Information Source** offering
- iv. Modify Course Offerings
 - Update** offering_title, semester, professor_id
 - Select Condition** by offering_id
 - Information Source** offering
- v. Assign Professors to Offerings
 - Integrated in Req.2(d)iv
- vi. Add Students to Offering
 - Insert** student_id, offering_id
 - Information Source** student_offering
- vii. Remove(Hide) Students from Offering Rosters
 - Update** student_id, hidden=true
 - Select Condition** by student_id, offering_id
 - Information Source** offering
- viii. Add Components
 - Insert** offering_id, component_type
 - Information Source** component
- ix. Hide Components
 - Update** component_type, hidden=true
 - Select Condition** by component_id
 - Information Source** component
- x. Modify Components
 - Update** component_type, hidden=false
 - Select Condition** by component_id
 - Information Source** component
- xi. Add Assignments
 - Refer to Req.2(c)ii
- xii. Modify Assignments
 - Refer to Req.2(c)iii
- xiii. Hide Assignments
 - Refer to Req.2(c)iv
- xiv. Move Students between Sections

1. Information Source Section ⇨ Component

Select Condition by student_id, by start_date, by component_id

2. Insert start_date, end_date, student_id, section_id

Information Source section

xv. Add Sections

Insert section_name, component

Information Source section

xvi. Modify Sections

Update section_name, hidden=false, instructor_id

Select Condition by section_id

Information Source section

xvii. Hide Sections

Update section_name, hidden=true, instructor_id

Select Condition by section_id

Information Source section

xviii. Assign Instructors to Sections

Integrated in Req.2(d)xvi.

xix. Assign Grades

Refer to Req.2(b)i

xx. View/Export Overall Course Grades

Refer to Req.2(c)v

xxi. View/Export Overall Component Grades

Refer to Req.2(c)vi

xxii. View/Export Course Rosters

Refer to Req.2(c)vii

xxiii. View/Export Component Rosters

Refer to Req.2(c)viii

xxiv. Import Final Course Grades

Update final_grade

Select Condition by offering_id, by student_id

Information Source offering_student

xxv. Import Final Component Grades

Update final_grade

Select Condition by component_id, by student_id

Information Source component_student

6.6 Business Rules

In this section, the business rules in the client organization are listed. They are classified by the roles in the client organization.

- For Students

- At any given time, a student can participate in at most one section within each component.
- A student cannot view the roster of the course, component or the section.
- A student cannot view the grades of any other students.
- For Instructors
 - An instructor cannot view grades associated with the students belonging to sections other than the ones that are assigned to the instructor.
 - An instructor can only view the roster of the sections that are assigned to that instructor.
 - An instructor can only assign/modify grades of students for sections that are assigned to that instructor.
- For Professors
 - A professor can only view and modify assignments, roster and grades within courses that are taught by that professor.
- Miscellaneous Constraints
 - A student can be an undergraduate instructor.
 - A student from a previous semester can enroll for the same course in a following semester to finish incomplete work in a subset of the components.

6.7 Maintenance Requirements

Maintenance requirements involved in the project cover the following two aspects:

- Backup
 - The software platform will provide the backup mechanism in the MySQL database management system.
 - The client will come up with a backup policy.
- Clean-up
 - The system will provide a mechanism to hide outdated data entries.
 - The client will come up with a clean-up policy.

6.8 Qualitative Requirements

In this section the qualitative requirements, which do not directly correspond to operations or actions are described. As indicated in Section 3, the client organization requires a specifically designed system to precisely represent the unique course structure and to efficiently manage the grades assigned to the students over the years. To achieve the goal, the system should meet the following qualitative requirements:

- Efficiency

- The grades will be managed efficiently.
- The course structure will be organized and precisely represented by the system.
- The client will not need any workarounds.
- The interface will be concise and precise for all users.
- Portability
 - The users will be able to access the system from various software and hardware platforms.
 - The system itself can be deployed to various hardware platforms.
- Reliability
 - The system will not crash on software bugs but fail gracefully.
 - Hardware failures and software platform failures will be recoverable.
- Security
 - Grades of individual students will be kept private. Individual students would only be able to view their own grades.
 - Without appropriate permissions, users would not be able to access any data in the system.

6.9 User Interface

The proposed system is virtually made up by two subsystems, the Gateway subsystem for regular users; and the management system(Administrator Gateway) for the administrator. Only key screens of the user interface design are presented in this document.

6.9.1 Conventions

The user interface follows the conventions of web pages:

- Hyperlinks for switching screens.
- Buttons for posting a request to the system.
- Tab key for switching between Hyperlinks, Buttons and other HTML form elements in page layout sequence.
- The back button of the browser can cancel an unposted request.

6.9.2 Login

The Login screen is where a user logs into the system. Depending on the type of user, the screen is redirected to either 6.9.3 or 6.9.8 after authentication.

