Bilkent University

Department of Computer Engineering

CS319
Object-Oriented Software Engineering

Classroom Helper

Group 1G

Design Report

Tuna Dalbeler (21802539), Ezgi Saygılı (21802871), Funda Tan (21801861),
Selahattin Cem Öztürk (21802856), Onur Korkmaz(21802925)

Instructor: Eray Tüzün
Teaching Assistants: Elgun Jabrayilzade, Erdem Tuna

Table of Contents

1. Introduction

1.1 Purpose of the System

The purpose of the Classroom Helper tool is to simplify the group formation, peer review and artefact review processes in classrooms. Our Classroom Helper tool is intended to be a usable, secure, reliable and portable tool. The target audience of our tool is instructors and students.

1.2 Design Goals

Usability:
The Classroom Helper tool must have a user-friendly interface to enable the users to use our tool without facing any difficulties. Also, our tool is planned to be intuitive and basic to increase usability.

Security:
Since the Classroom Helper tool collects sensitive information from the users, it must be secure.

Reliability:
The Classroom Helper tool should be reliable since it collects information and artefacts from the users which are needed to be accessible at any given time. System should get the input within 3 seconds and produce a response within 3 seconds. To prevent failures, Classroom Helper will have %95 uptime and will back up once in 3 days against any power loss. The Classroom Helper will use HTTPS protocol and have a secure connection.

Cost:
Because of very short time constraints, the program should be implemented with minimum cost and time

2. System Architecture

2.1 Subsystem Decomposition

Our system is decomposed into three subsystems by using model-view-controller architecture. “Model” subsystem interacts with the “Controller” subsystem and the “Controller” subsystem interacts with the “View” subsystem. With this decomposition, we reduced the coupling and made our system maintainable and flexible such that a change in a specific subsystem will not affect the whole system. In that way, we increased the coherence of our system and made future changes more applicable.

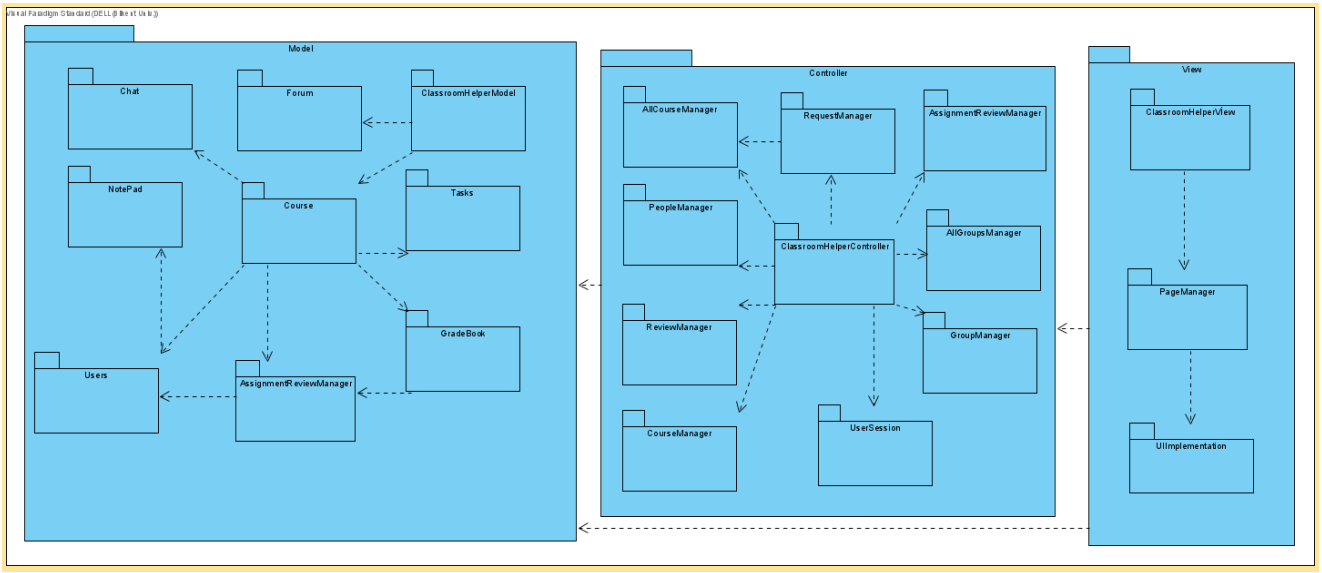

Figure 1 - Subsystem Decomposition Package Diagram

2.2 Hardware/Software Mapping

We will implement our system using Java and React. To run, our system will require a Java Runtime Environment and React support. Also, our system will require an internet connection and database to operate. The user’s browser must be the Firefox 88.0 or higher, Chrome 90.0.4430.85 or higher, Safari 14.0 or higher. (These are current versions.) Mobile platforms and older versions of these browsers are not supported.

2.3 Persistent Data Management

We need to store the data, usernames and passwords, grades, student submissions, etc, because of that, we will use a MySQL database to maintain the user and system data. The system will back up once in three days against any power loss and system failures. Student (also instructor) submitted files will be kept as files on the server.

2.4 Access Control and Security

Our system has two types of users which are “Student” and “Instructor”. The system will need the usernames and passwords to operate, because of that, our system has a verification process since it has a login sequence. Our system will have a secure connection and use HTTPS protocol.

Access Control Matrix

	Course	Group	Assignment	Artifact	Grade
Student		<<create>>	submitAssignment()	<<create>>	<<create>>
	enroll()	addGroup()	viewAssignment()	upload()	peerGrade()
	viewCourse()	sendInvitation()		delete()	displayStatistics()
	displayCourse()	declineRequest()		viewArtifact()	viewGrade()
	displayInstructorInfo()	leaveGroup()		createArtifacts()	
	displayGeneralTasks()	acceptRequest()		commentOnArtifact()	
		viewGroup()			
		sendJoinRequest()			
		removeGroup()			
		modifyGroup()			
		displayGroup()			
		addGroupTask()			
		removeGroupTask()			
		modifyGroupTask()			
		displayGroupTasks()			

Instructor	<<create>>	viewGroup()	<<create>>		<<create>>
	addCourse()	removeGroup()	createAssignment()		displayStatistics()
	removeCourse()	uploadGeneralTask()	deleteAssignment()	importQuestions()	gradeAssignment()
	modifyCourse()	addGroup()	viewAssignment()	artifactReview()	viewGrade()
	displayCourse()		uploadAssignment()	viewArtifact()	compareGrades()
	displayInstructorInfo()			getArtifactsforAssignment()	importGrades()
	addGeneralTask()			deleteArtifacts()	displayStudentDashboard()
	removeGeneralTask()			commentOnArtifact()	
	modifyGeneralTask()				
	displayGeneralTasks()				

