Mohamed Soudy
Anas Halbawi

Web Application Development – Project

Grade Center

Part 1: Project Description

Students at Carnegie Mellon are constantly trying to figure out their grades for courses, what they will end up with and calculate their final grades based on their current, best or worst case performance. This will require the students going back to their professors which can be a headache for both sides. Therefore, we thought that we can automate this issue using a web application that can carry out all these calculations. Our web application will consist of two different types of accounts, a student account and a professor account. The professor will simply input the current grades of the student (through his own professor's account), and then the student can log in to his student account and look at graphs (automatically created) that reflect his current grade and future projections. To maximize the experience for both the student and the professor on the application, we will introduce a couple of features for both users that will be explained in the latter section of this proposal.

Part 2: Project Implementation

Team members and division of work:
Anas Halbawi - will implement the student's account (including the server side and client side)

Mohamed Soudy - will implement the professor's account (including the server side and client side)

Web technologies and frameworks:
Django will be used to implement the framework
Google Chart tools to implement the graphs
WebISO Secure Login for student/professor verification

Mohamed Soudy
Anas Halbawi

List of features that will be completed:

Student profile:

1. Course list with current grades
2. Class standing (where you lie in the class according to the class average)
3. Courses enrolled in (current courses with grade/next semester courses without grade)
4. Course Work (Assignments added by professors are shown here).
5. A GPA Calculator is provided for the student that can calculate his current GPA (according to his current grades which is done automatically), or the student can input his predicted grades and then select the respective course from a drop-down list.
6. Course grade statistics (Students can see more details about their grades, through different graphs, shown in the course page).
    a. Types of graphs
        i. Grade projections
            1. If student aces all upcoming grades (Best case)
            2. If student performs the same on the upcoming grades
            3. If student gets all zeros (Worst case)
        ii. Grade distribution of certain quiz/exam/assignment
        iii. Graph showing average of class

Professor Profile:

1. Courses being taught by professor
2. Grade uploader (for each student)
3. Grade statistics (Graphs, which includes graphs that can be accessed by professor only or students and professors)
4. Grading method (Choose between cumulative or percentage grading)
5. Course Work section that the professor can use to add assignments/quizzes/exams which will automatically show up on the students' course work section who are taking that course.