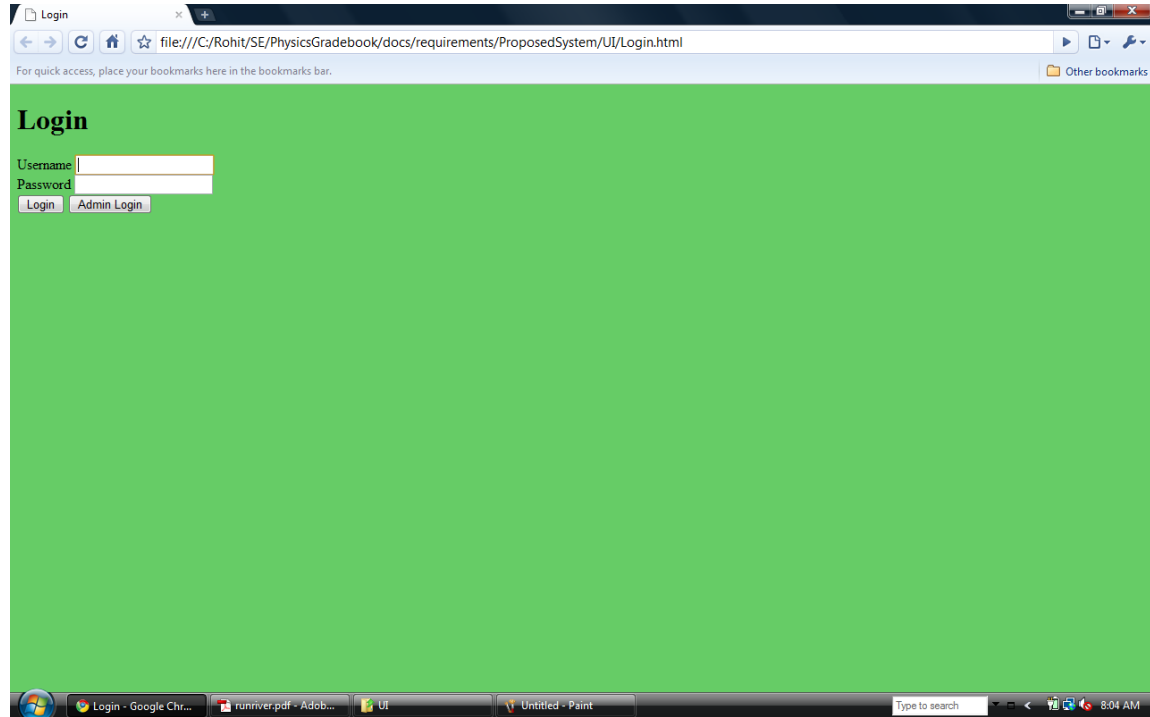


Figure 34: User Interface for Login

Event	Action
Click Login Button	Go to 6.9.3
Click Admin Login Button	Go to 6.9.8

Table 1: Login Screen

6.9.3 Gateway

The Gateway screen is the gateway for students, instructors and professors to access the system. The content in this screen depends on the roles the user plays in the system. Three sections are displayed for Instructor, Professor and Student.

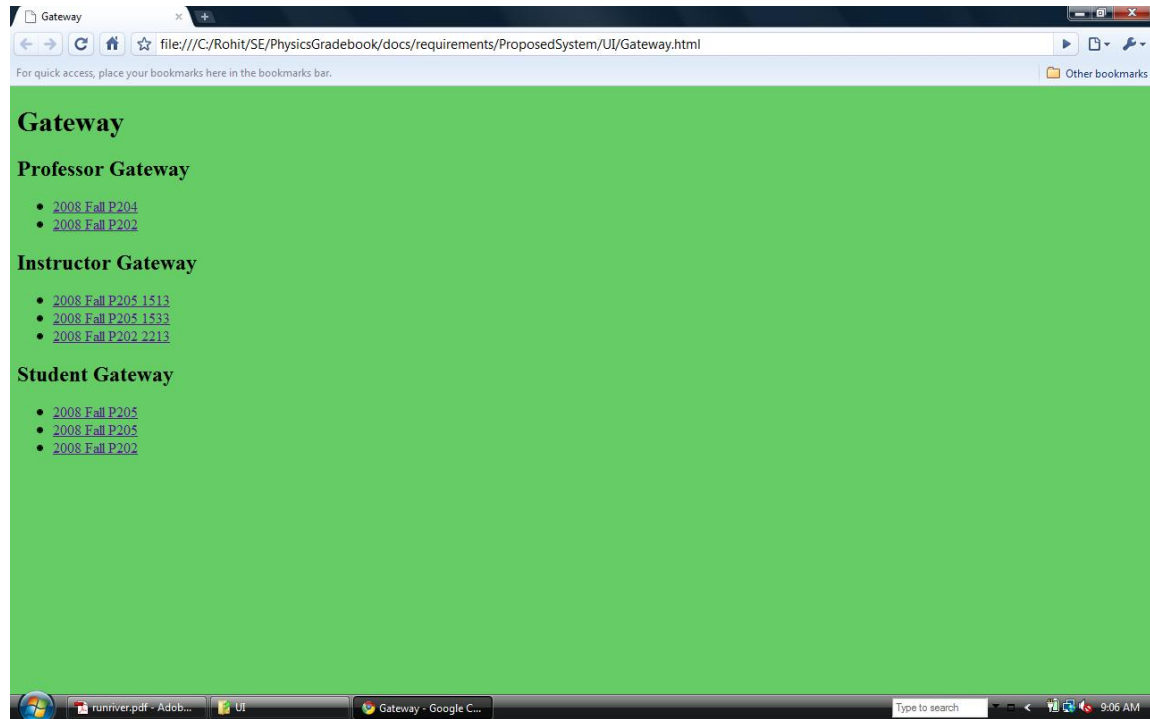


Figure 35: Gateway Screen for Regular Users

Event	Action
Click a Hyperlink in Professor Gateway	Go to 6.9.5
Click a Hyperlink in Instructor Gateway	Go to 6.9.7
Click a Hyperlink in Student Gateway	Go to 6.9.4

Table 2: Gateway Screen for Regular Users

6.9.4 Offering - Student

A student views his grades for all the assignments and final grades in any given offering of a course.