Table 1 - Access Control Matrix

3. Subsystem Services

Model Subsystem

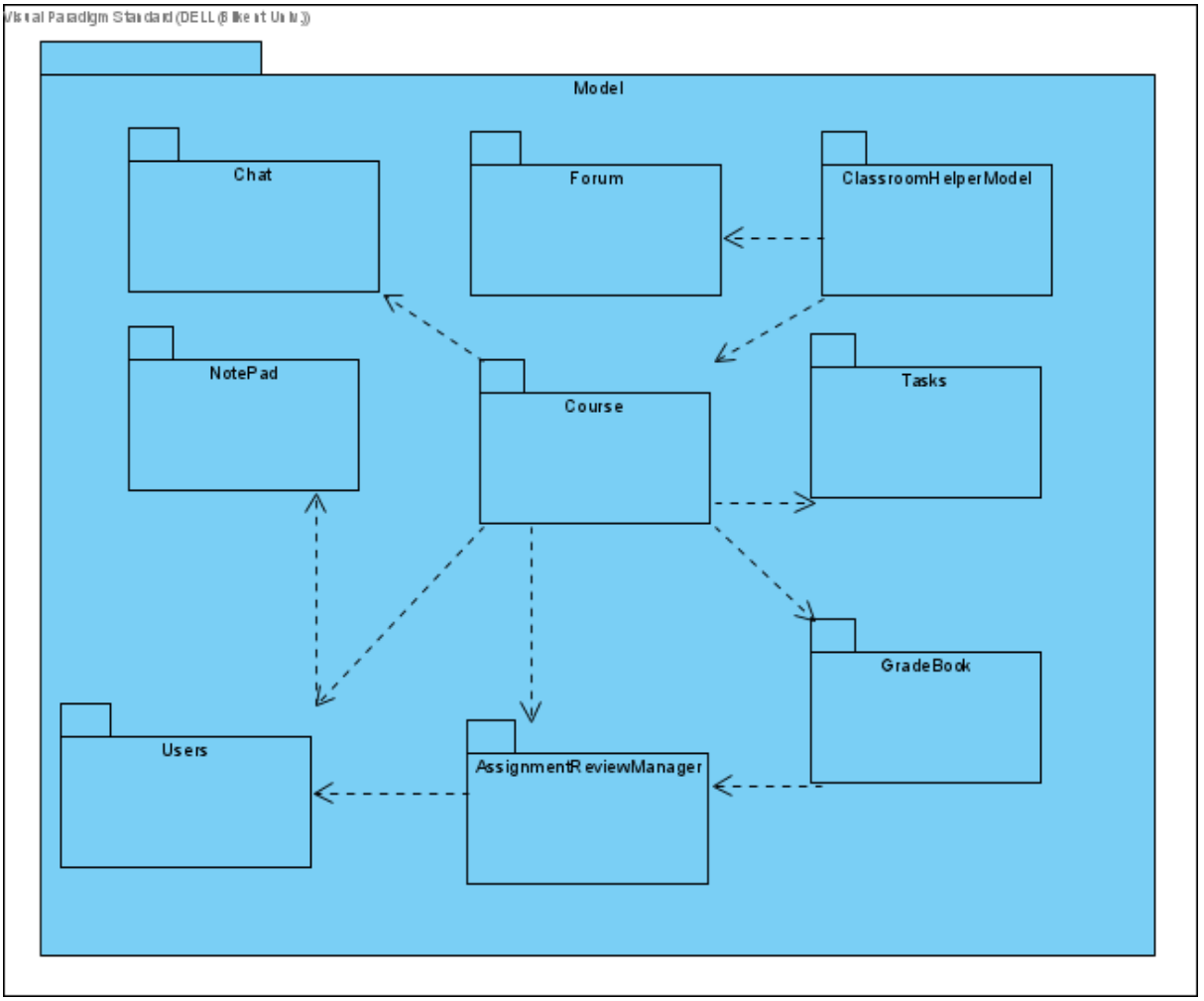

Figure 2 - Model Subsystem Package Diagram

“Model” subsystem has “Forum”, “Chat”, “Users”, “NotePad”, “GradeBook”, “Tasks”, “AssignmentReview” and “ClassroomHelperModel” packages and “Course”. Also, “Course” interacts with “Chat”, “Users”, “GradeBook”, “AssignmentReview” and “Tasks” packages. “User” interacts with the “NotePad” package and “ClassroomHelperModel” interacts with the “Forum” and “Course” packages. “GradeBook” depends on “AssignmentReview” and also, “AssignmentReview“ depends on the “Users” subsystem.

Users Subsystem

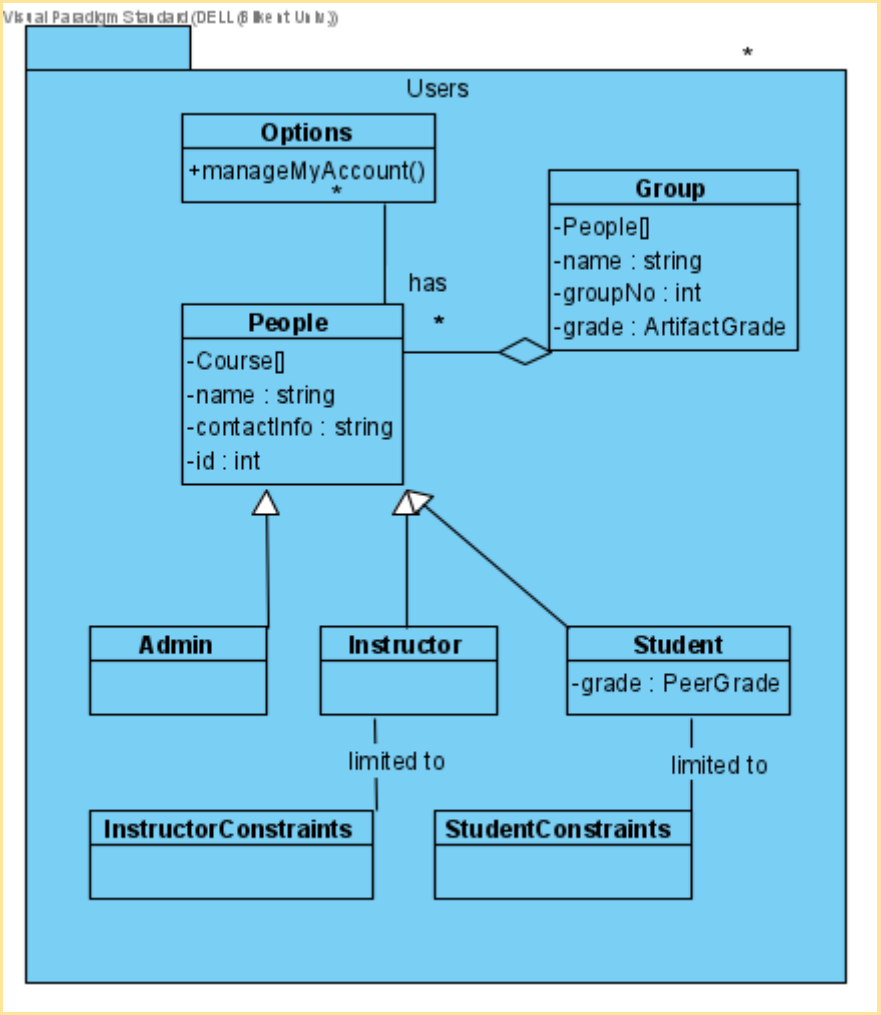

Figure 2 - Users Subsystem Package Diagram

“Users” subsystem consists of “Options”, “People”, “Group”, “Admin”, “Instructor”, “Student”, “InstructorConstraints”, and “StudentConstraints” classes. “Groups” class has a one-to-many relationship with the “People” class, that is, a group may have one or many people. “People” may be “Admin”, “Instructor” or “Student”. The “People” class has a Course array that stores courses of the given people. Also, the “People” class stores the “name”, “contactInfo” and “id” for that person.

Chat Subsystem

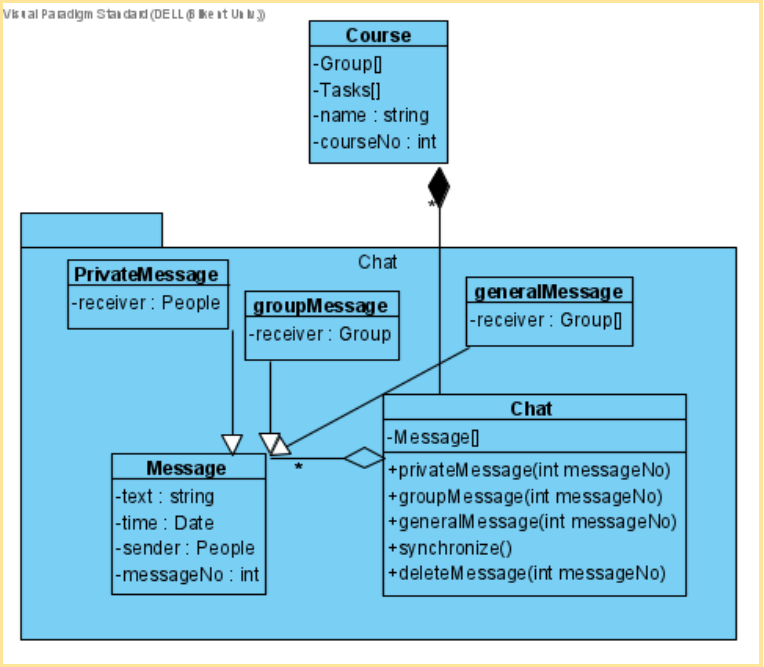

Figure 4 - Chat Subsystem Package Diagram

“Chat” subsystem consists of “PrivateMessage”, “groupMessage”, “generalMessage”, “Message”, “Chat”, and “Course” classes. “Chat” has a one-to-many relationship with “Message. Chat may have one or many messages. “Chat” has a Message array to store the messages and methods to maintain the messages provided by the users.

