Cairo University
Faculty of Computers and Information
Department of Computer Science

FCI E-campus

Supervised by

Dr. Mohammed Nassef

TA. Nora Abdelhamid

Implemented by

20140169	*Abdulaziz Mahmoud Abdulaziz*
20140163	*Abdelrahman Mahmoud Mohamed*
20140197	*Amr Magdy Mohamed Ezzat*
20140030	*Ahmed Abdelmohsen Ahmed*
20140170	*Abdallah Abdelazim Abdallah*

Academic Year 2017-2018

Midyear Documentation of Graduation Project

Table of Contents

Abstract:

When it comes to organizing their class schedule, lectures/sections/labs materials and upcoming assignments/exams, students here at Faculty of Computer and Information, Cairo University are in a terrible mess.

Students at FCI are scattered across many places. Some professors and TAs publish their announcements and materials on the faculty's official website, others use a third-party website such as Acadox, and moreover some professors and TAs send the materials and announcements to a student who is supposed to deliver that to all the other students. Besides that, the faculty's schedule is provided to students as just a plain document. Students lose track of assignments deliveries and quizzes that they have.

Moreover, the faculty doesn't provide a forum where the students can ask and discuss about things relating to their courses or things that aren't clear for them or they just don't understand. Students have to use social network websites such as Facebook to communicate which is so distracting and it won't be appropriate for professors and TAs to answer the questions and help the students.

So, after all of that students are scattered between the faculty's website, Acadox, Facebook, GDrive, Dropbox, Mega, etc.… After all of that they can't help being lost and unorganized.

Our app aims to help students stay organized by providing them with their schedule, depending on what year they're in, their major, minor and the courses they have registered. The schedule will tell them what lectures/labs/sections they have and the location of these in the faculty lectures/sections halls and labs. If the student doesn't know the location of his lecture or lab the app will help him and show him a map to where the lecture/section hall or lab is in the faculty. The Schedule will inform them if there's any quizzes, exams or assignments delivery at a certain lab or lecture.

Also, students receive notifications for new announcements, upcoming quizzes and assignments deliveries to help them keep track with their courses. Students will also be notified regularly to remind them of their assignments, quizzes and exams and urge them to do their studies and work early before due date.

The app will organize downloading and uploading materials and will provide each course's lectures, sections and labs with its materials that was uploaded.

That's not it the app will provide a forum for the students to communicate with each other. Students can help each other and what will be better is that now it will be appropriate for professors and TAs to enter the forum and answer some of these questions. The forum will be categorized and organized according to course, topic, etc...

The app will contain many more features that will help FCI students in many aspects

The tools used in implementing the project are:

1- The app frontend clients will be an **Android** app (and possibly a website especially an Admin website).
Tools: Android Studio IDE

2- The app backend will be built using **PHP Laravel** with **MySQL** database. The backend will be built using Service Oriented Architecture (SOA) providing RESTful web services to be used in the front-end clients.
Tools: PhpStorm IDE, Laragon, MySQL workbench, XAMPP, …

Functional requirements:

1- The app will allow each student, professor and TA to own an account but everyone has special permissions and capabilities in the system. The student does not need to be verified he can sign up normally providing basic information about himself, then he can register in the courses by a providing a course code (similar to Acadox) and also a group number (each student can be in only one group in each course). The professor and TA must be verified because they have more options like upload file, add materials, add tasks and more.

There are two types of students, normal student and moderator student. Each course has its moderator students that are chosen by the course's professor (and maybe TAs also). This moderator student is just a normal student but he has higher permissions in the course. Moderator students can upload materials in the extra resources section of the course and also manages the course's forum.

The verification of professors and TAs will be an automated process (similar to WhatsApp's signup) and it will be as follows:

In the app's sign up page screen will be an option to sign up as a student or teacher. The professors and TAs will choose to sign up as teachers. It will be mandatory for them to sign up with their FCI academic email (ex. m.nassef@fci-cu.edu.eg). The app then will automatically send an email to the specified email containing a secret code and the doctor/TA can open that email and put the secret code in our app's sign up screen and continue creating the account.

2- A Schedule with courses lectures, labs and sections each student has according to his registered courses. This schedule shows the user what he/she has at every slot of the day (like lecture, lab, section or delivering assignments and tasks or exams). Each student has his own schedule which include only stuff related to his registered courses.

3- Most of new students don’t know the location of their lecture/section halls which makes a big problem to them. They spend a lot of time searching for their lecture/section hall or they miss the lecture/section, we can solve this issue by adding a static map for the faculty.
This map is a static graphical map which guides the students by showing the buildings and the location of each lecture/section hall as well as professors’ offices, TAs’ office, the library, the mosque, the restaurant, etc…
As well as the students can search for a location by typing it into the search bar and the map will refer to their needs if any.

4- For each course the student is registered in there will be a resources screen which includes the materials for this course. The materials can be files of any type (PDF, Word, PowerPoint presentation) as well as any links to external resources such as videos or websites and also notes from the material uploader.
Students can view the materials and download them. Professors and TAs can upload new materials for the courses they teach.
In the resources screen there will be extra materials section which will include the materials uploaded by the moderator students. These resources can be also files or links such as summaries or links to useful websites or videos.
This material will be added in the server side, which can be requested to download by the frontend users (Students). Course resources and materials can only be viewed by the course’s registered students only.

5- Allow Professors & TAs to add materials which can be files, of any type, and “useful links to video, audio, text files or websites” to the course they assigned to, then a notification is sent to all the students assigned to this course informing them that there is a new material added to this course and when they click on the notification the resources screen for this course is opened, we also then can add highlighting to the new materials added.

6- Allow Professors & TAs to add tasks to the courses they are assigned to. These tasks can be assignments, lab tasks, quizzes, etc... Each task will have a due date and a description which can be a separate file or just a text viewed within the assignment’s screen. These tasks appear on students’ schedules. Only professors and TAs can add tasks and only for the courses they are assigned to. Students registered to that course will get a notification for the new tasks and they can view it in the course’s tasks screen.

7- The app will have an Announcements Screen to inform students with any new announcements made by the Admin (similar to announcements in FCI E-com). The announcements screen contains

announcements that does not relate to a specific course. The announcements will be information related to the faculty in general (ex. informing students that a day is an official holiday, informing students about the date of the first day of the term, etc…). All the students signed up to the app can see this announcements screen even if they are not registered in any courses.
When a new announcement is added, Notification will be sent to all users of the app (including students and teachers).

8- Each course will have a Q/A Forum. The course students can post in the course forum to ask a question (related to this course) and other students, professors and TAs can answer.
Moderator students can delete any forum post if they find that it was inappropriate such as unrelated to the course "or asked before (extra)".
The forum post will be Q/A-based and each post, very similar to stack overflow, will have a "solved" flag that the student who asked the question can set and it will indicate that this post is fulfilled and it doesn't require more solutions (however more can be added). This will help to focus on the unanswered questions.
The app saves all this forum posts and allows students to view them and search for a specific question by "tags".

Extra Stuff:

9- Tasks can be set to require students to upload their solutions. Students need to upload their solutions before due date. The course's TAs & professors can download these solutions, correct them and add the grade to each student.

10- Students will get notifications regularly to remind them of their tasks that has a near due date.
The Notification system in the application will be divided into two subsystems.
The first subsystem is responsible for notifying the users for any **new** incoming changes on the application. This will include notifying the users for new announcements, new assignments, new tasks added, new quizzes and new exams.
The notification system will not be a background service, but it will collaborate with the synchronization service on the client app. This sync service is responsible for listening to events on the backend. These events will be such as the arrival of a new announcement or the addition of a new assignment on the courses that the user is registered in. The sync service will update the application and calls the notification system to show a notification to the user.
The second subsystem is a reminder background service that is responsible for showing reminders notifications. Reminder notifications will be managed locally without the need for any backend sync (very similar to an alarm application). Take this example: If a user has an assignment

that will be delivered on Thursday the reminder service will send a notification before a week from the deadline, if the user hasn't marked this assignment as done, the reminder will continue and sends a notification to the user before 4 days from the deadline, then again if it's not marked as done it will send a notification to the user before 2 days from the deadline and so on it will continue to send notification before a day, then before 6 hours, … and so on until either the user either marks the assignment as done or the deadline is passed.

11- The app will allow users (professors, students or TAs) to send direct private messages to any other user. This is **not a chat service**. The messages will be similar to emails with a title, body & attachments such as images or raw files. The user can search for another user by his name or his ID in our system, then he can send a message to him as described above. Messages will be grouped in an inbox page and categorized and organized by the user it was sent to or received from (similar to chat rooms) so that all messages sent to or received from the same user appear in a single specific place inside the inbox.

Users Roles:

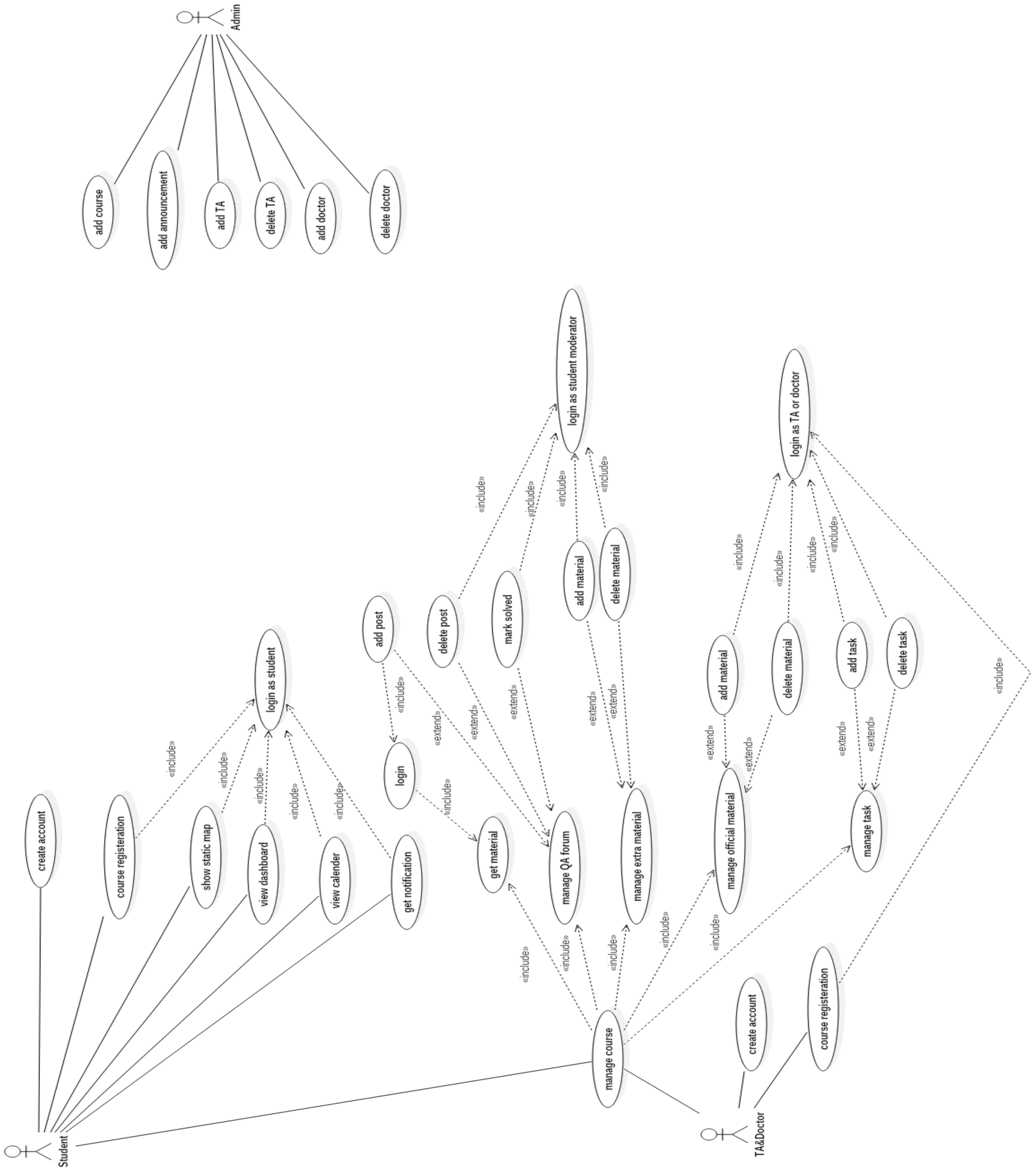
1. **Student**: student can register to courses and view its schedule, materials, forum, tasks, etc…
2. **Moderator Student**: This is just a normal student but with more permissions in the course. These permissions are that he can add materials in the extra materials section of the course (moderator student's materials won't be mixed with the official materials sent from doctors or TAs) and he also manages the forum helping deleting inappropriate posts.
3. **Doctor/TA**: both of them can be assigned to the courses and they can add materials, forum posts, tasks or assignments there. Each course can have one or more doctors. The course's doctors are responsible for choosing and approving the moderator students of this course. Also each course can have many TAs.
4. **Admin**: this user represents the faculty and can be a specific person hired by the faculty (one of the student's affairs employees for example). He maintains the app, maintains and update the list of the faculty's professors and TAs' FCI academic emails and also creates courses. He assigns doctors to the courses and assigns TAs to the courses specifying the group of each TA that he/she teaches in the course (TA can have more than one group and doctors have all groups). When the year ends he archives the current courses (saves its materials + schedule report) and resets them so they are ready to be assigned to new professors and TAs and registered by new students and lastly he can add schedules for the courses (schedules will be added manually or from an excel sheet).

   The admin also can receive reported problems from any app user. He will receive them and be notified for them.

Non-Functional requirements:

1. **Accessibility**
   - Due to the spreading of smart phones that uses android systems our application can be accessed by large community “as it will be an android app”.
2. **Availability**
   - Our app is available 24-hours.
3. **Regulatory compliance**
   - Schedules in our app will be very specific, without any fault and up-to-date with the college schedule.
4. **Cost**
   - Our app will be for free
5. **Deployment**
   - Our app will be available on app store.
6. **Efficiency**
   - Our app will use only the needed martials “schedule for the student will contains the schedule of the courses that he registered in only without containing the schedule of the other courses.
7. **Effectiveness**
   - Our app will have all the features that the students needed instead of using more than one app.
8. **Extensibility**
   - We can add easily more features in our app in the future “that is what we decide to do in our project there is extra functions that we will added later.
9. **Interoperability**
   - Our app can be used easily by other colleges.
10. **Maintainability**
    - If any user facing a problem in the app he can easily report it and we will repair it.
11. **Modifiability**
    - Some features can be modified easily like the shape of the schedule.
12. **Safety**
    - The data of any user will be completely save as it will not access by any other user.
13. **Open-source**
    - The source code will be available for any one.
14. **Usability**
    - Simple UI that help the user to use its functions easily.

Use-Case Diagram:



Sample Use-cases

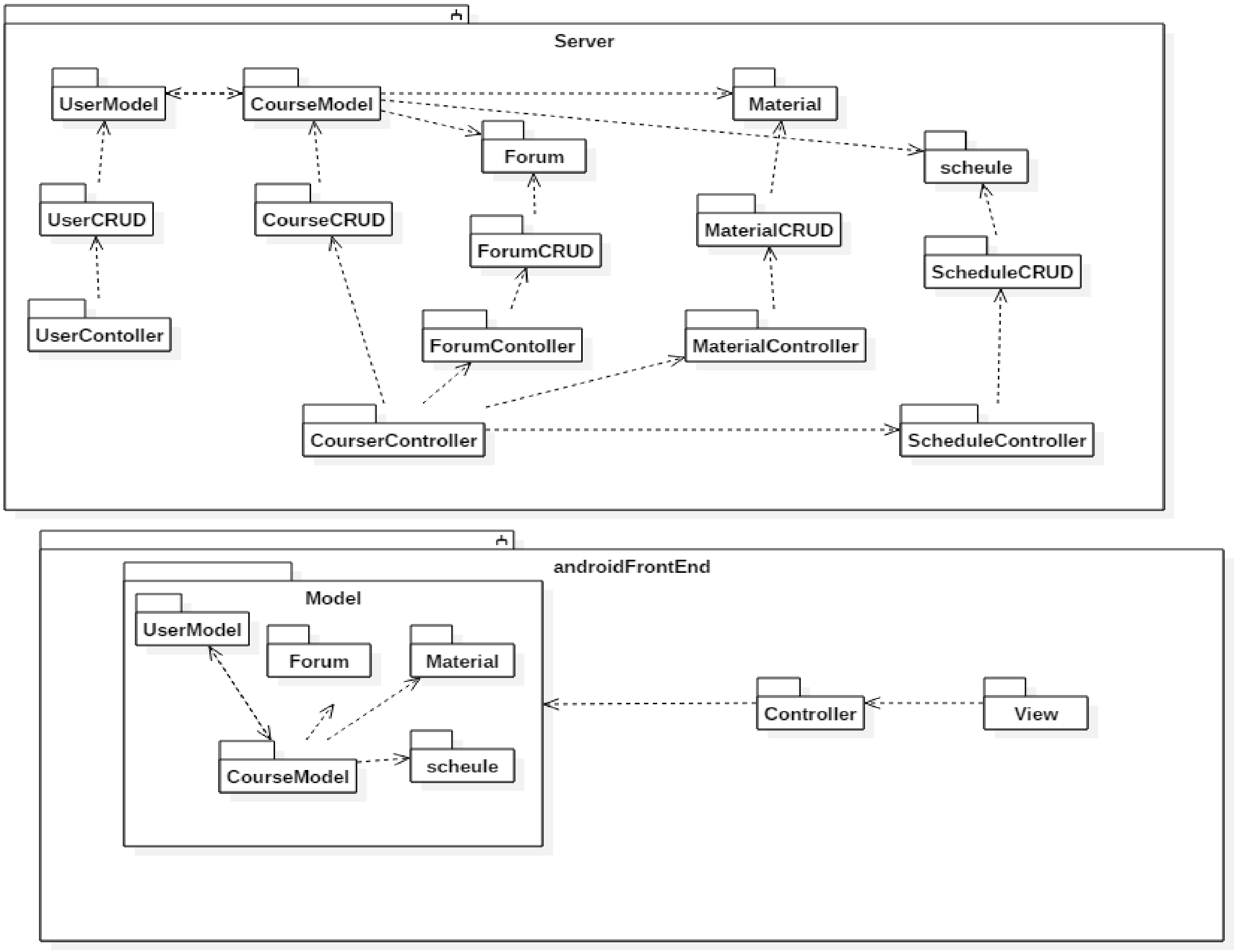
Use Case ID:	1	
Use Case Name:	Login	
Actors:	TA/Doctor – Student – Moderator student	
Pre-conditions:	User register in the app and validate his account	
Post-conditions:	________________________	
Flow of events:	**User Action**	**System Action**
	• User Selects to login	
		• System returns Login page
	• User enters his username • User enters his password	
		• System Verifies his information • System logs him in to his account
Exceptions:	**User Action**	**System Action**
	• User Enter username and Password	
		• username does not exist or password is wrong • System rejects the Login • System demands the user to enter correct username and password
Includes:		
Notes and Issues:	Login will identify if the account corresponds to TA/Doctor, Student or Moderator student.	

Use Case ID:	2	
Use Case Name:	TA/Doctor register	
Actors:	TA/Doctor	
Pre-conditions:	TA/Doctor has a valid academic E-mail	
Post-conditions:	TA/Doctor validate his account from the mail sent to his academic mail	
Flow of events:	**User Action**	**System Action**
	• TA/Doctor select to register	
		• System returns registration page
	• Enter the required information	
		• System check that neither of his data violate the specifications • System sent a verification code to the entered academic mail
	• TA/Doctor enters the verification code	
		• System match the verification code • creates the account
Exceptions:	**User Action**	**System Action**
	• TA/Doctor entered data that violate the specifications	
		• System recognize that the entered data violate the specifications • System demand the TA/Doctor to enter his data again in right manner.
Notes and Issues:	If TA/Doctor enter invalid E-mail he will not be able to validate his account so he will not be able to complete registration.	

Use Case ID:	3	
Use Case Name:	Course registration	
Actors:	Student	
Pre-conditions:	Login	
Post-conditions:	______________________	
Flow of events:	**User Action**	**System Action**
	• Student Show courses	
		• System return List of courses
	• Student select a specific course	
		• System return course information and some choices
	• Student Select register in this course	
		• System return a box to write course code
	• Student enter course code	
		• System check that the course code is right • System add this course to Student courses
Exceptions:	**User Action**	**System Action**
	• Student entered invalid course code	
		• System demand to enter course code again
Includes:		
Notes and Issues:	You should have the course code to register in the course	