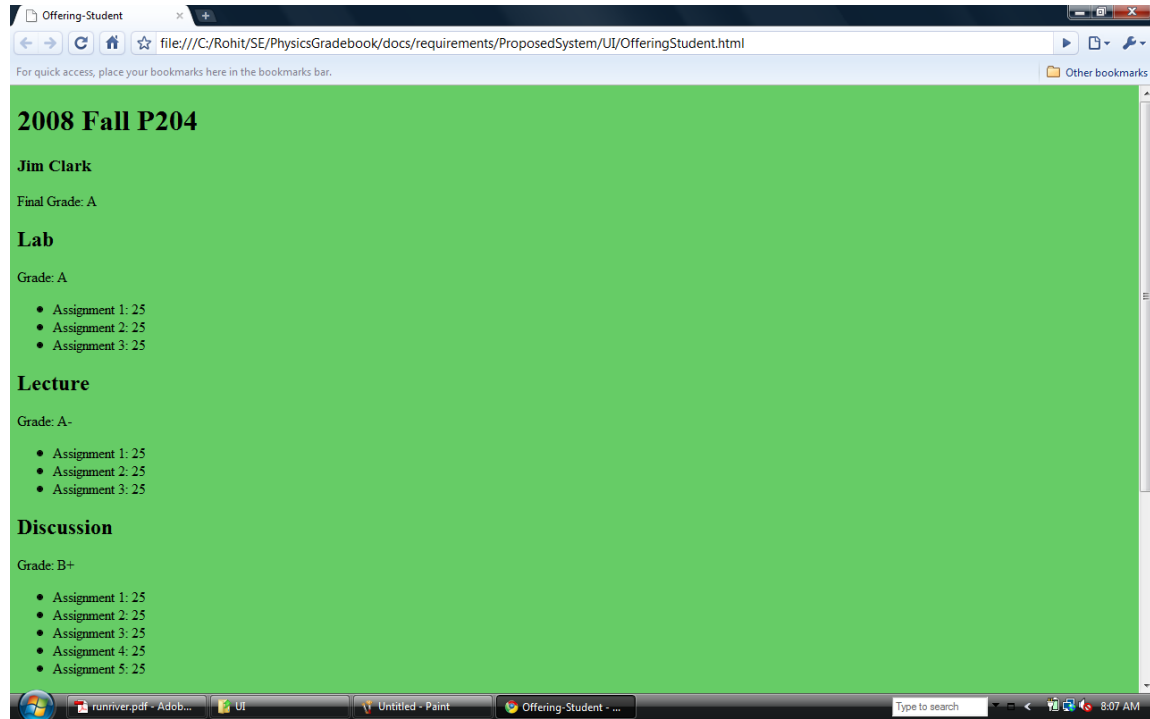


Figure 36: Offering Screen for a Student

Event	Action
N/A	N/A

Table 3: Offering Screen for a Student

6.9.5 Offering - Professor

A professor manages the assignments and grades in an offering with this screen.

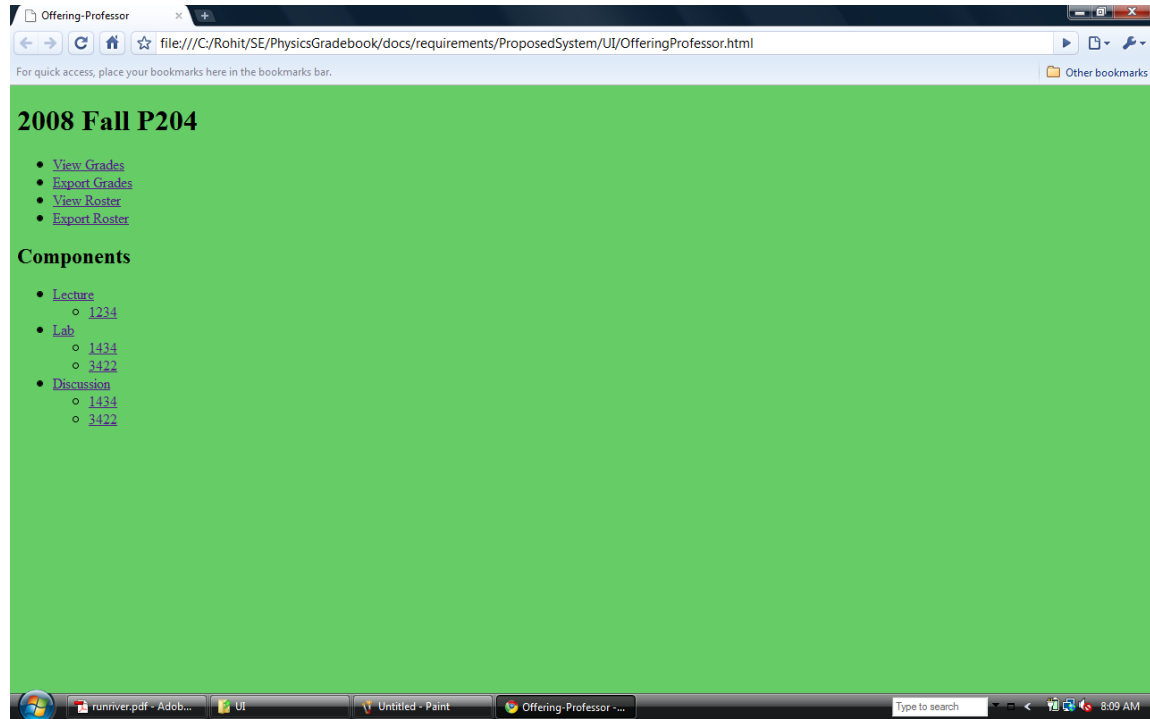


Figure 37: Offering Screen for Professor

Event	Action
Click View Grades	Show a list of all grades in the offering
Click Export Grades	Export a list of all grades in the offering
Click View Roster	Show the roster of the offering
Click Export Roster	Export the roster of the offering
Click Component Hyperlink	Go to 6.9.10
Click Section Hyperlink	Go to 6.9.7

Table 4: Offering Screen for Professor

6.9.6 Component - Professor

A professor manages the assignments and grades in a component with this screen.