GradeBook Subsystem

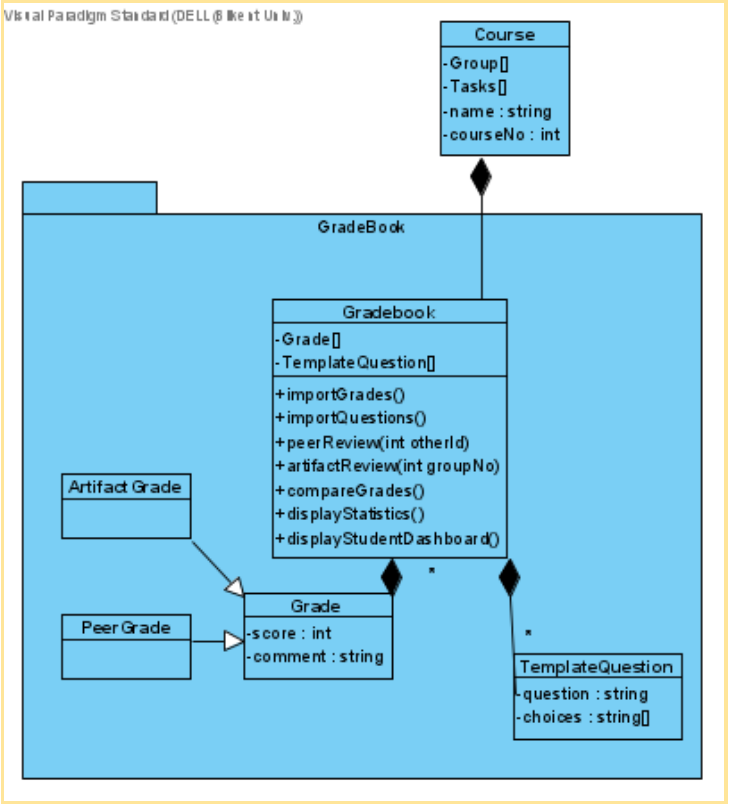

Figure 5 - GradeBook Subsystem Package Diagram

“Gradebook” subsystem consists of “Gradebook”, “ArtifactGrade”, “PeerGrade”, “Grade”, “TemplateQuestion” and “Course”. “GradeBook” has a one-to-many relationship with “Grade” and “TemplateQuestion”. Gradebook may have one or more grades and template questions. “Gradebook” has a Grade array to store the grades and methods to maintain the grades of the students.

Tasks Subsystem

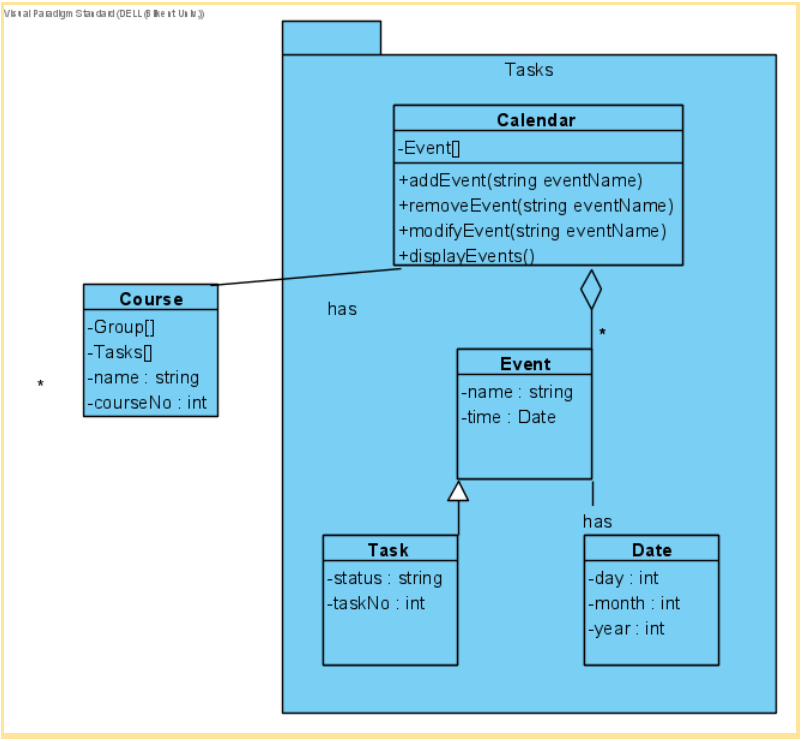

Figure 6 - Tasks Subsystem Package Diagram

“Tasks” subsystem consists of “Calendar”, “Event”, “Date”, “Tasks” and “Course” classes. There is a one-to-many relationship between “Calendar” and “Event”. The calendar may have one or more events. Also, “Calendar” has an Event array to store arrays and methods to maintain and display the stored events.

Notepad Subsystem

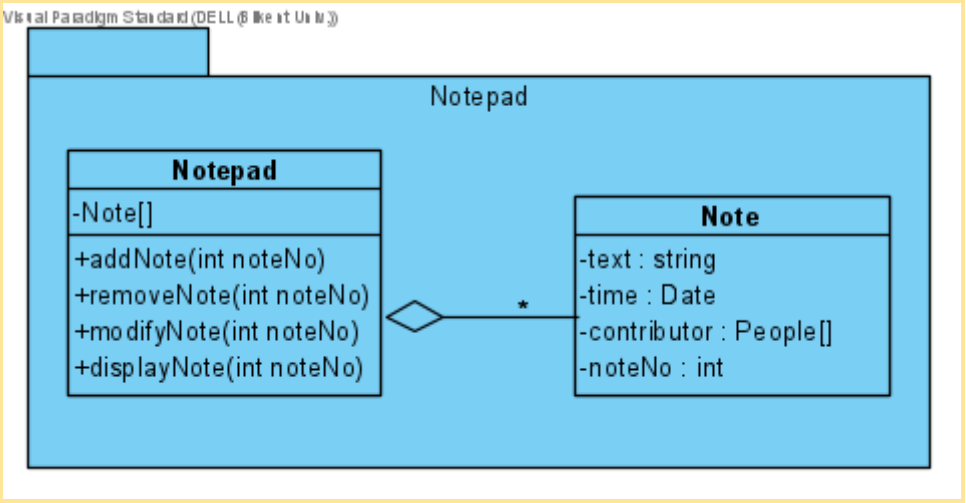

Figure 7 - Notepad Subsystem Package Diagram

The “Notepad” subsystem consists of “Notepad” and “Note” classes. “Notepad” and “Note” have a one-to-many relationship. Notepad can have one or more notes. “Notepad” has a Note array and methods to maintain the notes.

Forum Subsystem

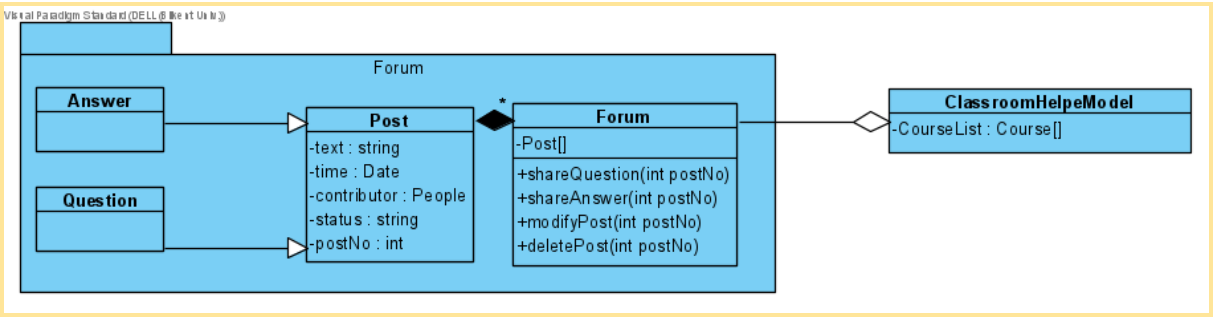

Figure 8 - Forum Subsystem Package Diagram

“Forum” subsystem consists of the “Forum” package and the “ClassroomHelperModel” class. “ClassroomHelperModel” class has an array called CourseList which consists of courses. “Forum” has “Post” and “Post” can be “Answer” or “Question”.