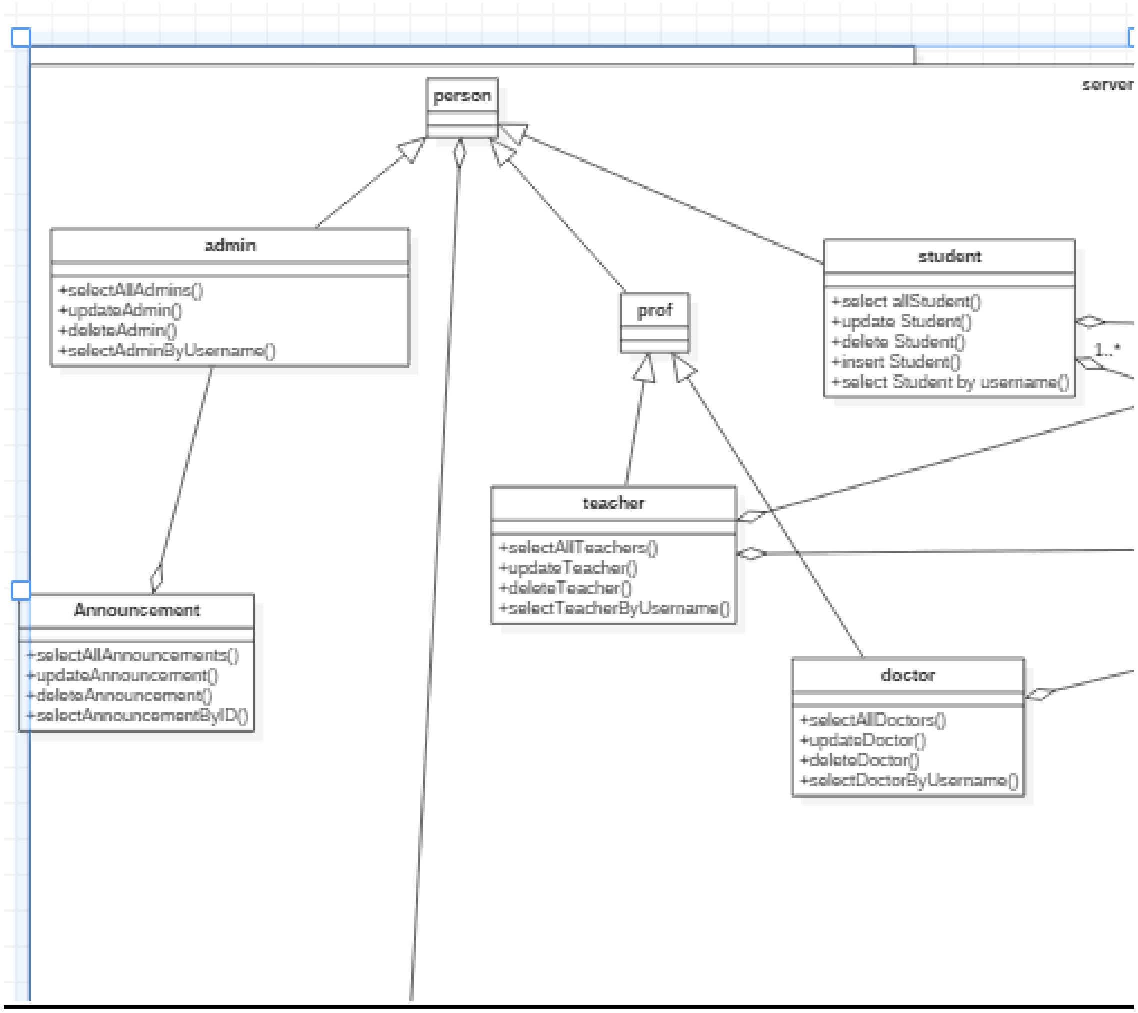
Use Case ID:	4	
Use Case Name:	Add Material	
Actors:	TA/Doctor – Moderator student	
Pre-conditions:	Login – Assign to this course	
Post-conditions:	______________________	
Flow of events:	**User Action**	**System Action**
	• User shows the courses he assigned to	
		• System return List of courses
	• User select the course he wanted	
		• System return course information and some choices
	• User select add material	
		• System return a box to upload material
	• User upload material	
		• System assign the uploaded material to the course • System send notification to the students assigned to this course
Exceptions:	**User Action**	**System Action**
Includes:		
Notes and Issues:	If you are student the add material button will not appear to you	

Component Diagram:

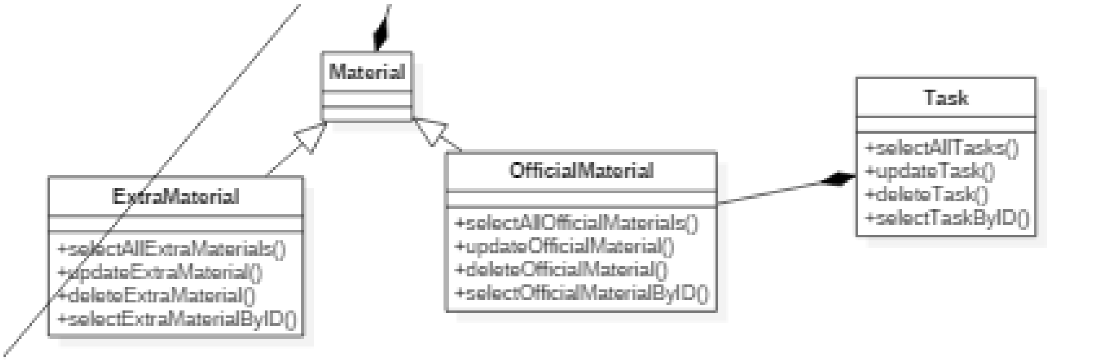


Class Diagram:

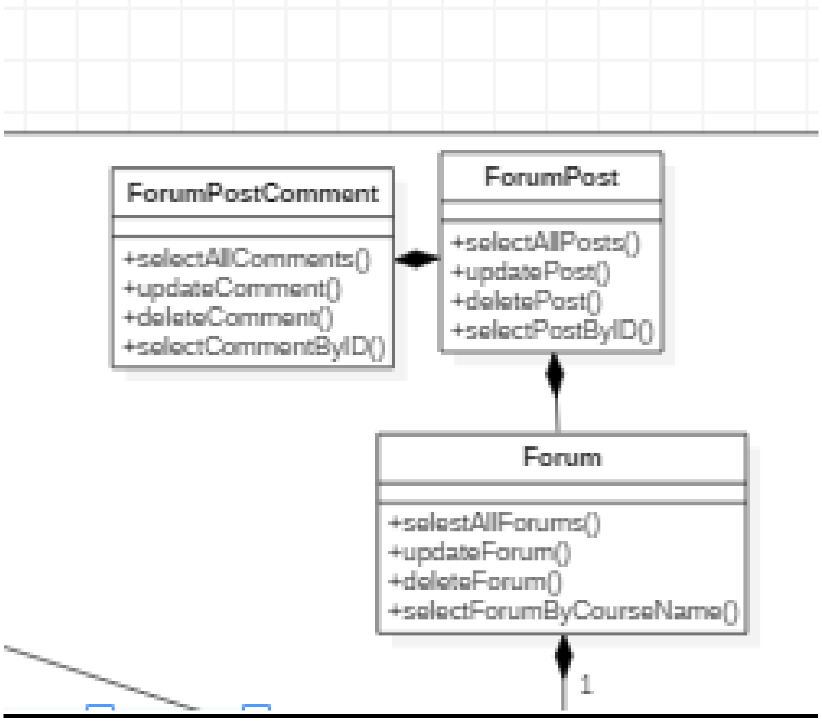
- User CRUD



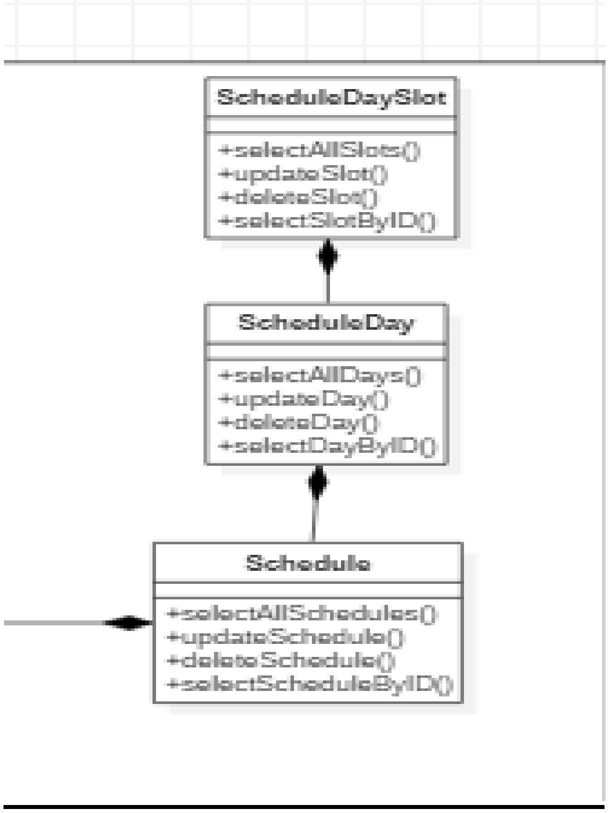
- Material Package


Material
Task
+selectAllTasks()
+updateTask()
+deleteTask()
+selectTaskByID()
ExtraMaterial
+selectAllExtraMaterials()
+updateExtraMaterial()
+deleteExtraMaterial()
+selectExtraMaterialByID()
OfficialMaterial
+selectAllOfficialMaterials()
+updateOfficialMaterial()
+deleteOfficialMaterial()
+selectOfficialMaterialByID()