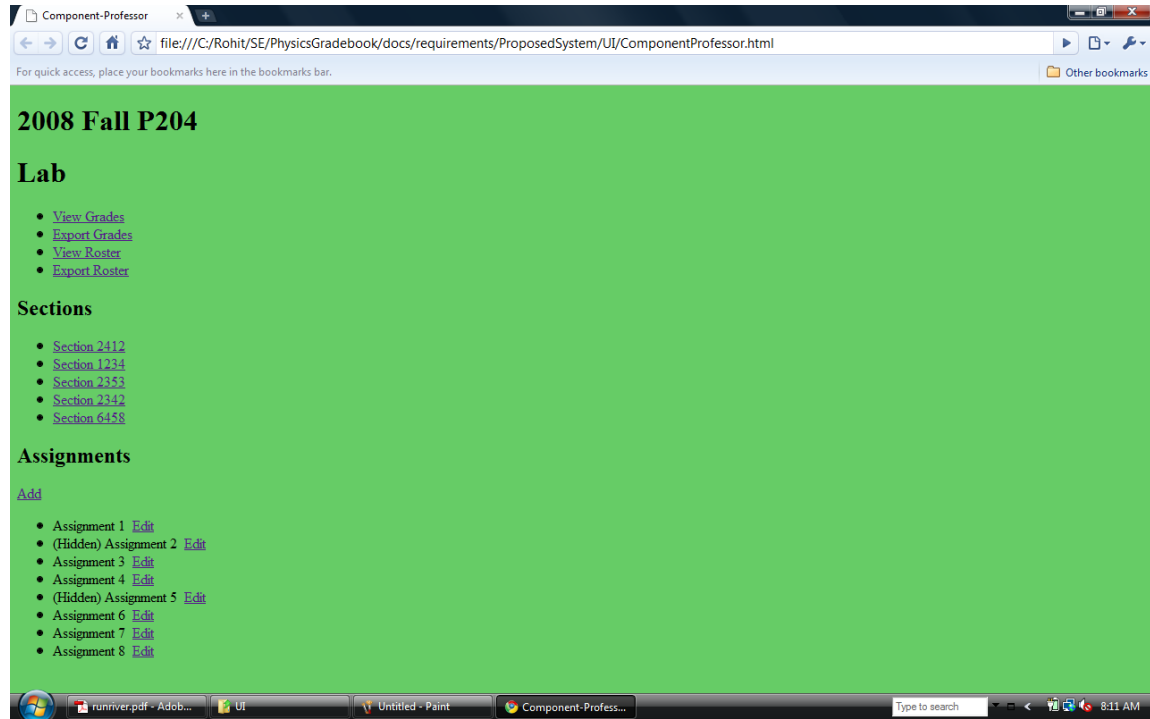


Figure 38: Component Screen for Professor

Event	Action
Click View Grades	Show a list of all grades in the component
Click Export Grades	Export a list of all grades in the component
Click View Roster	Show the roster of the component
Click Export Roster	Export the roster of the component
Click Add	Add a new assignment
Click Edit	Modify an existing assignment
Click Section Hyperlink	Go to 6.9.7

Table 5: Component Screen for Professor

6.9.7 Section - Instructor

An instructor or professor manages the grades in a section with this screen.