AssignmentReview Subsystem

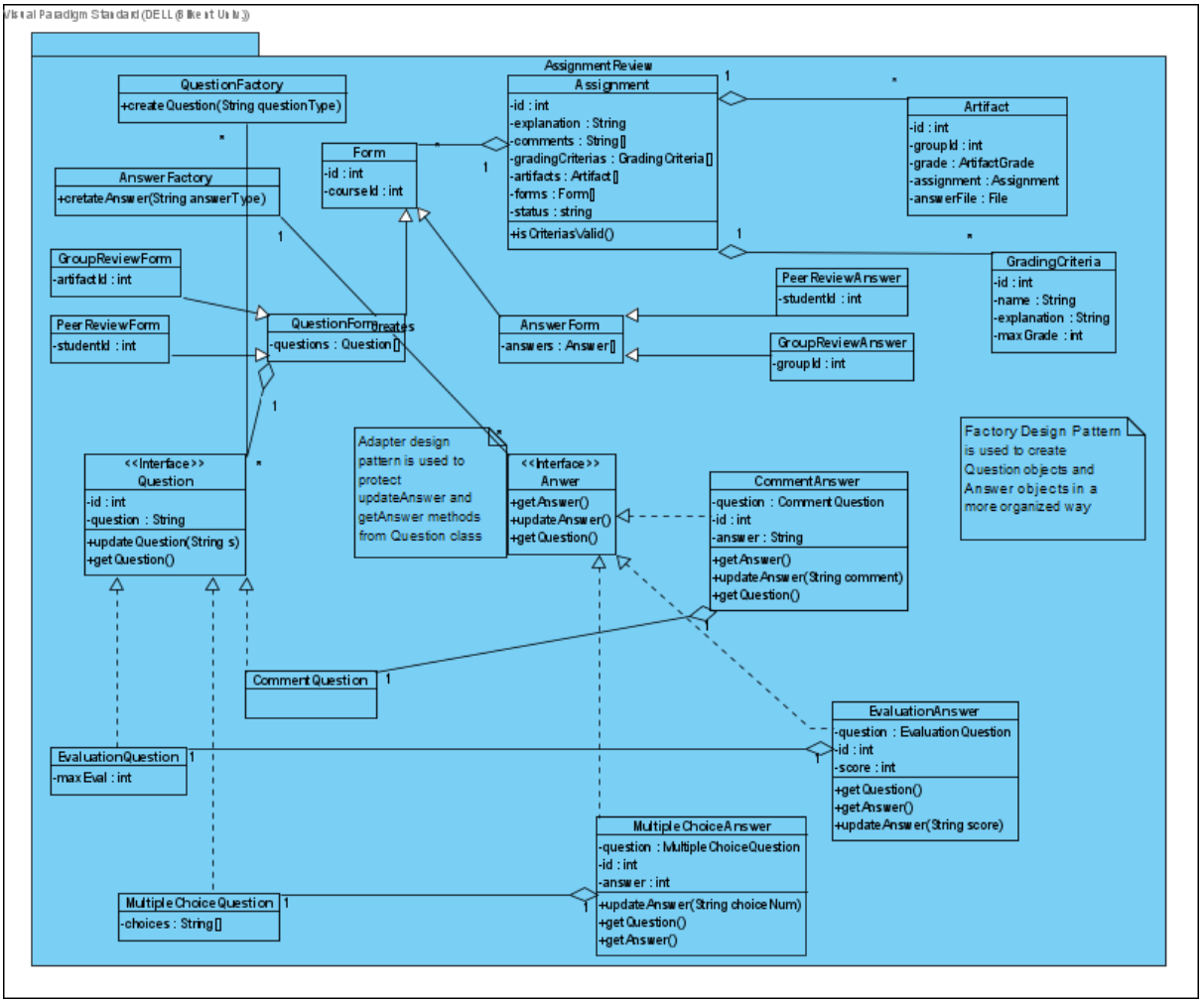

Figure 9 - Assignment Subsystem Package Diagram

“AssignmentReview” subsystem is to organize preparing assignments and generating forms for instructors and uploading files for artifacts and answering review forms for students. Instructor can create Assignment, then AssignmentReviewManager creates Artifacts for students belonging to that course. Then students can update “Artifact” objects.Instructors can create GroupReviewForm and PeerReviewForm, then ReviewManager will create corresponding Answer objects for students or groups. To generate forms, there are several features in CommentQuestion, EvaluationQuestion, MultipleChoiceQuestion. To make them answerable, We use Adapter design pattern with “Answer” interface and “MultipleChoiceAnswer”, “EvaluationAnswer” and “CommentAnswer” classes. Also, to make it easy to create “Question” and “Answer” objects for “Assignment”, “Artifact” and “Form”, we use Factory design pattern.

Controller Subsystem

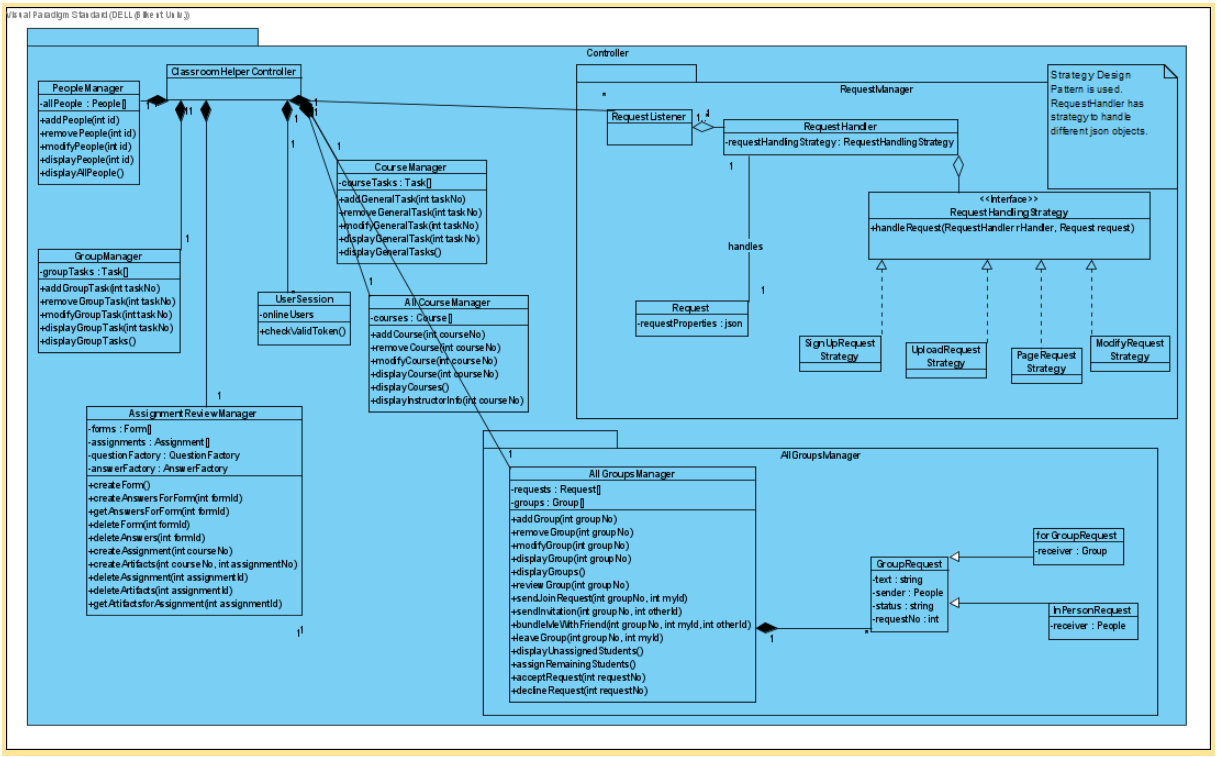

Figure 10 - Controller Subsystem Package Diagram

The “Controller” subsystem has the “ClassroomHelperController” class which has “PeopleManager”, “GroupManager”, “AllGroupsManager”, “CourseManager”, “UserSession”,”AssignmentReviewManager” and “RequestManager”. “ClassroomHelperController”, “PeopleManager”, “GroupManager”,“CourseManager”, “UserSession”, “AssignmentReview“ classes composes their own subsystems.

RequestManager Subsystem

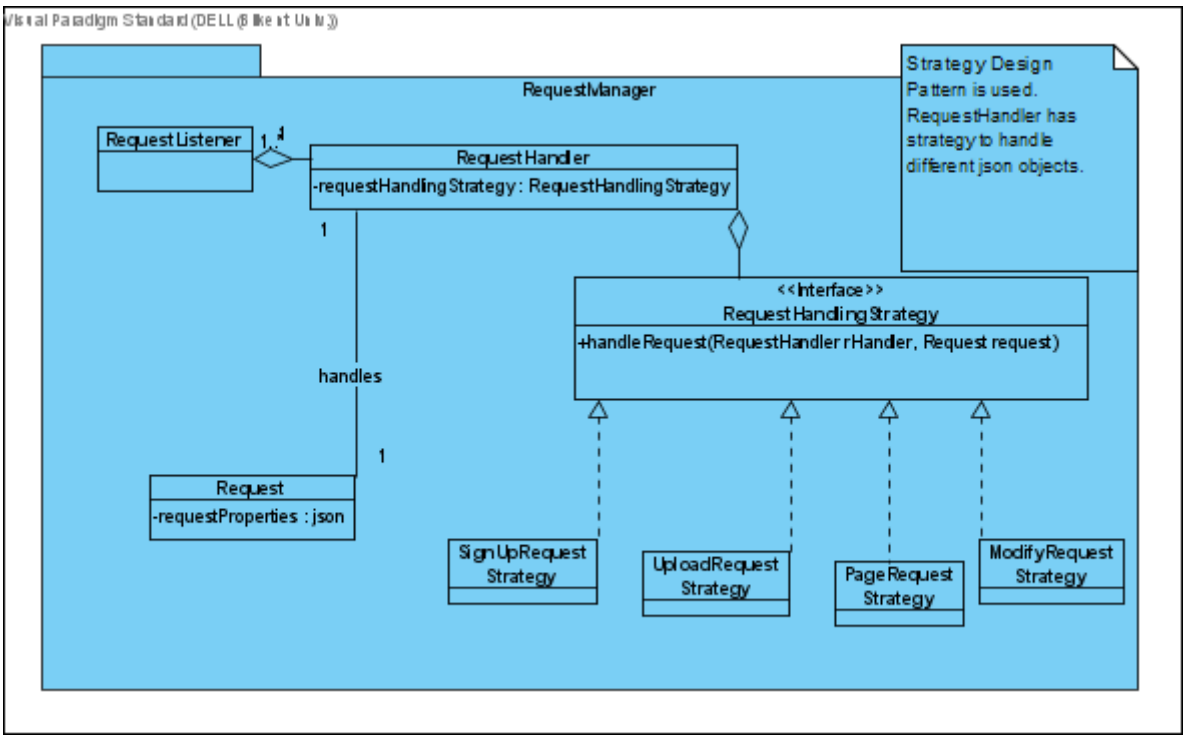

Figure 11 - Request Subsystem Package Diagram

The “RequestListener” class has “RequestHandler” and “RequestHandler” handles requests. Request class has JSON object and there are different types of strategies as RequestHandlingStrategy to handle requests.