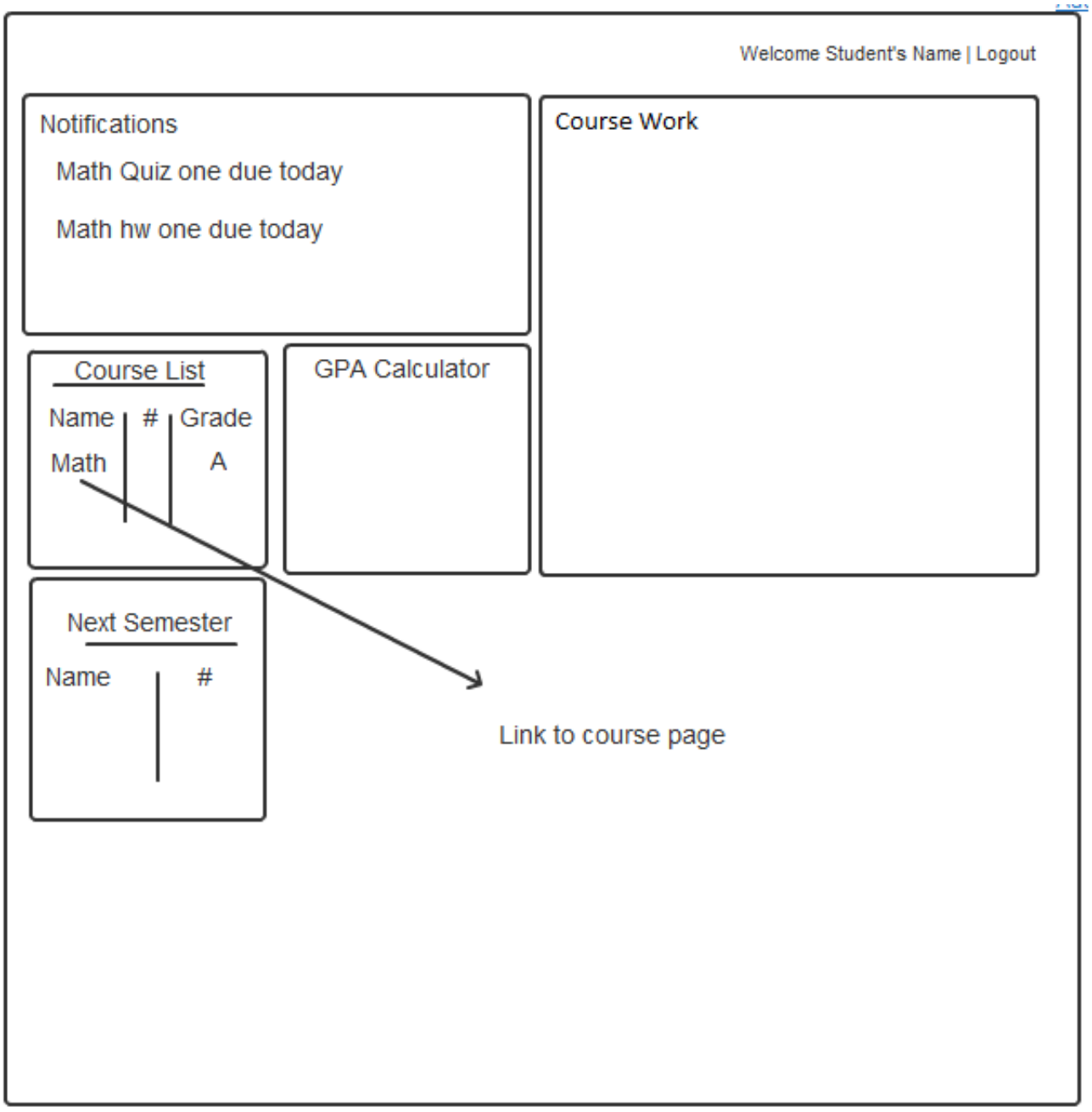
Course Page:

1. Course Description.
2. Grade of students accessing the course page, including detailed statistics (explained above).

Mohamed Soudy
Anas Halbawi

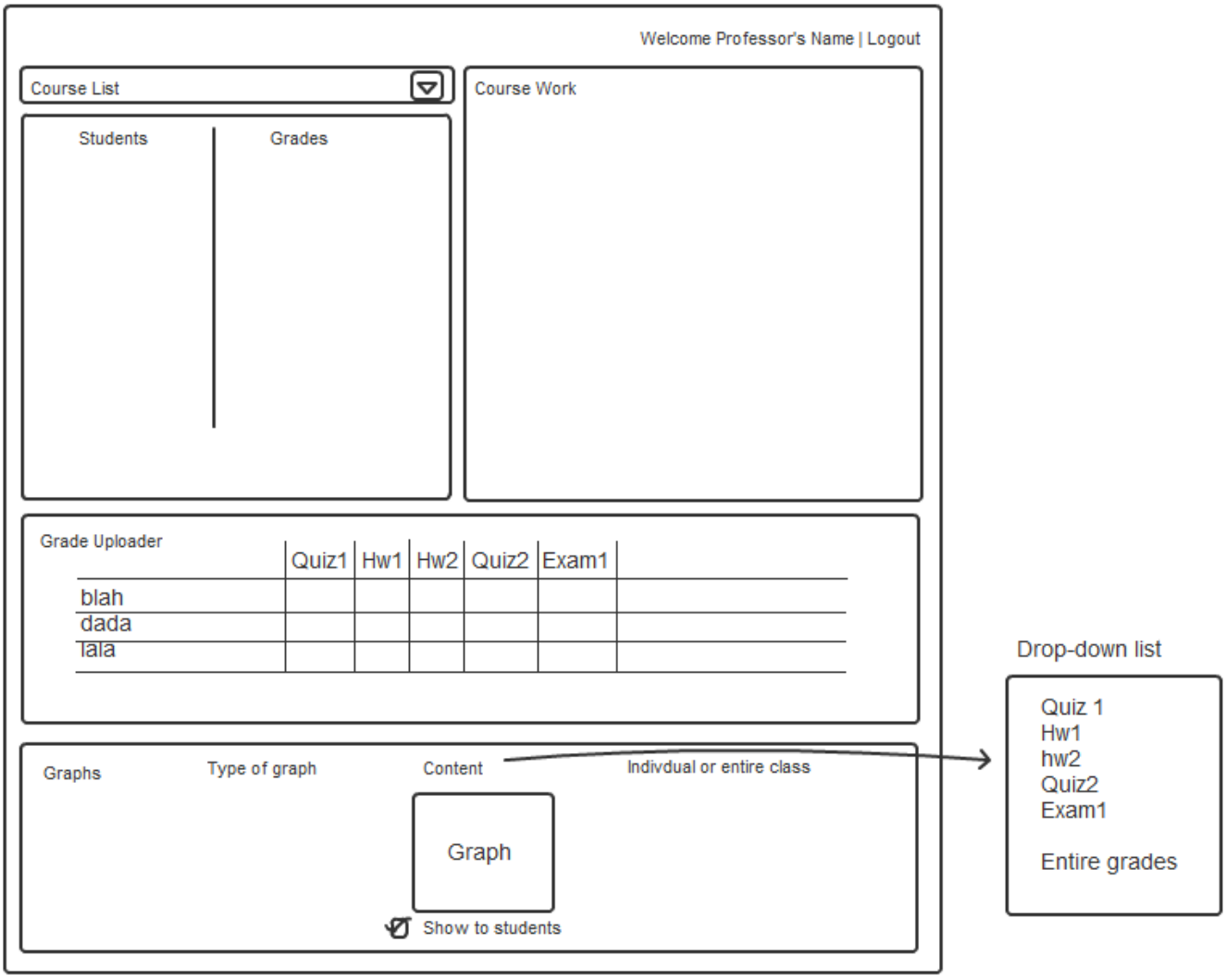
Part 3: User Interface

Student's account



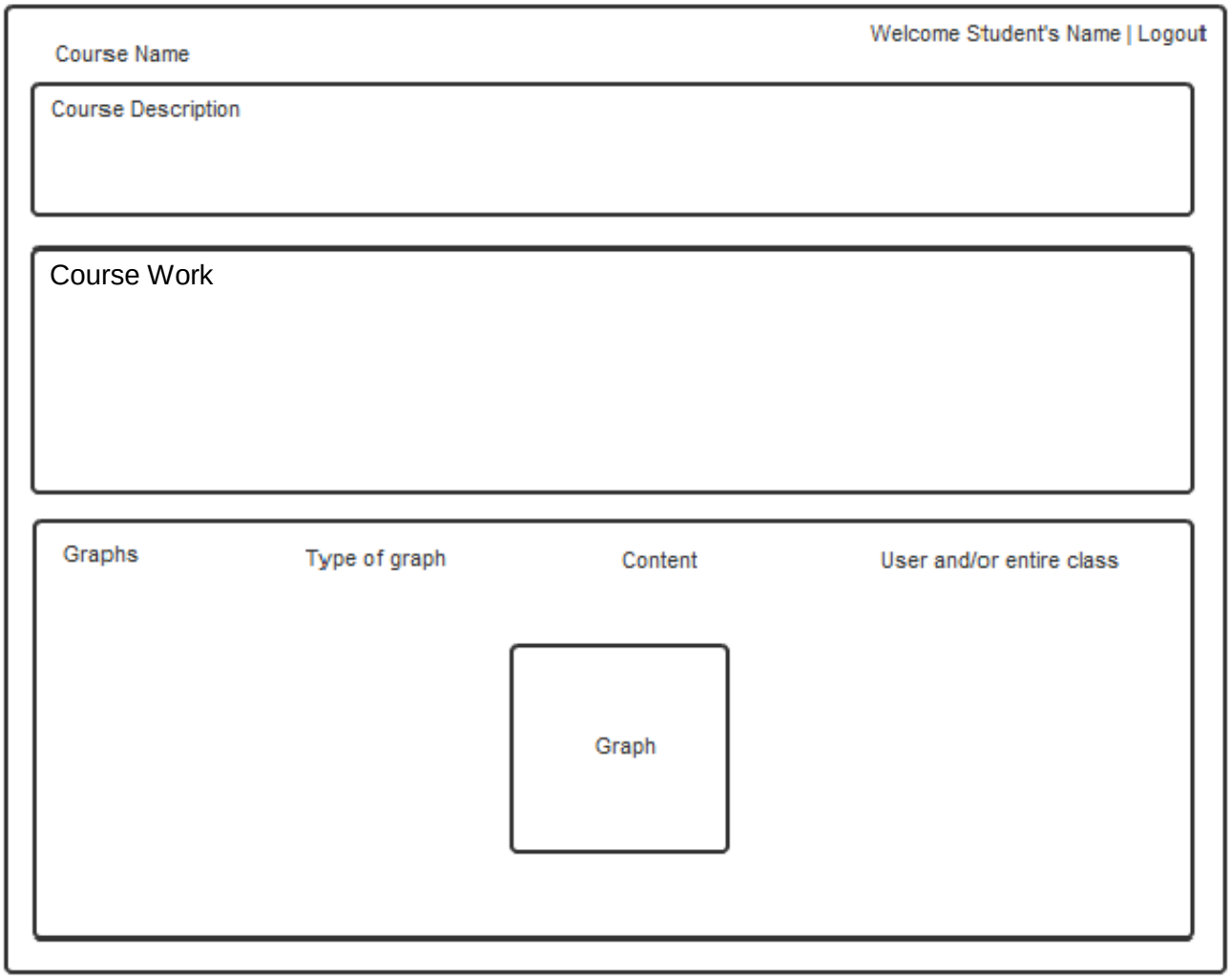
Mohamed Soudy
Anas Halbawi

Professor's account



Mohamed Soudy
Anas Halbawi

Course page:



Mohamed Soudy
Anas Halbawi

Part 4: Data Structures

Student model:

- User Name
- Andrew ID
- JSON string consisting of courses enrolled in

Course model:

- Course Name
- Course ID
- Professor teaching the course
- JSON string of students taking the course (by andrew ID)
- Grades (JSON string of dictionary (Key: andrew ID, Value: (dictionary key: exam/homework/quiz, value: grade)))
- individual grades graph (integer 1 or 0)
- individual graph for a specific assignment/quiz/exam (integer 1 or 0)
- Class average graph (integer 1 or 0)
- Class average graph for a specific assignment/quiz/exam (integer 1 or 0)
- Grades distribution of entire class of specific assignment/quiz/exam (integer 1 or 0)

Course Work model:

- Course ID
- Course work (homework, exam, quiz, etc...)