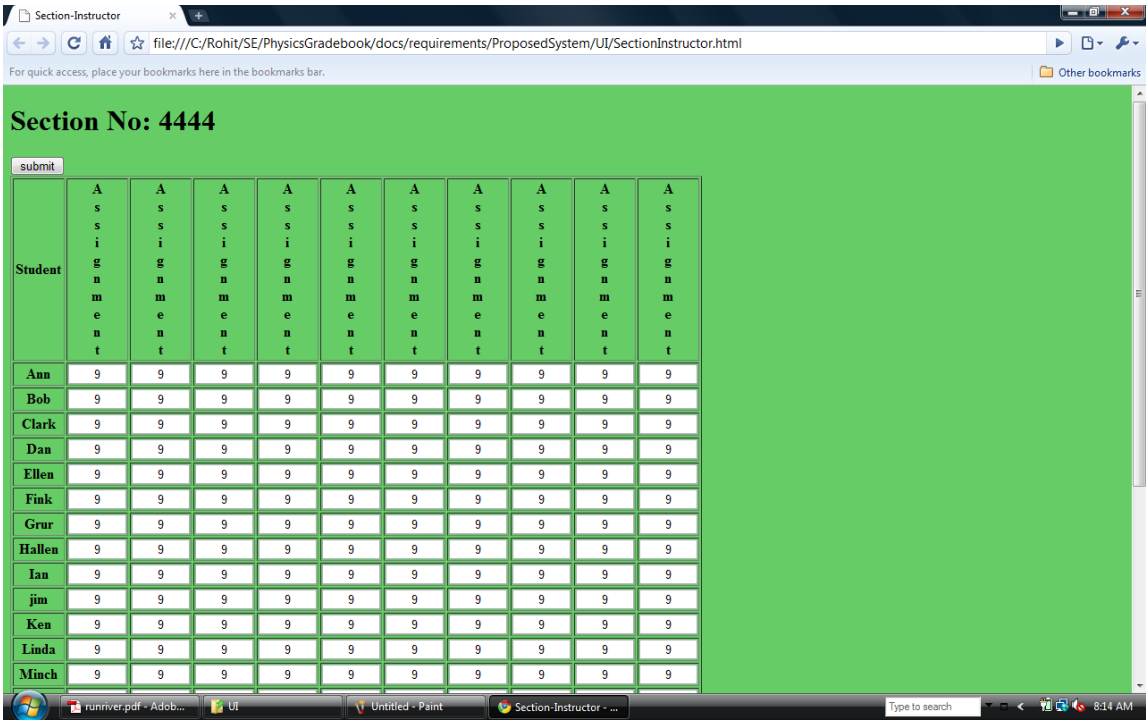


Figure 39: Section Screen for Instructor

Event	Action
Click Submit	Update the grades of students for assignments

Table 6: Section Screen for Instructor

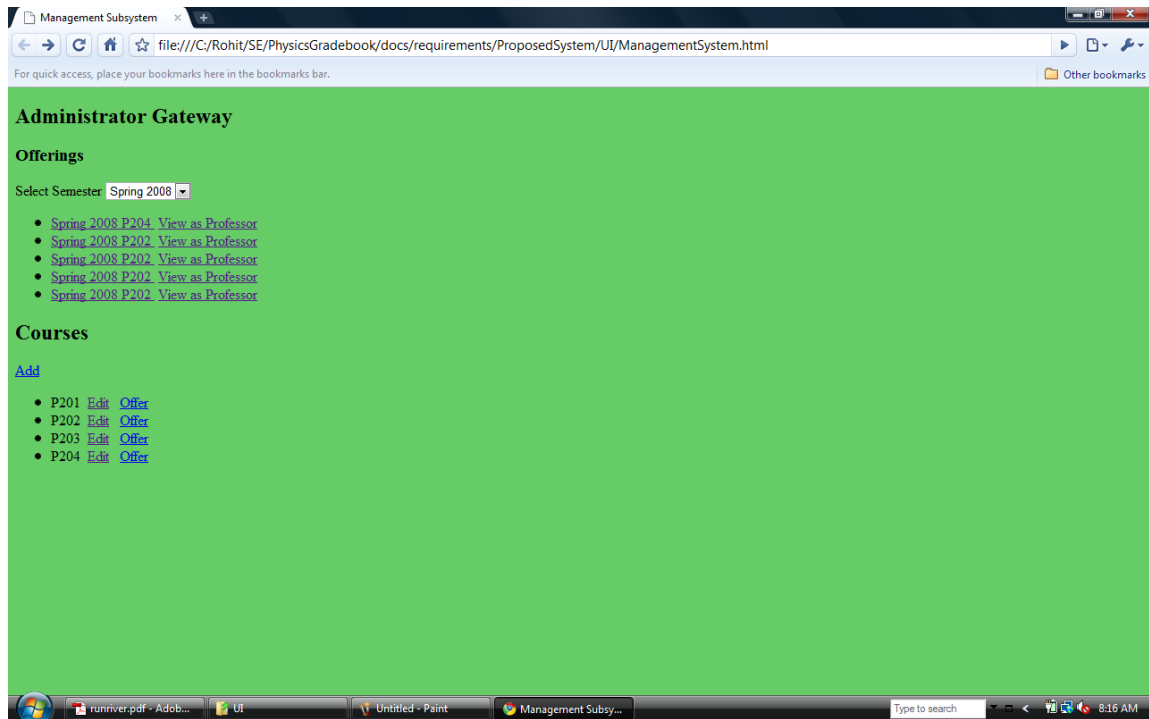


Figure 40: Administrator Gateway Screen

6.9.8 Administrator Gateway

Administrator Gateway screen is the screen for the management system.

Event	Action
Click Offering Hyperlink	Go to 6.9.9
Click Add	Add a new Course
Click Edit	Modify an existing Course
Click Offer	Offer a course in a semester

Table 7: Administrator Gateway Screen

6.9.9 Offering - Administrator

The administrator manage an offering in this screen.