AllGroupsManager Subsystem

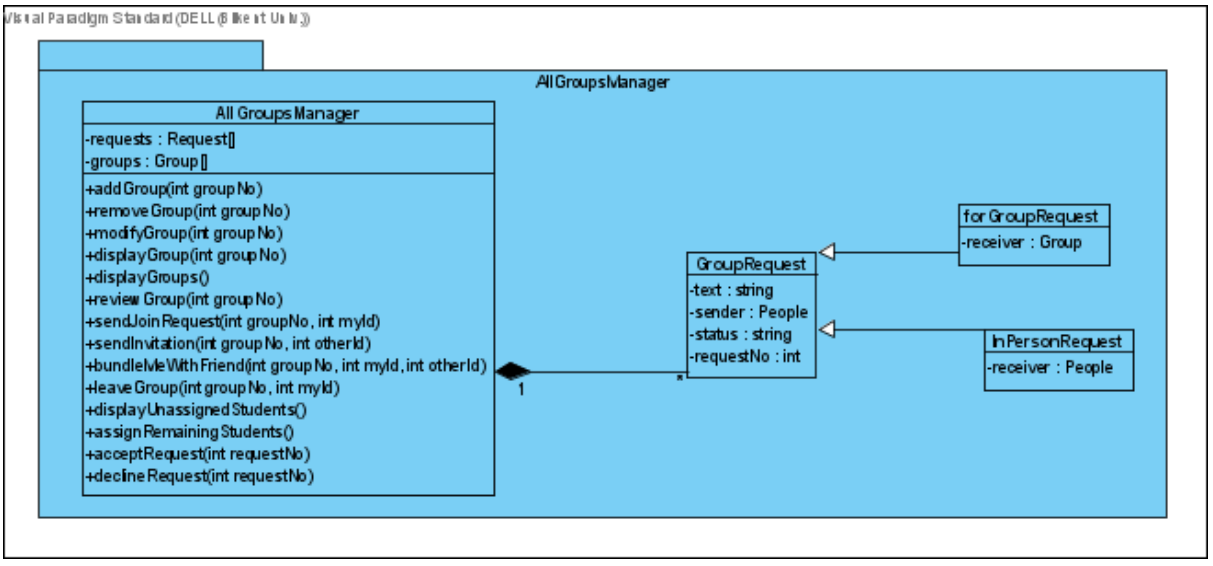

Figure 12 - AllGroupsManager Subsytem Package Diagram

View Subsystem

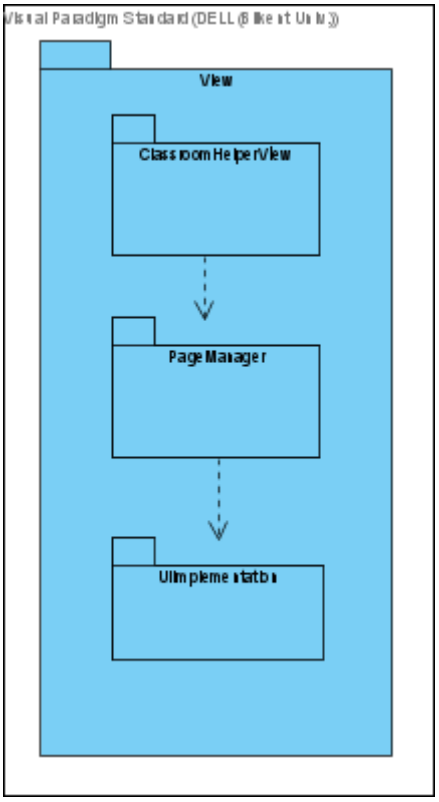

Figure 13 - View Subsystem Package Diagram

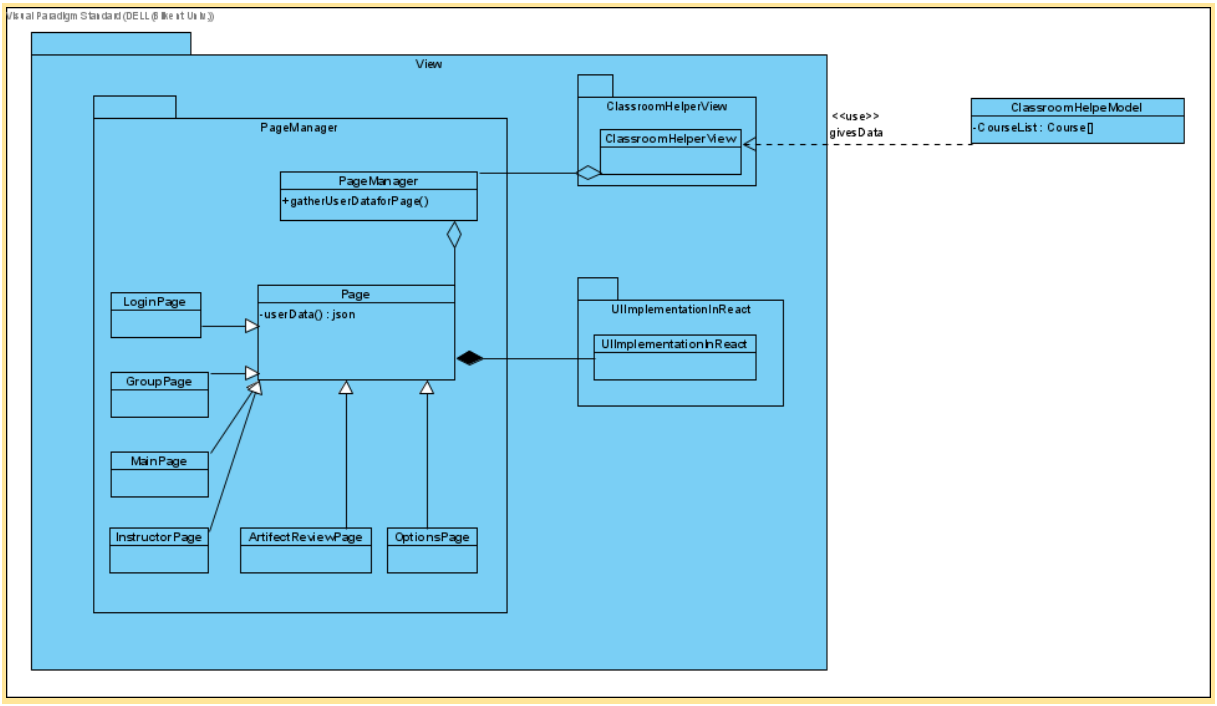

Figure 14 - View Subsystem Package Diagram

“ClassroomHelperView”, “LoginPage”, “GroupPage”, “MainPage”, “InstructorPage”, “ArtifactReviewPage”, “OptionsPage”, “ClassroomHelperView”, “PageManager”, “Page” and

“UIImplementationInReact” classes. “ClassroomHelperModel” has a CourseList array and provides the data for the “ClassroomHelperView”.

4. Low-Level Design

4.1. Object Design Trade-Offs

Functionality vs Usability:

Classroom Helper tool will have a simple and user-friendly interface enabling users to quickly grasp the use of the tool. User interface components and verbal interactions will be similar to that of contemporary communication tools. Names, messages, buttons, icons, and graphics will require a basic and intuitive understanding which will contribute to the usability of the product. On the other hand, the functionality of the tool will be kept limited to reduce complexity. By eliminating confusing and complicated operations, the usability of the tool will be optimized.