- Description
- Due date

Professor model:

- User Name
- Andrew ID
- JSON string consisting of courses teaching

Mohamed Soudy
Anas Halbawi

Part 5: URLs Map

URLs in parent folder

- GradeCenter/

URLs in application folder (GradeCenter)

URLs	Description	Return Type	Views	Data accessed
r'^(\d\d-\d\d\d)$'	Goes to course page	html	GradeCenter.views.getCourse	Course Model
r'^(\w+)$'	Goes to student/professor page	html	GradeCenter.views.getUser	Student/Professor Model
r'^getgrade/$'	Get grade for a specific course	json	GradeCenter.views.getGrade	Course Model
r'^getgraph/$'	Gets graph according to the type of graph chosen by student/professor.	html	GradeCenter.views.getGraph	Course Model
r'^updategrade/$'	Save the grade, whether it's a quiz, assignment or exam	html	GradeCenter.views.updateGrade	Course Model
r'^addcourse/$'	Add a course to the current user's profile	html	GradeCenter.views.addCourse	Course Model
r'^addwork/$'	Adds an assignment, quiz or exam, to the course work of a certain course.	html	GradeCenter.views.addWork	Course Work Model
r'^getwork/$'	gets the course work from a certain course	html	GradeCenter.views.getWork	Course Work Model