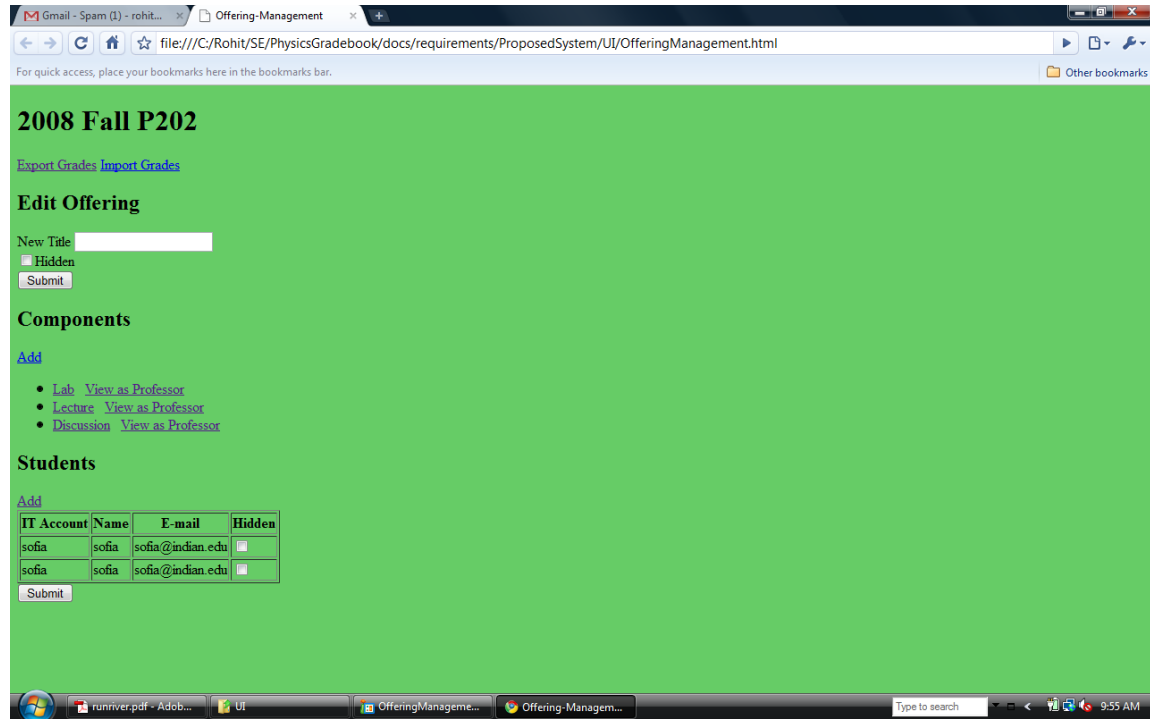


Figure 41: Offering Screen for Administrator

Event	Action
Click Export Grades	Export grades
Click Import Grades	Import grades
Click Hidden checkbox in Edit Offering	N/A
Click Submit button in Edit Offering	Modify existing offering
Click Component Hyperlink	Go to 6.9.10
Click Add in Components	Add new component to a course
Click View as Professor	Go to 6.9.5
Click Add in Students	Add a student to the offering
Click checkbox under Hidden column in Students	N/A
Click Submit button in Students	Hide student from roster of the offering

Table 8: Offering Screen for Administrator

6.9.10 Component - Administrator

The administrator manages a component in this screen.

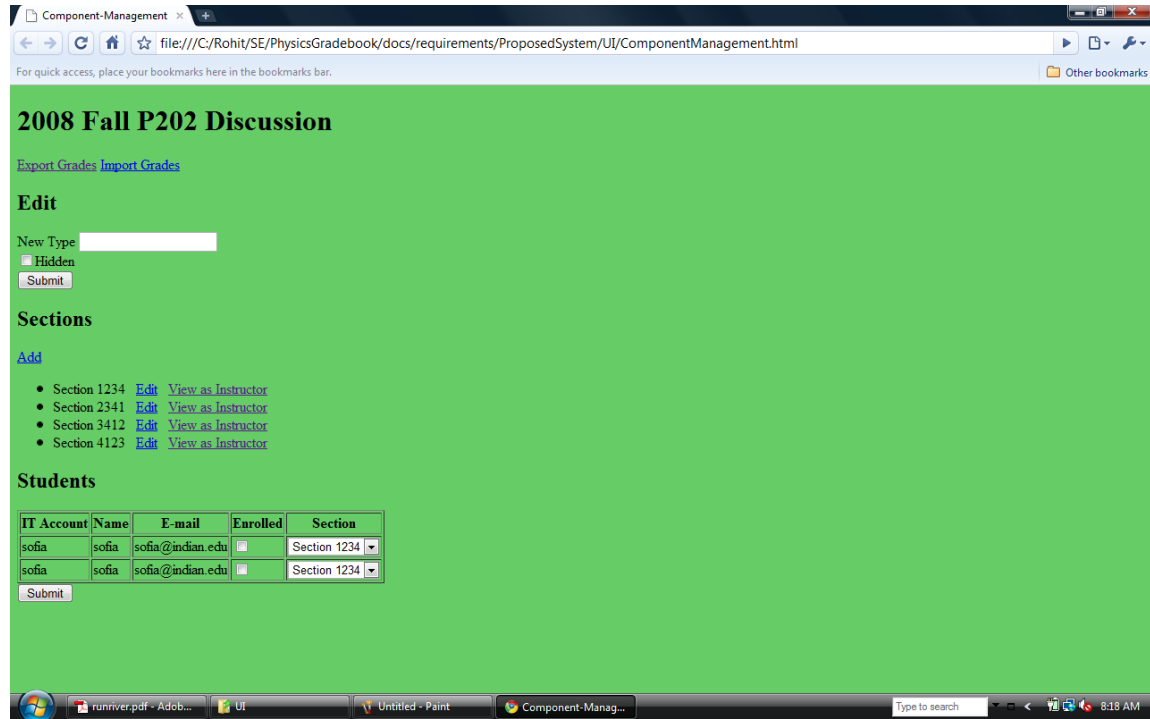


Figure 42: Component Screen for Administrator

Event	Action
Click Export Grades	Export grades
Click Import Grades	Import grades
Click Hidden checkbox in Edit Component	N/A
Click Submit button in Edit Component	Modify this Component
Click Edit in Sections	Modify a Section.
Click Add in Sections	Add a new Section to this Component
Click View as Instructor	Go to 6.9.7
Click Add in Students	Add a student to the offering
Click Checkbox Students	N/A
Click an Item in the Dropdown List in Students	N/A
Click Submit button in Students	Change the section and enrollment of students

Table 9: Component Screen for Administrator

6.10 Installation

The proposed system is a complete replacement of the current system. The client will immediately switch to the proposed system when deployed. A transition phase is not needed, which has two implications:

- The data from the current system is not going to be transported to the proposed system.
- A period of concurrent operation is not expected.

The users of the system are expected to change each semester. The team will prepare the primary client on the usage of the entire system. The primary client is responsible for training other users based on roles for the relevant aspects of the system at the beginning of each semester.

6.11 Limitations

The proposed system will not implement the following functionalities:

- Calculating Final Grades
- Reporting Final Grades to the Registrar Office

The secretaries in the Physics Department and the primary client will implement these functionalities manually as processes external to the proposed system.

7 Quality Assurance Plan

7.1 Purpose and Scope

This Quality Assurance Plan (QAP) provides a framework for developing an effective, reliable, and maintainable software system for the IU Physics Gradebook System project. The QAP requires that certain procedures be followed and certain documents be written during the course of the project. The QAP is written in general accordance with the IEEE standard STD 730-1984 [3].