Cost vs Portability:

As a result of the limited time and energy allocated for the project, the Classroom Helper tool will only work on desktop operating systems that run Java and support React. Also, the user’s browser should be the latest of Firefox, Safari, or Chrome. So, users will not be able to use the tool on mobile platforms and older versions of these browsers will not be supported because of time constraints.

Performance vs Reliability:

For security requirements, the tool will have a secure connection and use HTTPS protocol. In the meantime between failure requirements, the tool will have 95% uptime (1 hour in a day might be down). For data loss tolerance, the tool will back up once in 3 days to protect against power loss. All of these will contribute to the reliability of the product while to some extent hindering the overall performance.

4.2. Final Object Design

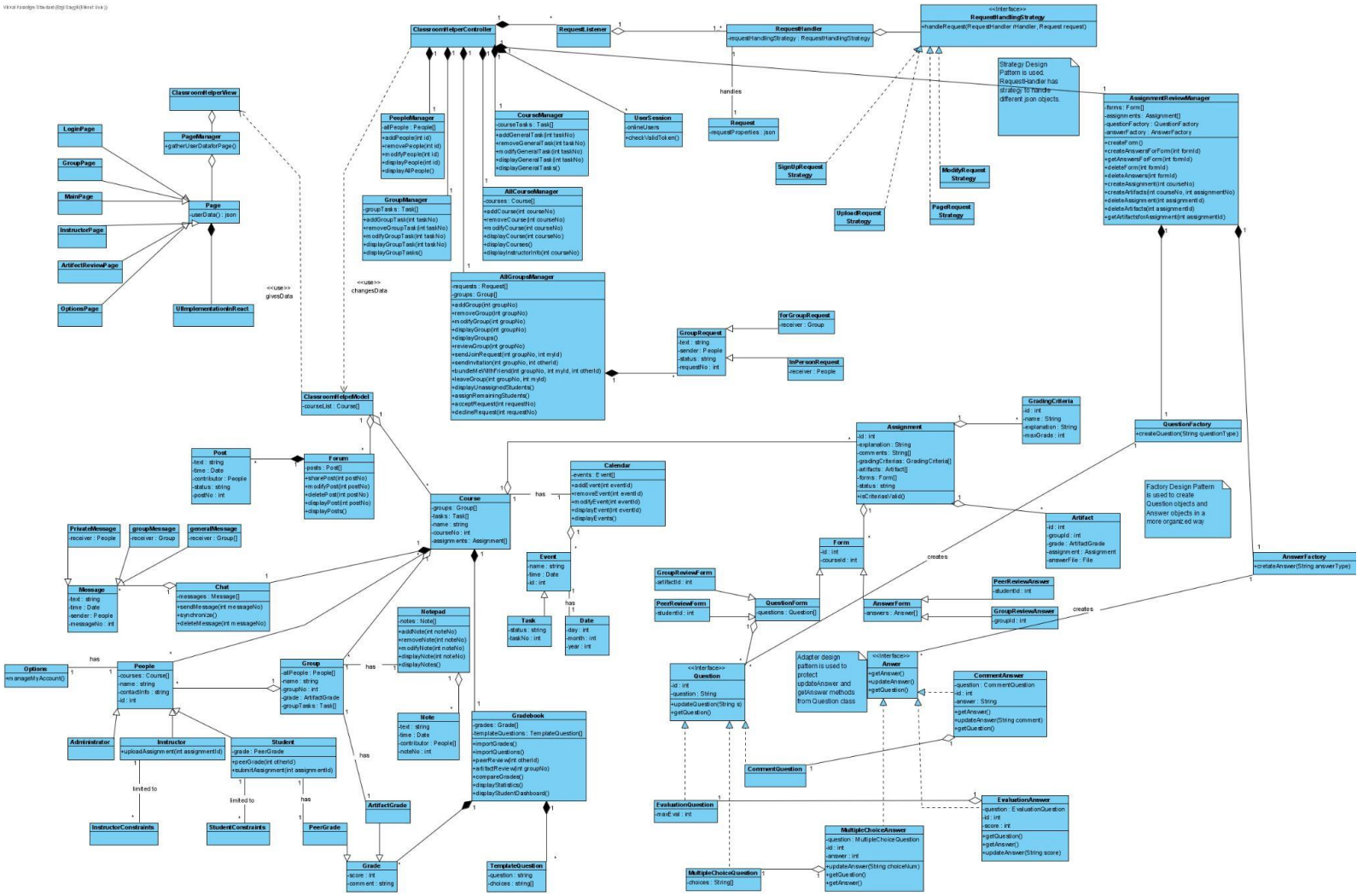

Figure 15 - Final Object Design

The class diagram of the “Classroom Helper” tool is illustrated above.

AllGroupsManager Class:

This class will be used for managing groups in a single course object. This class also has the Request class and Group class as an attribute.

Attributes:

- private Request[] requests: An array that contains all the requests.
- private Group[] groups: An array that contains all the groups.

Methods:

- public void addGroup(int groupNo): creates a group to the course.
- public void removeGroup(int groupNo): removes the group that has a matching group number.
- public void modifyGroup(int groupNo): modifies the group that has a matching group number.
- public void displayGroup(int groupNo): displays the corresponding group's name, number, grade, and students who are assigned to that group.
- public void displayGroups(): Displays all the groups that are currently in the course.
- public Group reviewGroup(int groupNo): returns a group object that has a matching group number.
- public int sendJoinRequest(int groupNo, int myId): sends a join request to a group that has a matching group number.
- public int sendInvitation(int groupNo, int otherId): sends an invitation to a student that has a matching id number.
- public int bundleMeWithFriend(int groupNo, int myId, int otherId): sends a join request to a group that contains two students.
- public void leaveGroup(int groupNo, int myId): enables people to leave their current group.
- public void displayUnassignedStudents(): displays the students that have not been assigned to a group.
- public void assignRemainingStudents(): assigns the students that have not been assigned to a group.
- public void acceptRequest(int requestNo): makes the request's status as accepted so that the student becomes assigned to the group.
- public void declineRequest(int requestNo): declines the join request.

GroupRequest Class:

This class contains the properties of the group request object.

Attributes:

- private string text: Message of the request.
- private People sender: The person who created the request.
- private string status: Indicates that the request is accepted or not.
- private int requestNo: Unique group request id.

Methods:

- public GroupRequest(string text, People sender, int requestNo): Constructor of GroupRequest class.
- public void setStatus(string status): sets the status of the group request.

ForGroupRequest Subclass:

This subclass was inherited from the GroupRequest class.

Attributes:

- private Group receiver: Indicates the receiver group of the group request.

Methods:

- public ForGroupRequest(string text, People sender, int requestNo, Group receiver): Constructor of ForGroupRequest class.
- public void setStatus(string status): sets the status of the ForGroupRequest.

InPersonRequest Subclass:

This subclass was inherited from the GroupRequest class.

Attributes:

- private People receiver: Indicates the receiver person of the request.

Methods:

- public InPersonRequest(string text, People sender, int requestNo, Person receiver): Constructor of InPersonRequest class.
- public void setStatus(string status): sets the status of the InPersonRequest.

Event Class:

This class contains the properties of the event object.

Attributes:

- private string name: Indicates the name of the event.
- private Date time: Indicates the due date of the event.
- private int id: Unique event id.

Methods:

- public Event(string name, int time): Constructor of the Event class.
- public void setTime(Date time): sets the time of the event.

Date Class:

This class contains the properties of the date object.

Attributes:

- private int day: Indicates the day.
- private int month: Indicates the month.
- private int year: Indicates the year.

Methods:

- public Date(int day, int month, int year): Constructor of Date class.

Task Sub-Class:

This subclass was inherited from the Event class.

Attributes:

- private string status: Indicates the status of the task.
- private int taskNo: Unique task id.

Methods:

- public Task(string status, int taskNo): Constructor of Task class.
- public void setStatus(string status): set the status of the Task.

AllCourseManager Class:

This class manipulates all course objects. This class also has the Course class as an attribute.

Attributes:

- private Course[] courses: An array that contains all the courses.

Methods:

- public void addCourse(int courseNo): adds a course.
- public void removeCourse(int courseNo): removes the course that has a matching course number.
- public void modifyCourse(int courseNo): modifies the course that has a matching course number.
- public void displayCourse(int courseNo): displays the course that has a matching course number. Course's name, number etc.
- public void displayCourses(): displays all the courses.
- public void displayInstructorInfo(int courseNo): displays the instructor information for the course that has a matching course number.

.

CourseManager Class:

This class manipulates the single course object's properties. This class also has the Task class as an attribute.

Attributes:

- private Task[] tasks: An array that contains all the tasks. These are mandatory course tasks assigned by the instructors.

Methods:

- public void addGeneralTask(int taskNo): adds a task to the course.
- public void removeGeneralTask(int taskNo): removes the task that has a matching task number from the course.
- public void modifyGeneralTask(int taskNo): modifies the task that has a matching task number.
- public void displayGeneralTask(int taskNo): displays the corresponding task's number and status.
- public void displayGeneralTasks(): displays tasks in the course.

GroupManager Class:

This class manipulates the single group object's properties. This class also has the Task class as an attribute.

Attributes:

- private Task[] tasks: An array that contains all the tasks. These are the in-group tasks assigned by group members.

Methods:

- public void addGroupTask(int taskNo, int groupNo): add a task to the group.
- public void removeGroupTast(int taskNo, int groupNo): removes the corresponding task from the group.
- public void modifyGroupTask(int taskNo, string status): changes the status of the task
- public void displayGroupTask(int taskNo): displays the corresponding task's number and status.
- public void displayGroupTasks(): displays all the tasks of the group.

Calendar Class:

This Calendar class contains the properties of the calendar object and its actions. This class also has the Event class as an attribute.

Attributes:

- private Event[] events: An array that contains all the events of the course and the group.

Methods:

- public void addEvent(int eventId): Adds an event to its events array.
- public void removeEvents(int eventId): Removes the event from the array if there is a matched event name.
- public void modifyEvent(int eventId): modifies the matched event's time.
- public void displayEvent(int eventId): displays the corresponding event's name, time and id.
- public void displayEvents(): displays all the events in its list.
- public Calendar(): Constructor of the Calendar class.

PeopleManager Class:

This class manipulates the people who are enrolled in a course. This class also has the People class as an attribute.

Attributes:

- private People[] allPeople: An array that contains all the people in the group.

Methods:

- public void addPeople(int id): Adds person to course's person list.
- public void removePeople(int id): Removes the person who has the matching id.
- public void modifyPeople(int id): modifies the mathed person's information.
- public void displayPeople(int id): displays the matched person's information.
- public void displayAllPeople(): displays all the people in its list.

Course Class:

Course class contains the properties of the course and its actions. This class also has the Group class, Task class and Assignment class as an attribute.

Attributes:

- private Group[] groups: An array that contains all the groups in the course.
- private Task[] tasks: An array that contains all the tasks in the course (general tasks).
- private string name: Name of the course.
- private int courseNo: Unique course id.
- private Assignment[] assignments: An array that contains all the assignments in the course.

Methods:

- public Course(string name): Constructor of the Course class.

Group Class:

Group class contains the properties of the group and its actions. This class also has the People class as an attribute.

Attributes:

- private People[] allPeople: An array that contains all the people in the group.
- private string name: Name of the group.
- private int groupNo: Unique group id.
- private ArtifactGrade grade: Grade of the group.
- private Task[] groupTasks: An array that contains all the tasks in the group (group tasks).

Methods:

- public Group(string name): Constructor of the Group class.
- public void setGrade(int grade): Sets the artifact grade of the group.

People Class:

People class contains the properties of the people object and its actions. This class also has the Course class as an attribute.

Attributes:

- private Course[] courses: An array that contains all the courses that are enrolled.
- private string name: Name of the person.
- private string contactInfo: contact information of the person.
- private int id: Unique id.

Methods:

- public People(string name): Constructor of the People class.
- public void setContactInfo(string contactInfo): Sets the contact information of the person.

Administrator Sub-Class:

This subclass was inherited from People class.

Methods:

- public Administrator(): Constructor of the Administrator sub-class.

Instructor Sub-Class:

This subclass was inherited from the People class.

Methods:

- public Instructor(string name): Constructor of the Instructor sub-class.

Student Subclass:

This subclass was inherited from the People class.

Attributes:

- private PeerGrade grade: contains the peer grade

Methods:

- public Student(string name): Constructor of the Student subclass.

InstructorConstraints class:

Defines constraints/limits for Instructor users.

StudentConstraints class:

Defines constraints/limits for Student users.

Options class:

Enables People to modify their name, contact information and id.

Methods:

- public void manageMyAccount(): modifies people's courses, name, contact information and id.
- public Options(): Constructor of the Options class.

Post Class:

This class contains the properties of the post object.

Attributes:

- private string text: Indicates the text of the post.
- private Date time: Indicates the due date of the post.
- private People contributor: Indicates the contributor of the post.
- private string status: Indicates the status of the post.
- private int postNo: Indicates the number of the post.

Methods:

- public Post(): Constructor of the Post subclass.
- public void setPostInfo(string text, Date time, People contributor): Sets information of the post.
- public void setStatus(string status): sets the status of the Post.

Forum Class:

This class manipulates the forum object properties. This class also has the Post class as an attribute.

Attributes:

- private Post[] posts: An array that contains the posts.

Methods:

- public void sharePost(int postNo): adds a post that has a matching post number.
- public void modifyPost(int postNo): modifies a post that has a matching post number.
- public void deletePost(int postNo): displays a post that has a matching post number.
- public void displayPost(int postNo): displays the corresponding post's information.
- public void displayPosts(): displays all the posts in its list.
- public Forum(): Constructor of the Forum class.

Note Class:

This class contains the properties of the note object.

Attributes:

- private string text: Indicates the text of the note.
- private Date time: Indicates the due date of the note.
- private People[] contributors: An array that contains people. Indicates the contributors of the note.
- private int noteNo: Indicates the number of the note.

Methods:

- public Note(): Constructor of the Note class.
- public void setNoteInfo(string text, Date time, People[] contributor): Sets information of the note.

Notepad Class:

This class manipulates the notepad object properties. This class also has the Note class as an attribute.

Attributes:

- private Note[] notes: An array that contains notes.

Methods:

- public void addNote(int noteNo): adds a note that has a matching note number.

- public void removeNote(int noteNo): removes a note that has a matching note number.
- public void modifyNote(int noteNo): modifies a note that has a matching note number.
- public void displayNote(int noteNo): displays a note that has a matching note number.
- public Notepad(): Constructor of the Notepad class.

Message Class:

This class contains the properties of the message object.

Attributes:

- private string text: Indicates the text of the message.
- private Date time: Indicates the due date of the message.
- private People sender: Indicates the sender of the message.
- private int messageNo: Indicates the number of the message.

Methods:

- public Message(): Constructor of the Message class.
- public void setMessageInfo(string text, Date time, People sender): Sets information of the message.

PrivateMessage Subclass:

This subclass was inherited from the Message class.

Attributes:

- private People receiver: contains the receiver.

Methods:

- public PrivateMessage(): Constructor of the PrivateMessage class.
- public void setReceiverInfo(People receiver): Sets receiver of the private message.

GroupMessage Subclass:

This subclass was inherited from the Message class.

Attributes:

- private Group receiver: contains the receiver.

Methods:

- public GroupMessage(): Constructor of the GroupMessage class.
- public void setReceiverInfo(People receiver): Sets receiver of the group message.

GeneralMessage Subclass:

This subclass was inherited from the Message class.

Attributes:

- private Group[] receiver: An array that contains the receiver.

Methods:

- public GeneralMessage(): Constructor of the GeneralMessage class.
- public void setReceiverInfo(People receiver): Sets receiver of the general message.

Chat Class:

This class manipulates the chat object properties. This class also has the Message class as an attribute.

Attributes:

- private Message[] messages: An array that contains the messages being sent and received.

Methods:

- public void sendMessage(int messageNo): adds a message that has a matching message number.
- private void synchronize(): synchronizes messages.
- public void deleteMessage(int messageNo): removes a message that has a matching message number.
- public Chat(): Constructor of the Chat class.

Grade Class:

This class contains the properties of the grade object.

Attributes:

- private int score: Indicates the score for the grade.
- private string comment: Indicates the comment for the grade.

Methods:

- public Grade(int score): Constructor of the Grade class.
- public void setComment(string comment): Sets comment for the grade.

PeerGrade Subclass:

This subclass was inherited from the Grade class.

Methods:

- public PeerGrade(): Constructor of the PeerGrade class.

ArtifactGrade Subclass:

This subclass was inherited from the Grade class.

Methods:

- public ArtifactGrade(): Constructor of the ArtifactGrade class.

TemplateQuestion Class:

This class contains the properties of the template question object.

Attributes:

- private string question: Indicates the question text for the template question.
- private string[] choices: Indicates the choices for the template question.

Methods:

- public TemplateQuestion(string question): Constructor of the TemplateQuestion class.
- public void setChoices(string[] choices): Sets choices for the template question.

Gradebook Class:

This class manipulates the gradebook object properties. This class also has the TemplateQuestion and Grade classes as attributes.

Attributes:

- private TemplateQuestion[] templateQuestions: An array that contains the template questions.
- private Grade[] grades: An array that contains the grades.