This QAP covers the the user interfaces of the IU Physics Gradebook System. The system is used by the Physics Department of Indiana University for keeping the grades for its students over years.

7.2 Organization, Tasks, and Responsibilities

The project requires the following designated roles:

- Quality Assurance Monitor (QAM) - Naga Rekha Malae
 - The QAM is appointed for the project by the course management. The role of the QAM will be:
 - * To ensure that this QAP is in accordance with the general guidelines.
 - * To ensure that the project is in compliance with its plans.
- Program Librarian (PL) - Yu Feng
 - The PL will maintain the Released Software Library (see 7.6).
- Quality Assurance Recorder (QAR) - Rohit Chandran
 - The QAR will maintain the Change Log (see 7.2) and Software Verification Summary (see 7.8).

7.3 Documentation Required

The following documents are required for the project:

- Requirements Specification
- Quality Assurance Plan
- Preliminary Design Description Test Plan (Software Verification/Validation and Acceptance Plan)
- Detailed Design Description
- User's Documentation
- Programmer's Reference Manual
- Change Log
- Software Verification Summary
- Computer Program Development Plan

The Requirements Specification, Preliminary Design Description, Quality Assurance Plan, Detailed Design Description, and Test Plan are the documents produced at project Milestones 2, 3, and 4 [2].

The User's Documentation must include the User's Reference Manual, the training for the users will be done by the Client.

The Programmer's Reference Manual must include sections corresponding to:

- a global static description, with
 - entity-relationship diagrams
 - data dictionary
 - procedure hierarchy charts
- a global dynamic description, with
 - Data Flow Diagrams (DFD)
- per module documentation, with
 - structure charts {HIPO, Warnier-Orr, or equivalent}

The Programmer's Reference Manual must provide explicit connections from items in the specifications and requirements through the logical and physical design to implementation.

The Change Log will document all changes to the final product that affect milestones already past. The Change Log must track the effects of these changes to all past milestones. As appropriate for the span of changed milestones, entries in the Change Log must include sections on requirements, specifications, design, and implementation. The Change Log will also record any reported problems with accepted modules and the corrective actions taken for these problems (see 7.7).

Each entry in the Change Log shall include

- Date reported - the date when change was requested or problem observed
- Originator - the individual(s) who requested the change or observed the problem reported
- Condition - the reasons for the change or symptoms of the problem date
- Action taken - when the change was implemented or corrective action completed description of action taken
- Components affected - milestones, modules, etc. affected by the action
- Implementer - the individual(s) who took the change/corrective action
- Tests - description of the tests or verification/validation taken with respect to this action

The Software Verification Summary is discussed in 7.6, 7.9 and 7.12.

7.4 Standards, Practices, and Conventions

All programs and modules developed in the course of this project must conform to standards on code development, organization, and internal documentation.

7.4.1 Program Standards

- Strict convention should be followed
- All variables should be at least 4 characters long
- Number of lines in a procedure should be limited
- Number of procedures in a file should be limited
- If Object Oriented method is used in programming, number of methods in a class should also be limited
- The interface between modules should be minimized
- The use of global variables should be minimized

All programs and modules developed in this project will be subject to unit test, acceptance test, and verification and validation according to the procedures described in the Software Verification/ Validation and Acceptance Plan. The Software Verification/Validation and Acceptance Plan will contain the following information: Requirements to be verified Software-code test plan Type of test: function, performance, stress, structure Test assumptions Requirements being tested Test cases used Expected outcome Test completion criteria Acceptance criteria Document verification plan (e.g., user's manual, programmers manual, internal documentation) System integration plan

The general requirements for testing, verification, and validation include the following:

- All modules must undergo unit test before being included in the library of released software (see 7.6).
- The author or authors of a module may not be wholly responsible for unit testing of the module. In particular other team members must participate in defining the tests and evaluating the test results and may, if appropriate, be required to participate in the running of the tests.
- An Acceptance Test, as specified in the Software Verification/ Validation and Acceptance Plan, must be conducted before the completed system is delivered to the client.
- All documents produced in the course of this project must be verified prior to project completion.
- A Software Verification Summary shall be maintained to record all tests and verification/validation steps and the results of these actions. This Software Verification Summary shall be maintained by the QAR (see 7.2).

7.5 Reviews and Audits

The following reviews and audits will be conducted:

- Requirements Specification Review
- Design Review
- Testing Audit

The Reviews will occur during the development of the respective milestone documents. The Reviews will cover a draft of those documents and will be attended by other teams and members of the course management.

The Testing Audit will be conducted by the QAM as part of the product acceptance process. The primary activity of the Testing Audit will be the review of the Software Verification Summary to insure that all required tests and verification steps have been successfully completed. The Change Log will also be reviewed to insure that all issues are resolved.

7.6 Configuration Management

Configuration management must be controlled by the PL (see 7.2) who will maintain a Released Software Library. A software module is "released" after it has passed the unit tests specified in the Software Verification/Validation and Acceptance Plan (see 7.4). All development and testing on a given module must use other modules from the Released Software Library only.

For configuration management, subversion is used. The branching and tagging facilities of subversion meet the configuration management requirements.

Notification emails are sent for new software releases and the release history is captured in a release note.

7.7 Problem Reporting and Corrective Action

All problems with released software modules and any corrective action related to these problems must be recorded in the Change Log (see 7.3).

7.8 Tools, Techniques, and Methodologies

- Project progress control is done with the help of weekly logs made by the secretary, Rohit and provided to the supervisor.
- Project action follow-up is also done with the help of weekly logs made by the secretary, Rohit and provided to the supervisor.
- Documentation is done and internally edited with the use of Latex.