Methods:

- public void importGrades(): imports grades from other platforms.
- private void importQuestions(): imports questions from other platforms.
- public void peerReview(int otherId): enables a student to peer review another student in his/her group that has a matching id number.
- public void artifactReview(int groupNo): enables a group to review another group's artifact in the course that has a matching group number.
- public void compareGrades(): compares Grades given through artifact review with imported grades given by Instructor.
- public void displayStatistics(): displays statistics for the comparison of grades.
- public void displayStudentDashboard(): displays reviews about students.

AssignmentReviewManager Class:

This class manipulates the form object's and assignment object's properties. This class also has the Form class and Assignment class as attributes.

Attributes:

- private Form[] forms: An array that contains all the forms.
- private Assignment[] assignments: An array that contains all the assignments.

Methods:

- public void createForm(): creates a form.
- public void createAnswersForForm(int formId): adds answers to the corresponding form.
- public Answer[] getAnswersForForm(int formId): returns the answers for the corresponding form.
- public void deleteForm(int formId): deletes the corresponding form.
- public void deleteAnswers(int formId): deletes the answers for the corresponding form.
- public void createAssignment(int courseNo): adds an assignment to the corresponding course.
- public void createArtifacts(int courseNo, int assignmentNo): adds an artifact to the corresponding assignment of the corresponding course.
- public void deleteAssignment(int assignmentId): deletes the corresponding assignment.
- public void deleteArtifacts(int assignmentId): deletes the artifacts for the corresponding assignment.
- public Artifact[] getArtifactsforAssignment(int assignmentId): returns the artifacts for the corresponding assignment.

Assignment Class:

This class contains the properties of the assignment object.

Attributes:

- private int id: Unique assignment id.
- private string explanation: Indicates the explanation of the assignment.
- private string[] comments: Indicates the comments of the assignment.
- private GradingCriteria[] gradingCriterias: An array that contains grading criterias.
- private Artifact[] artifacts: An array that contains artifacts.
- private Form[] forms: An array that contains forms.
- private string status: Indicates the status of the assignment.

Methods:

- public Assignment(): Constructor of the Assignment class.
- public void setAssignmentInfo(string explanation, string[] comments, GradingCriteria[] gradingCriterias, Artifact[] artifacts, Form[] forms): Sets information of the assignment.
- public void setStatus(string status): sets the status of the assignment.
- public boolean isCriteriasValid(): checks whether criterias are valid.

GradingCriteria Class:

This class contains the properties of the grading criteria object.

Attributes:

- private int id: Unique grading criteria id.
- private string name: Indicates the name of the grading criteria.
- private string explanation: Indicates the explanation of the grading criteria.
- private int maxGrade: Indicates the maximum grade of the grading criteria.

Methods:

- public GradingCriteria(): Constructor of the GradingCriteria class.
- public void setCriteriaInfo(string name, string explanation, int maxGrade): Sets information of the grading criteria.

Artifact Class:

This class contains the properties of the artifact object.

Attributes:

- private int id: Unique artifact id.
- private int groupId: Unique group id.
- private ArtifactGrade grade: Indicates the grade of the artifact.
- private Assignment assignment: Indicates the assignment of the artifact.
- private File answerFile: Indicates the file of the artifact.

Methods:

- public Artifact(): Constructor of the Artifact class.
- public void setArtifactInfo(int groupId, ArtifactGrade grade): Sets information of the grading criteria.

Form Class:

This class contains the properties of the form object.

Attributes:

- private int id: Unique form id.
- private int courseId: Unique course id.

Methods:

- public Form(): Constructor of the Form class.
- public void setForm(int courseId): Sets course for the form.

QuestionForm Subclass:

This subclass was inherited from the Form class.

Attributes:

- private Question[] questions: An array that contains questions.

Methods:

- public QuestionForm(): Constructor of the QuestionForm class.

AnswerForm Subclass:

This subclass was inherited from the Form class.

Attributes:

- private Answer[] answers: An array that contains answers.

Methods:

- public AnswerForm(): Constructor of the AnswerForm class.

GroupReviewForm Subclass:

This subclass was inherited from the QuestionForm class.

Attributes:

- private int artifactId: Unique artifact id.

Methods:

- public GroupReviewForm(): Constructor of the GroupReviewForm class.

PeerReviewForm Subclass:

This subclass was inherited from the QuestionForm class.

Attributes:

- private int studentId: Unique student id.

Methods:

- public PeerReviewForm(): Constructor of the PeerReviewForm class.

PeerReviewAnswer Subclass:

This subclass was inherited from the AnswerForm class.

Attributes:

- private int studentId: Unique student id.

Methods:

- public PeerReviewAnswer(): Constructor of the PeerReviewForm class.

GroupReviewAnswer Subclass:

This subclass was inherited from the AnswerForm class.

Attributes:

- private int groupId: Unique group id.

Methods:

- public GroupReviewAnswer(): Constructor of the GroupReviewAnswer class.

Question Class:

This class contains the properties of the question object.

Attributes:

- private int id: Unique question id.
- private string question: Indicates the statement of the question.

Methods:

- public Question(): Constructor of the Question class.
- public void setQuestionInfo(string question): Sets information of the question.
- public Question getQuestion(): returns the question.
- public void updateQuestion(string s): updates the statement of the question.

CommentQuestion Subclass:

This subclass was inherited from the Question class.

Methods:

- public CommentQuestion(): Constructor of the CommentQuestion class.

EvaluationQuestion Subclass:

This subclass was inherited from the Question class.

Attributes:

- private int maxEval: Indicates the maximum evaluation value.

Methods:

- public EvaluationQuestion(): Constructor of the EvaluationQuestion class.

MultipleChoiceQuestion Subclass:

This subclass was inherited from the Question class.

Attributes:

- private string[] choices: Indicates the choices of the question.

Methods:

- public MultipleChoiceQuestion(): Constructor of the MultipleChoiceQuestion class.

Answer Class:

This class is an interface for an Answer object.

Methods:

- public Answer getAnswer(): returns the answer.
- public void updateAnswer(string): updates the statement of the answer.
- public Question getQuestion(): returns the question.

CommentAnswer Class:

This class implements the Answer class.

Attributes:

- private CommentQuestion question: Indicates the question for the comment answer.
- private int id: Unique comment answer id.
- private string answer: Indicates the statement of the comment answer.

Methods:

- public CommentAnswer getAnswer(): returns the comment answer.
- public void updateAnswer(string): updates the statement of the comment answer.
- public CommentQuestion getQuestion(): returns the comment question.

EvaluationAnswer Class:

This class implements the Answer class.

Attributes:

- private EvaluationQuestion question: Indicates the question for the evaluation answer.
- private int id: Unique evaluation answer id.
- private int score: Indicates the score for the evaluation answer.

Methods:

- public EvaluationAnswer getAnswer(): returns the evaluation answer.
- public void updateAnswer(string): updates the statement of the evaluation answer.
- public EvaluationQuestion getQuestion(): returns the evaluation question.

MultipleChoiceAnswer Class:

This class implements the Answer class.

Attributes:

- private MultipleChoiceQuestion question: Indicates the question for the multiple choice answer.
- private int id: Unique multiple choice answer id.
- private int answer: Indicates the choice for the multiple choice answer.

Methods:

- public MultipleChoiceAnswer getAnswer(): returns the multiple choice answer.
- public void updateAnswer(string choiceNum): updates the choice for the multiple choice answer.
- public MultipleChoiceQuestion getQuestion(): returns the multiple choice question.

4.3. Packages

org.springframework: A framework for building backends with RestApi.[1]

javax.persistence : Java Persistence is the API for the management for persistence and object/relational mapping.

java.io: Reading and writing files to the system such as artifacts etc..

react-doc-viewer: A web browser document viewer for Artifact Reviewing.[2]

react-burger-menu: A burger menu for navigation between pages.[3]

react-widgets: Extended React input options. [4]

[1] https://spring.io/projects/spring-boot
[2] https://www.npmjs.com/package/react-doc-viewer#basic
[3] https://www.npmjs.com/package/react-burger-menu
[4] https://jquense.github.io/react-widgets/

4.4. Design Patterns

- Strategy Design pattern in Requests
- Factory Design pattern in Answers
- Factory Design pattern in Questions.

5. Improvement Summary

- We have decided to apply Model-View-Controller architecture.
- We have added Design Patterns and a Design Patterns section (Section 4.4).
- Artifact class is added to the Final Object Design.
- Access Control Matrices for students and instructors are added.
- Persistent Data Management (Section 2.3) is updated.
- Subsystem Decomposition Package Diagram (Figure 1) is updated.

6. Glossary & References

https://spring.io/projects/spring-boot

https://www.npmjs.com/package/react-doc-viewer#basic

https://www.npmjs.com/package/react-burger-menu

https://jquense.github.io/react-widgets/