7.9 Code Control

Configuration management will include keeping the Released Software Library, which contains standard, current-version software (see 7.6).

- The subversion facility provided by Google Code is used to ensure that all modifications are identifiable and traceable.
- The development work is distributed on the team members' computers.
- The released modules will be immediately deployed on a machine provided by the primary client in the Physics Department.
- The source code of the standard libraries will reside on Google Code, network-accessible from the stations used in actual development. This will provide all team members with consistent access to the latest versions of released software.

7.10 Media Control

Team members would hold their own working copy of the source code. This itself would act as a backup. If Google Code is down for a long period of time, a repository can be reconstructed from any of the the three working copies.

7.11 Supplier Control

Supplier control is not applicable to this project.

7.12 Records Collection, Maintenance, and Retention

The QAR must maintain the Change Log (see 7.3) and Software Verification Summary (see 7.8).

8 Appendices

- The Project Plan is provided as a separate page at the end of the document.

9 Client Comments

1. Introduction: A mention of security from external hackers might be comforting to the end user.

Figure 1. Client Business model: discussion, lab, etc are generic names. Does not mean that there cannot be more components and does not mean there will only be one gradebook per component. One of our problems is the huge flat file we get when we print or display all the grades for the lecture which includes a hundred or so clicker questions, 200 or so homework problems and exam grades. Multiple gradebooks, or at least the means to split the gradebooks into manageable chunks for viewing and entry may be needed.

4.1.3 The course professor only rarely enters grades for the lecture. Usually homework, test grades etc are entered by the course secretary or grading AI.

4.4.5 Administrator role. No mention that the administrator also has all the features of the course professor. Otherwise how do secretaries enter grades? We don't want the students thinking the secretary is the course professor or instructor. Is this where the title thing comes in?

I would like some clarification on why the student, instructor and professor roles are weak but not the administrator role. Not clear what this means hence I am not clear on the impact this has on the software.

Pg 15, figure 14 - why do we authenticate via CAS for other roles but the administrator role doesn't? Seems to me a good protection would be to authenticate everyone. Perhaps we can have a local password, good only from the machine where the database resides so info can be accessed if the network is down?

Page 17, figure 17. Is there no grade import function for AIs/instructors? If not perhaps we need a stub in here in case we want to add the feature in the future? Same for automatic grade calculations under very restrictive conditions.

Pg 20, figure 20. Keep in mind a grader may have 500 or so grades to enter for each assignment. We need a requirement that says the grades can be entered via a list box or whatever it is called so all he/she has to do is scroll the list up or down, find the name and enter the grade. The assignments listed singly as a column or listed as a 2d spreadsheet. Reminder/prompt to save changes on exit rather than as they are individually entered.

Page 21: I am not at all sure whether we currently need the final grade function. Perhaps in the future but for now the final grade can be just another assignment column (I call it "current grade").

I don't see anywhere in the requirements that says the database is alpha and numeric. This is the prime directive, not just a requirement.

Pg 26, fig 28. Import final grades. It is not clear why the administrator ended up with this function. If the course prof can see them online and can download them, why does the administrator need a final grade export function?

I am beginning to see the light in terms of component versus offering but it is still not at all clear.

I see one additional requirement - not a have to have but would make my life considerably easier. A report that shows the number of students enrolled in each section/component of a course.

Pg 38 and 39, 40. Again, not absolutely required but having a small HTML area or even just an area to post a jpeg so announcements/advertising could be made. Danger is the administrator and some profs will have a lot of information on their log in screens. The KISS principle undoubtedly applies here - just giving you a rope to hang yourself on by adding requirements leading to developer overload and cost over-runs. Good if you are PeopleSoft, not good if you are paying for it (and you are :)

I think the too big gradebook issue/multiple gradebook per section is not yet adequately addressed. Otherwise I think it is great work and I am looking forward to seeing the product develop. Quite good work overall.

Glossary

Term	Definition
Apache	Computer program that is responsible for accepting HTTP requests from web clients, which are known as web browsers, and serving them HTTP responses along with optional data contents, which usually are web pages such as HTML documents and linked objects (images, etc.).
Assignment	Homework given to students enrolled in the components. It could also be a test. In our project, an assignment means just the name of the assignment.
Course	Offered to students as part of the curriculum or program.
Component	Division of a course.
Central Authentication Service	A login service that allows access to multiple password-protected web systems after logging in once on a central authentication server; this is often referred to as single sign-on.
Data Flow Diagram	Graphical representation of the "flow" of data through an information system.
Entity-Relationship Model	Relational schema database modeling method used in software engineering to produce a type of conceptual data model (or semantic data model) of a system, often a relational database.
MySQL	Relational database management system (RDBMS). The program runs as a server providing multi-user access to a number of databases.
OnCourse	Indiana University's online collaboration and learning environment, powered by the Sakai community, supports teaching and learning, committees, projects, research, and portfolios for Indiana University's community of students, faculty, and staff.
PHP	Scripting language, originally designed for producing dynamic web pages.
Section	Division of a component
Web Hosting Service	Type of Internet hosting service that allows individuals and organizations to make their own website accessible via the World Wide Web.

References

- [1] Feasibility Study for Physics Gradebook System.
- [2] P465-6 & P565-6 Software Engineering for Information Systems I & II: Information Packet, Computer Science Department, Indiana University, 2008.
- [3] IEEE Standard for Software Quality Assurance Plans (STD 730-1984), Inst. of Electrical and Electronics Engineers, New York, 1984.
- [4] Requirement Specifications of Sakai Gradebook. <https://source.sakaiproject.org/svn/gradebook/trunk/xdocs/specs23/index.htm>
- [5] Information about Integrating CAS with a website. <http://kb.iu.edu/data/atfc.html>