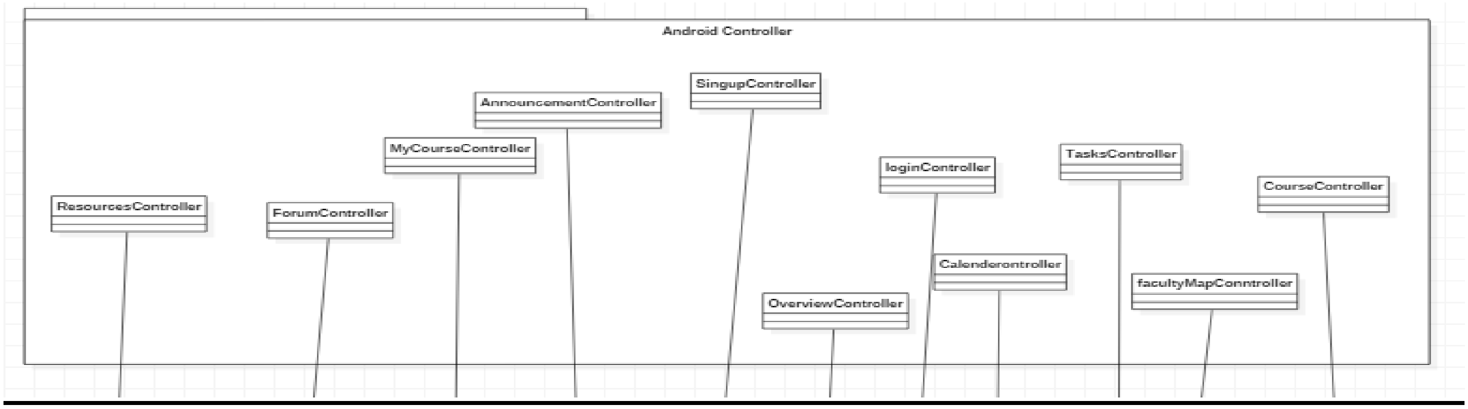
- Forum Package



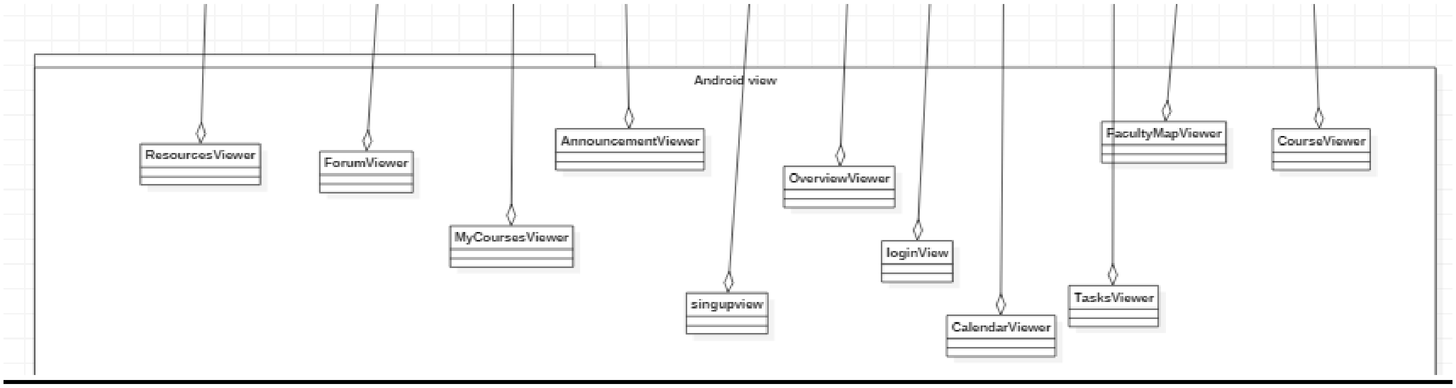
- Schedule Package



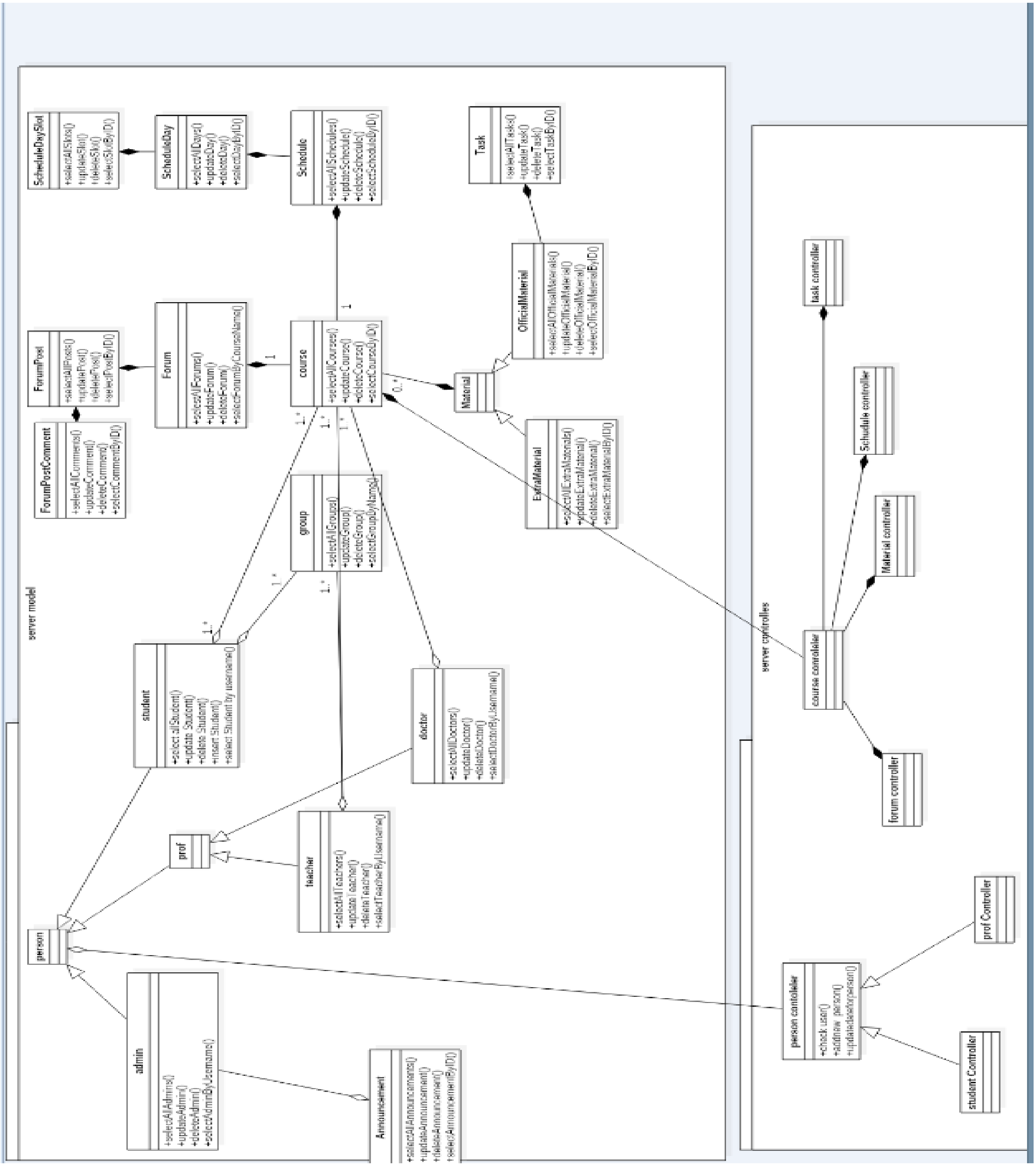
- Android Controller



- Android View

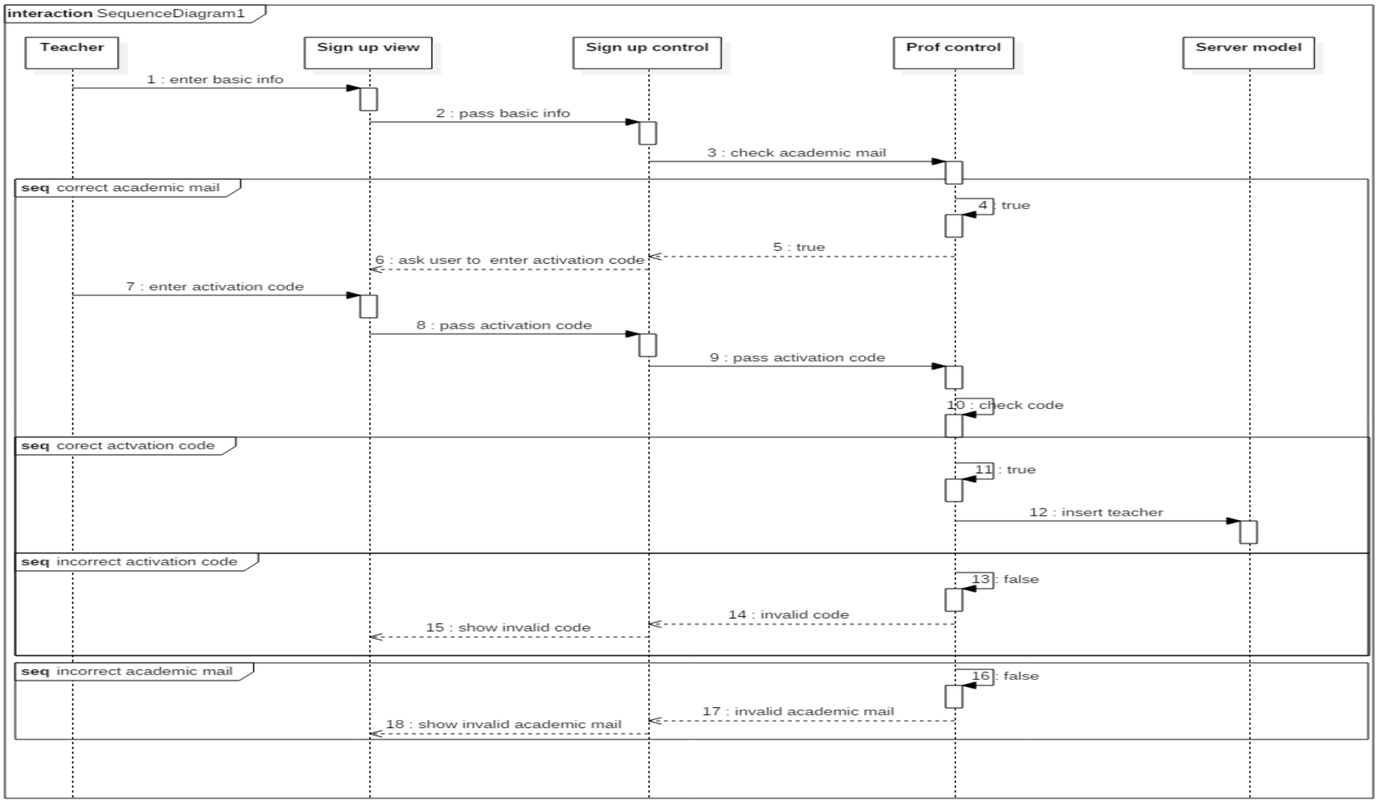


- Server Model & Controller

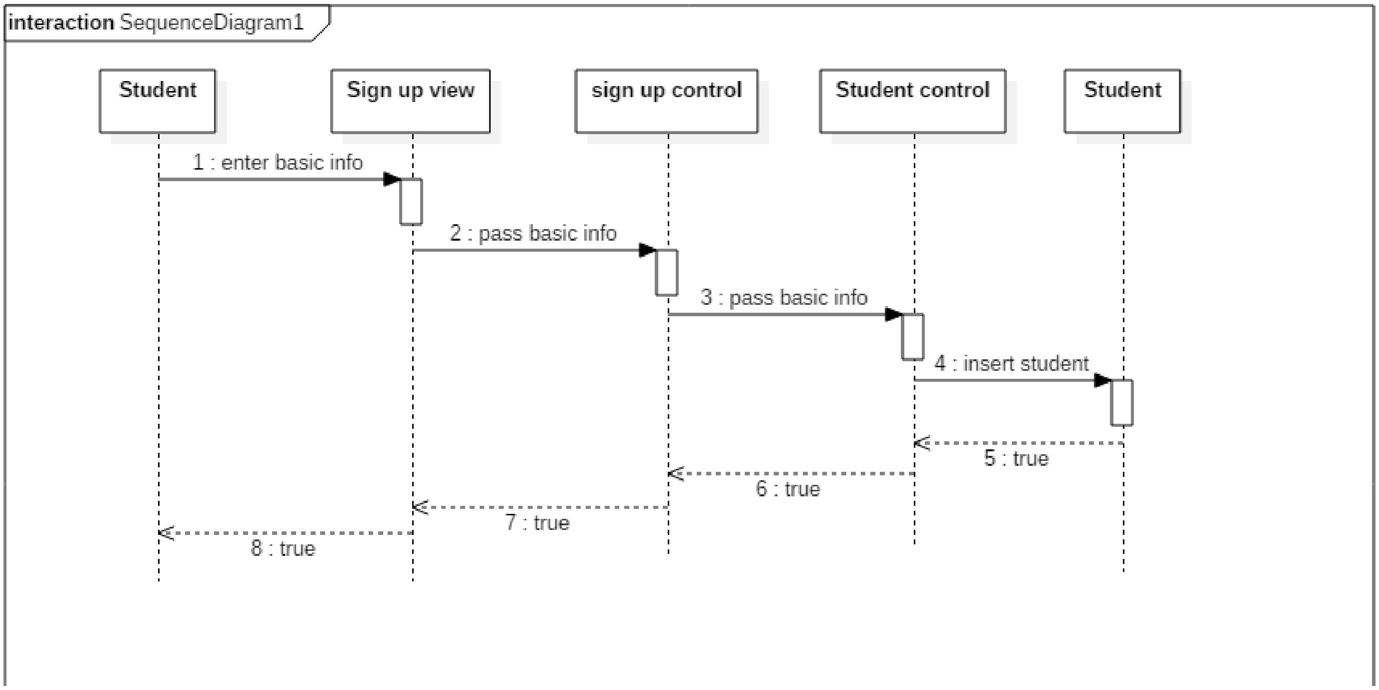


Sequence Diagram:

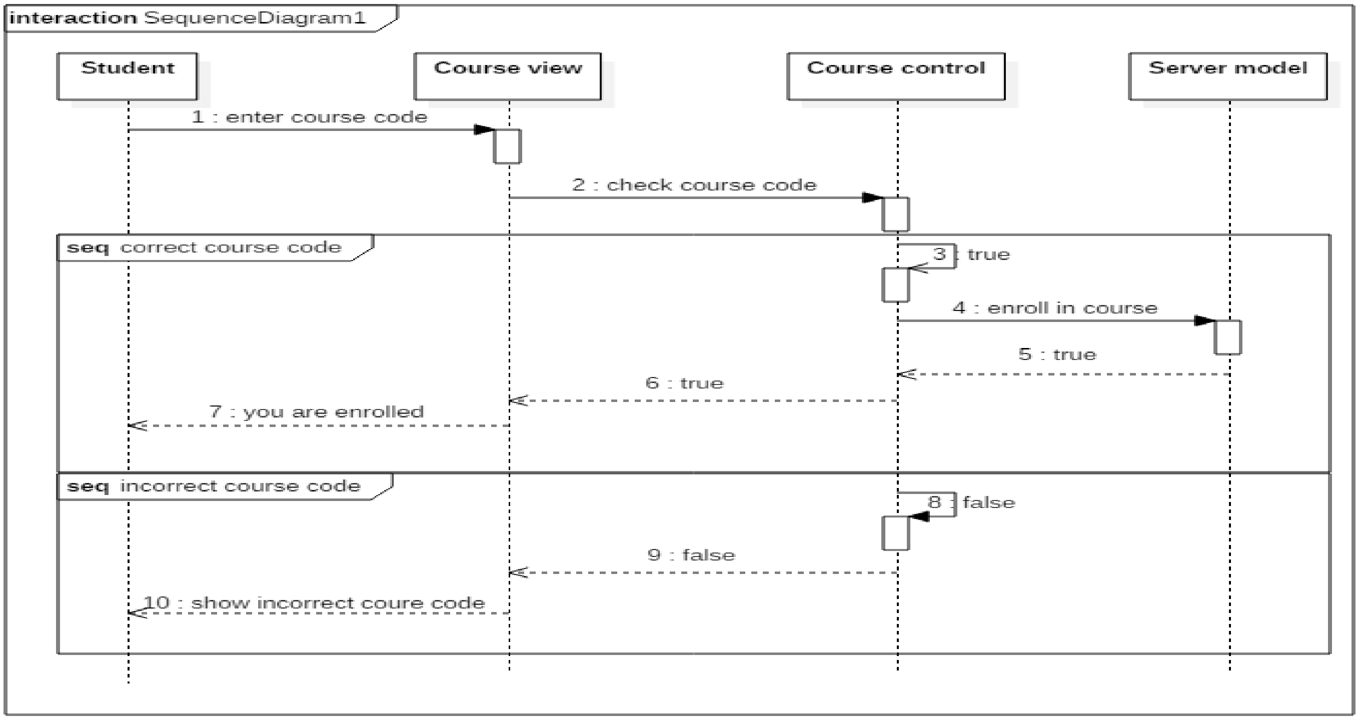
- ## Teacher Sign-up



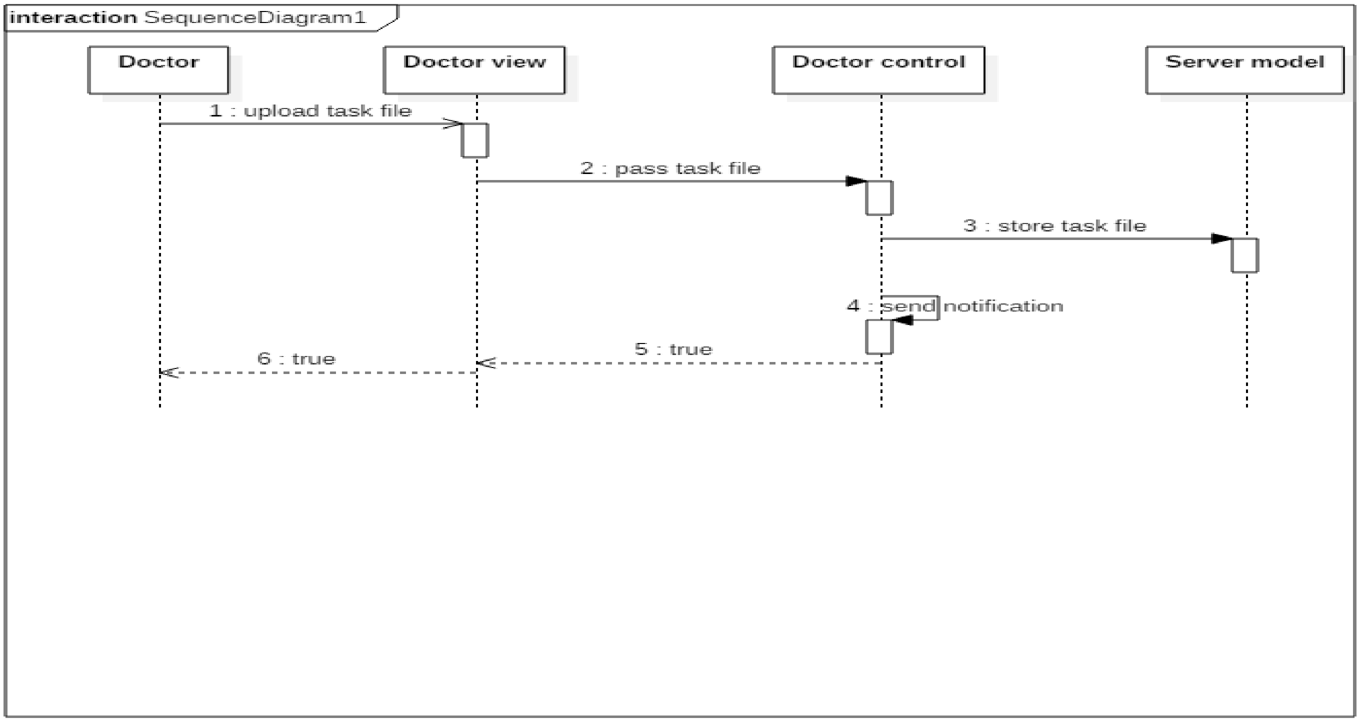
- ## Student Sign-up:



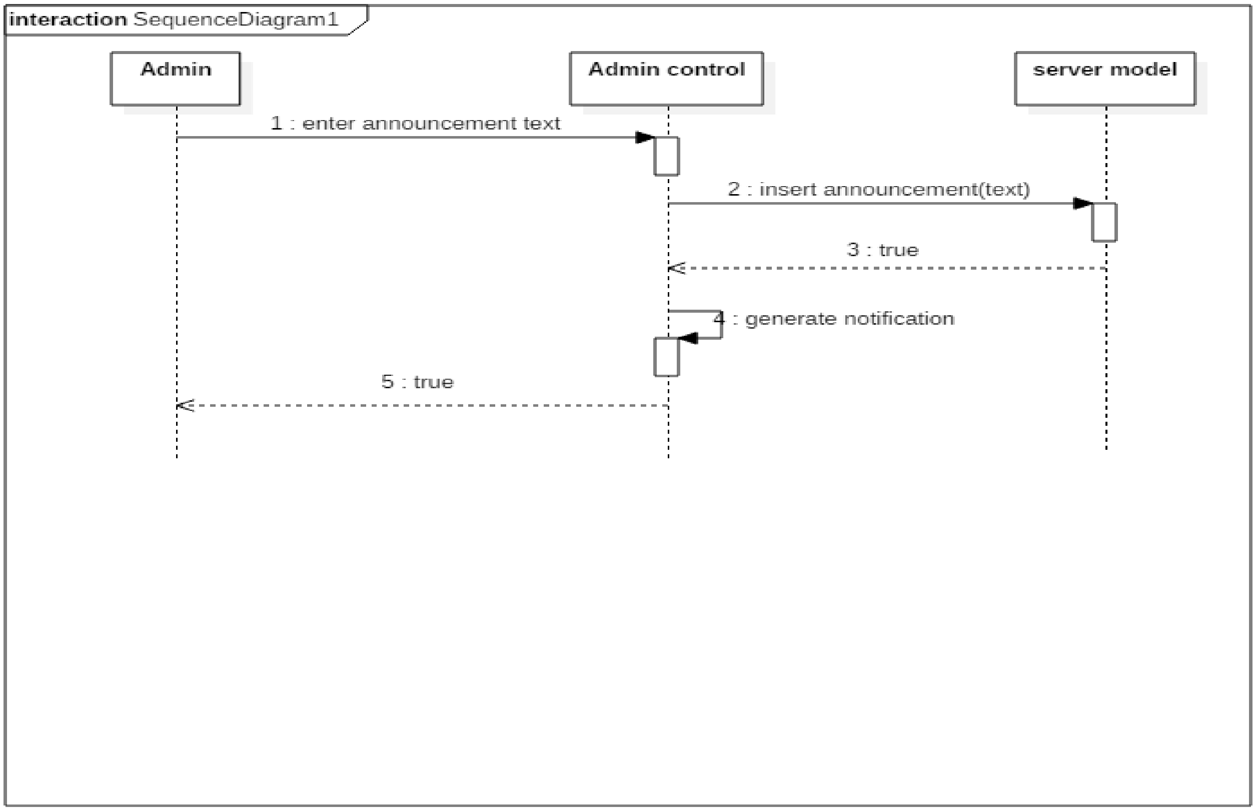
- Course Registration:



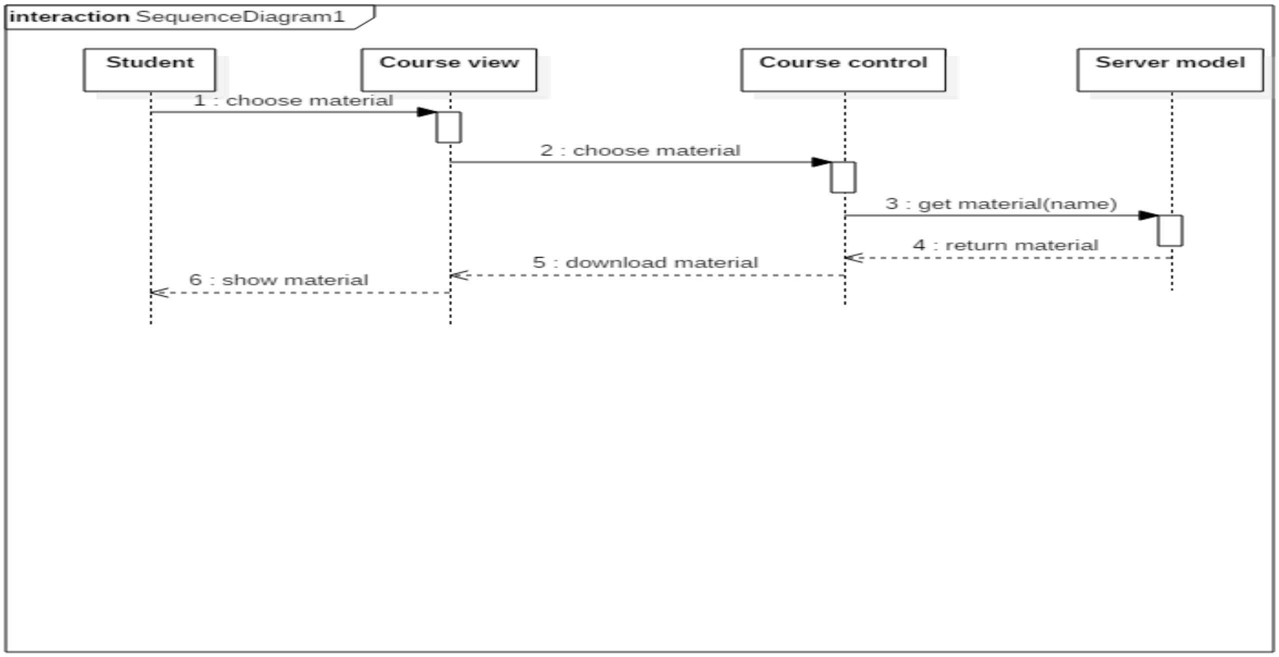
- Add Task:



- <u>Add Announcement:</u>

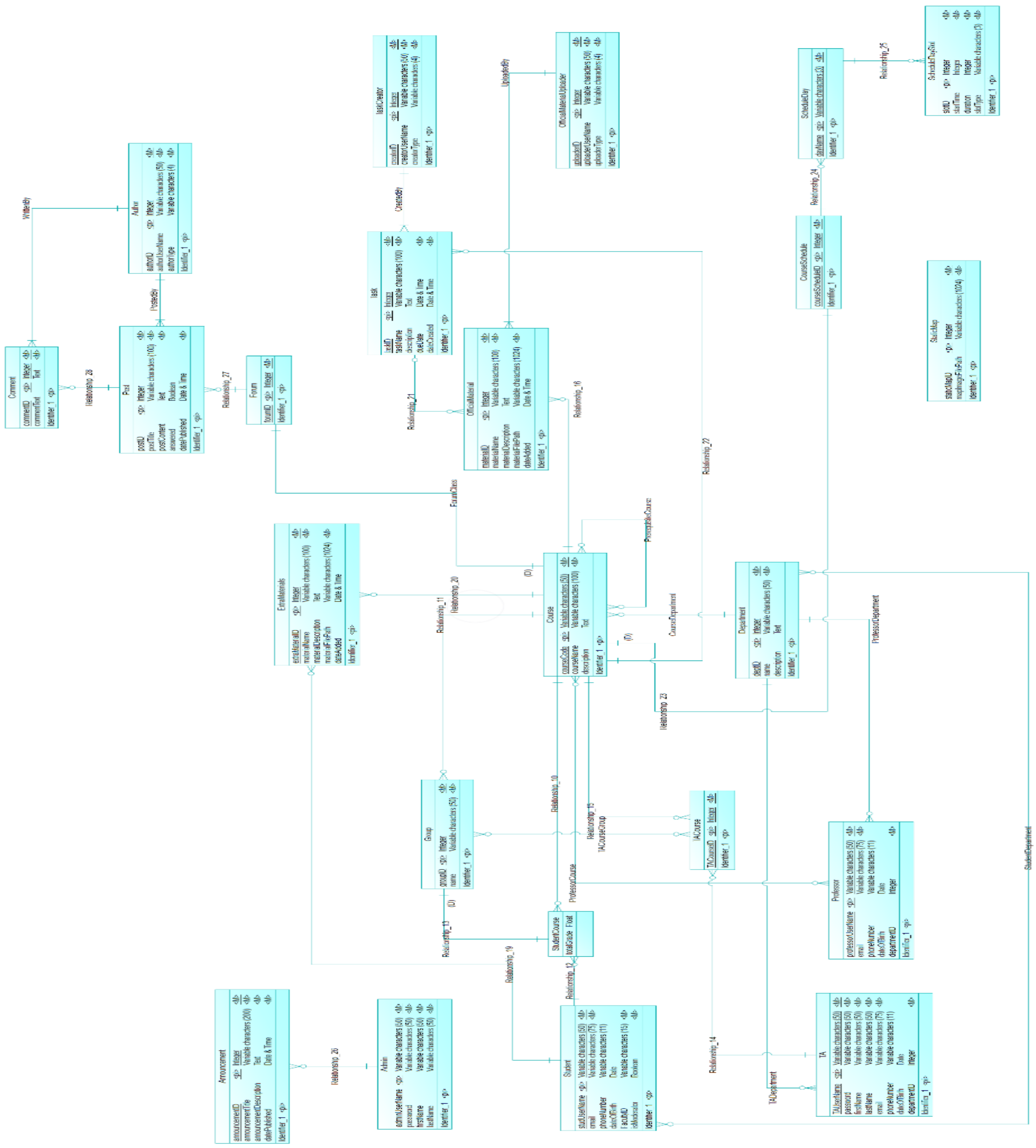

interaction SequenceDiagram1
Admin
Admin control
server model
1 : enter announcement text
2 : insert announcement(text)
3 : true
4 : generate notification
5 : true

- <u>Get Material:</u>

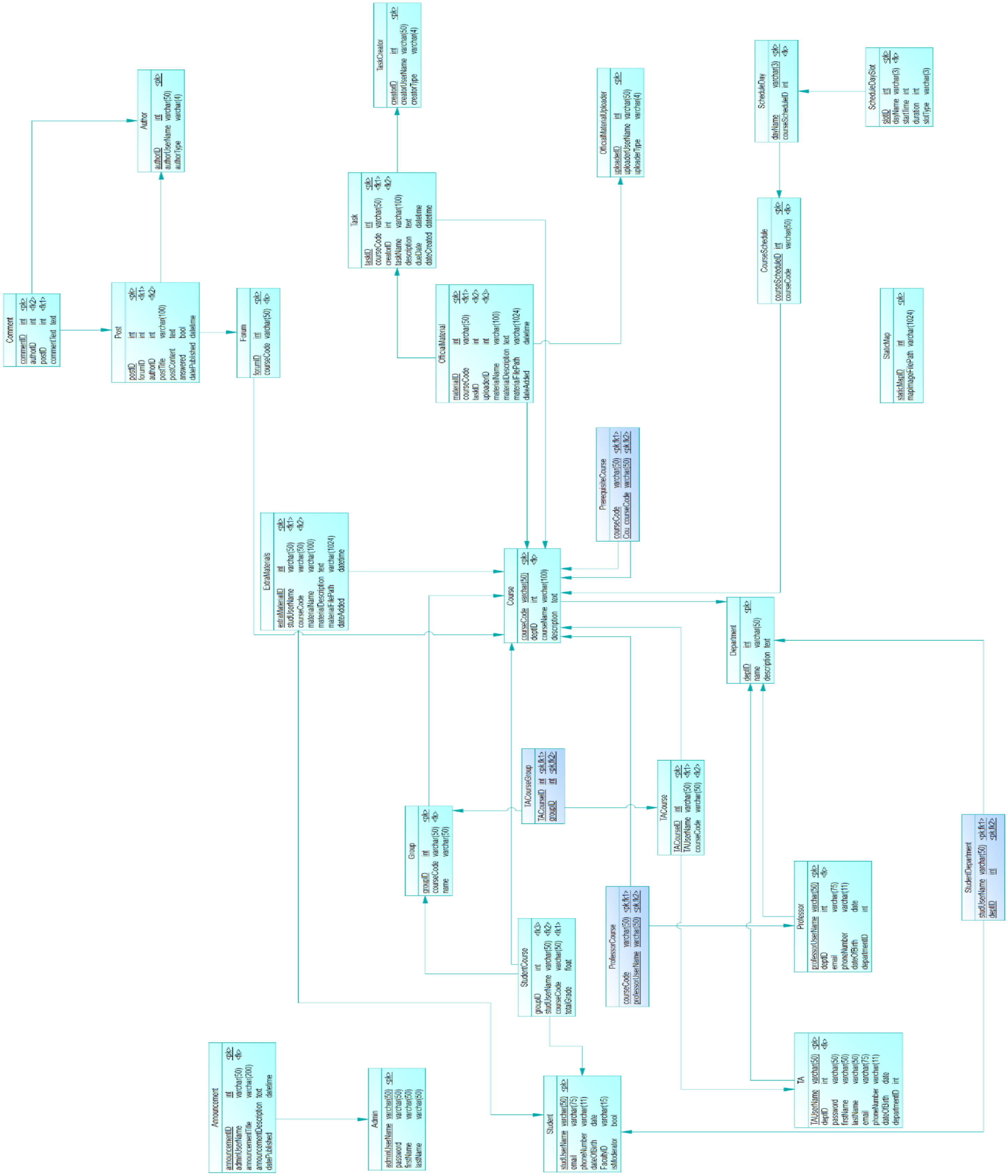


ERD:

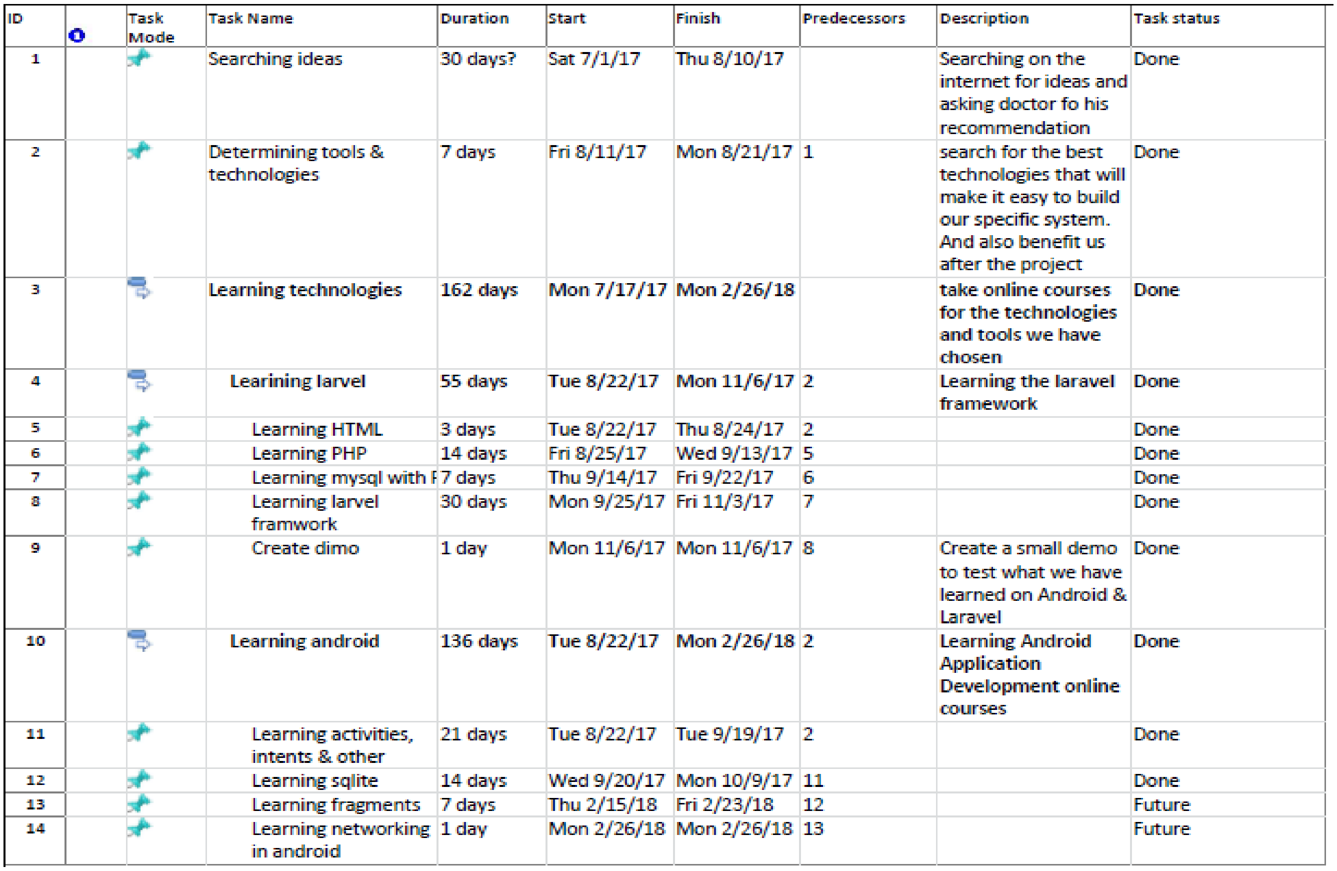
- ## Conceptual Data Model:



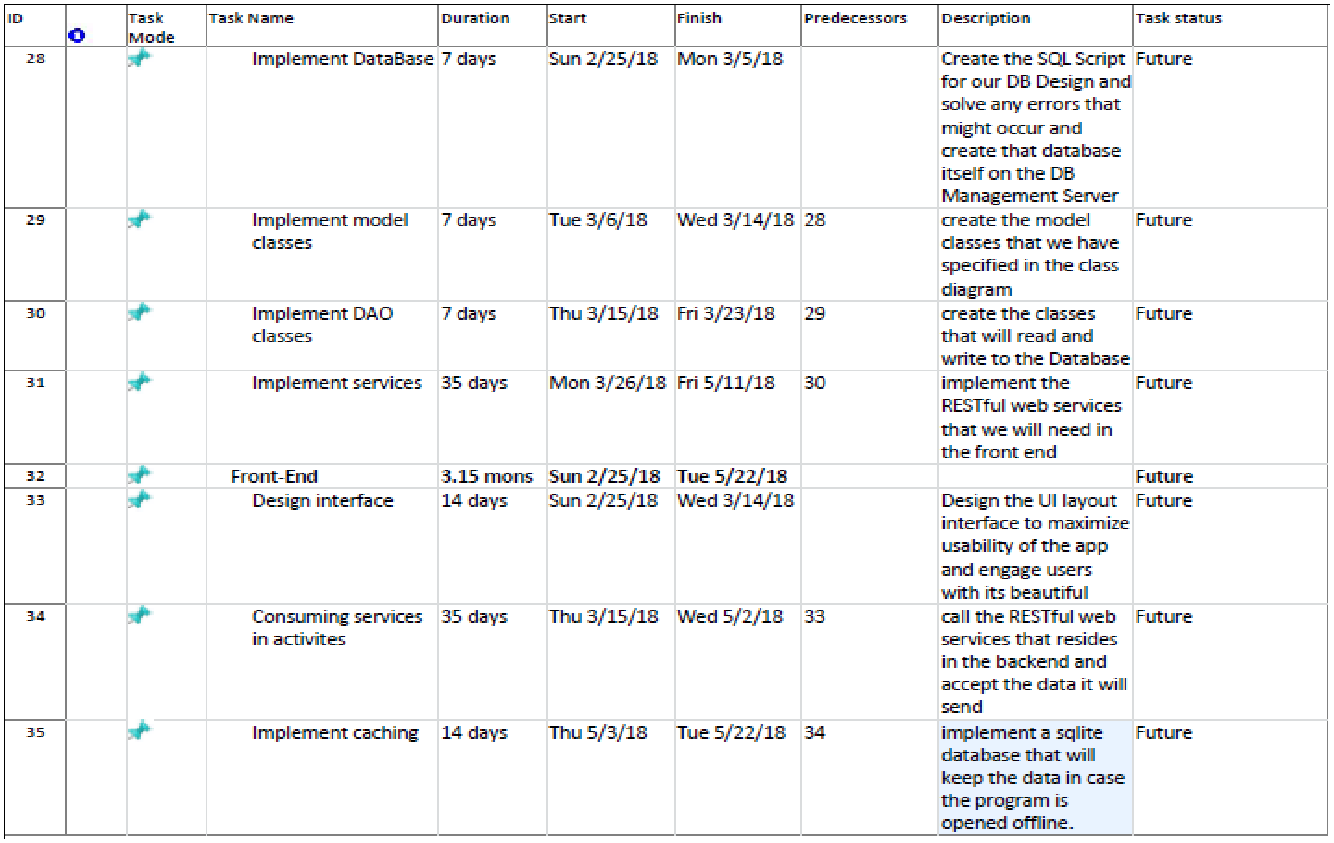
- <u>Physical Data Model:</u>



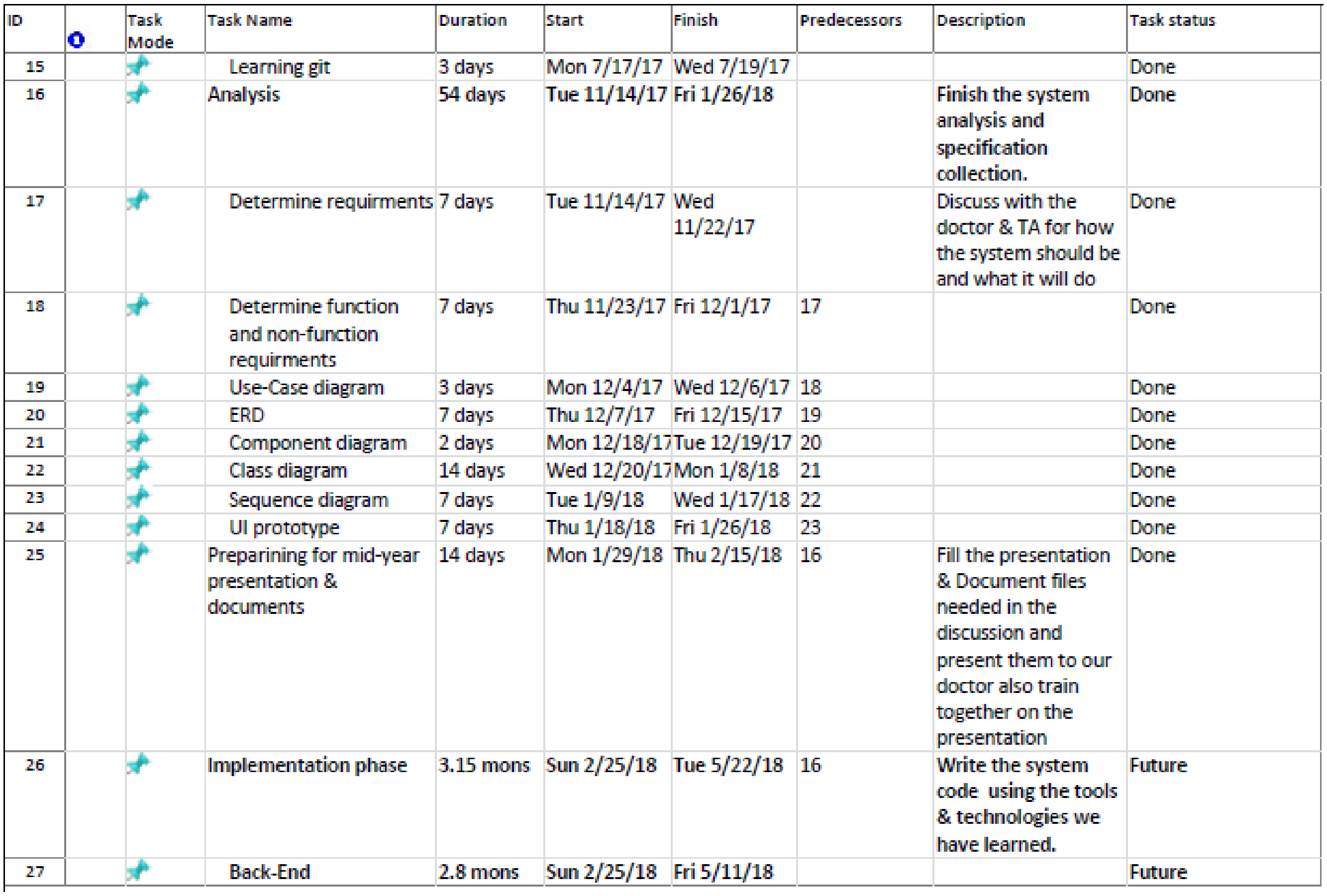
Work Plan:



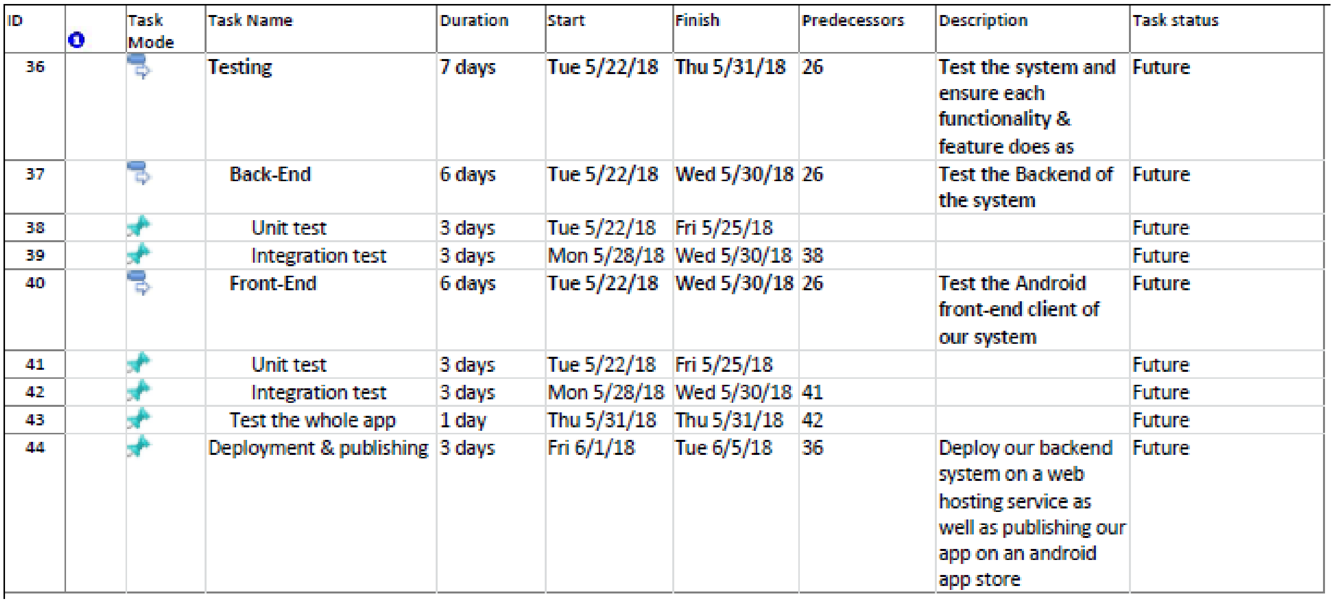
ID	ⓘ	Task Mode	Task Name	Duration	Start	Finish	Predecessors	Description	Task status
1		📌	Searching ideas	30 days?	Sat 7/1/17	Thu 8/10/17		Searching on the internet for ideas and asking doctor fo his recommendation	Done
2		📌	Determining tools & technologies	7 days	Fri 8/11/17	Mon 8/21/17	1	search for the best technologies that will make it easy to build our specific system. And also benefit us after the project	Done
3		➡	**Learning technologies**	**162 days**	**Mon 7/17/17**	**Mon 2/26/18**		**take online courses for the technologies and tools we have chosen**	**Done**
4		➡	**Learining larvel**	**55 days**	**Tue 8/22/17**	**Mon 11/6/17**	**2**	**Learning the laravel framework**	**Done**
5		📌	Learning HTML	3 days	Tue 8/22/17	Thu 8/24/17	2		Done
6		📌	Learning PHP	14 days	Fri 8/25/17	Wed 9/13/17	5		Done
7		📌	Learning mysql with P	7 days	Thu 9/14/17	Fri 9/22/17	6		Done
8		📌	Learning larvel framwork	30 days	Mon 9/25/17	Fri 11/3/17	7		Done
9		📌	Create dimo	1 day	Mon 11/6/17	Mon 11/6/17	8	Create a small demo to test what we have learned on Android & Laravel	Done
10		➡	**Learning android**	**136 days**	**Tue 8/22/17**	**Mon 2/26/18**	**2**	**Learning Android Application Development online courses**	**Done**
11		📌	Learning activities, intents & other	21 days	Tue 8/22/17	Tue 9/19/17	2		Done
12		📌	Learning sqlite	14 days	Wed 9/20/17	Mon 10/9/17	11		Done
13		📌	Learning fragments	7 days	Thu 2/15/18	Fri 2/23/18	12		Future
14		📌	Learning networking in android	1 day	Mon 2/26/18	Mon 2/26/18	13		Future



ID	ⓘ	Task Mode	Task Name	Duration	Start	Finish	Predecessors	Description	Task status
28		📌	Implement DataBase	7 days	Sun 2/25/18	Mon 3/5/18		Create the SQL Script for our DB Design and solve any errors that might occur and create that database itself on the DB Management Server	Future
29		📌	Implement model classes	7 days	Tue 3/6/18	Wed 3/14/18	28	create the model classes that we have specified in the class diagram	Future
30		📌	Implement DAO classes	7 days	Thu 3/15/18	Fri 3/23/18	29	create the classes that will read and write to the Database	Future
31		📌	Implement services	35 days	Mon 3/26/18	Fri 5/11/18	30	implement the RESTful web services that we will need in the front end	Future
32		📌	Front-End	3.15 mons	Sun 2/25/18	Tue 5/22/18			Future
33		📌	Design interface	14 days	Sun 2/25/18	Wed 3/14/18		Design the UI layout interface to maximize usability of the app and engage users with its beautiful	Future
34		📌	Consuming services in activites	35 days	Thu 3/15/18	Wed 5/2/18	33	call the RESTful web services that resides in the backend and accept the data it will send	Future
35		📌	Implement caching	14 days	Thu 5/3/18	Tue 5/22/18	34	implement a sqlite database that will keep the data in case the program is opened offline.	Future

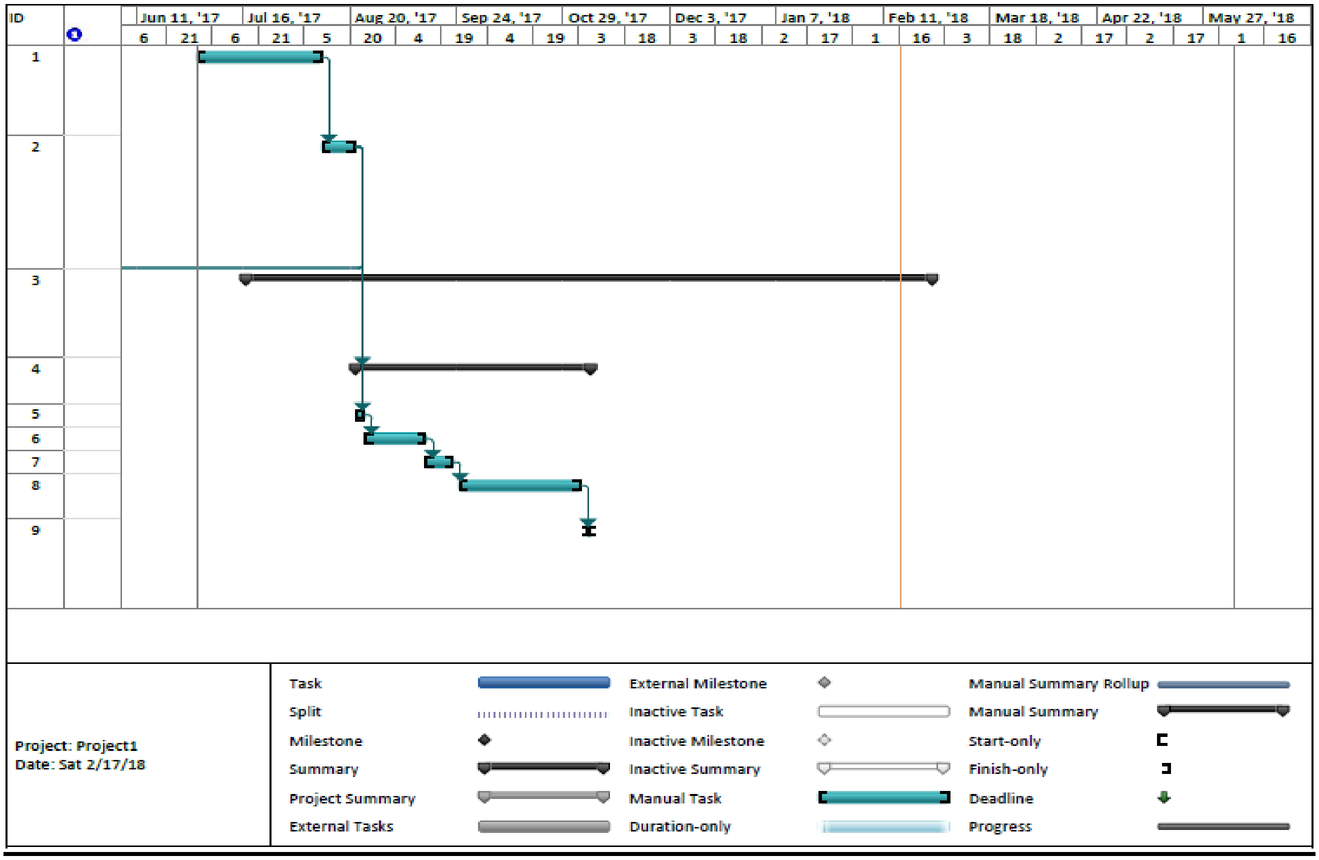


ID	ℹ	Task Mode	Task Name	Duration	Start	Finish	Predecessors	Description	Task status
15		📌	Learning git	3 days	Mon 7/17/17	Wed 7/19/17			Done
16		📌	**Analysis**	**54 days**	**Tue 11/14/17**	**Fri 1/26/18**		**Finish the system analysis and specification collection.**	**Done**
17		📌	Determine requirments	7 days	Tue 11/14/17	Wed 11/22/17		Discuss with the doctor & TA for how the system should be and what it will do	Done
18		📌	Determine function and non-function requirments	7 days	Thu 11/23/17	Fri 12/1/17	17		Done
19		📌	Use-Case diagram	3 days	Mon 12/4/17	Wed 12/6/17	18		Done
20		📌	ERD	7 days	Thu 12/7/17	Fri 12/15/17	19		Done
21		📌	Component diagram	2 days	Mon 12/18/17	Tue 12/19/17	20		Done
22		📌	Class diagram	14 days	Wed 12/20/17	Mon 1/8/18	21		Done
23		📌	Sequence diagram	7 days	Tue 1/9/18	Wed 1/17/18	22		Done
24		📌	UI prototype	7 days	Thu 1/18/18	Fri 1/26/18	23		Done
25		📌	Preparining for mid-year presentation & documents	14 days	Mon 1/29/18	Thu 2/15/18	16	Fill the presentation & Document files needed in the discussion and present them to our doctor also train together on the presentation	Done
26		📌	**Implementation phase**	**3.15 mons**	**Sun 2/25/18**	**Tue 5/22/18**	16	**Write the system code using the tools & technologies we have learned.**	**Future**
27		📌	**Back-End**	**2.8 mons**	**Sun 2/25/18**	**Fri 5/11/18**			**Future**



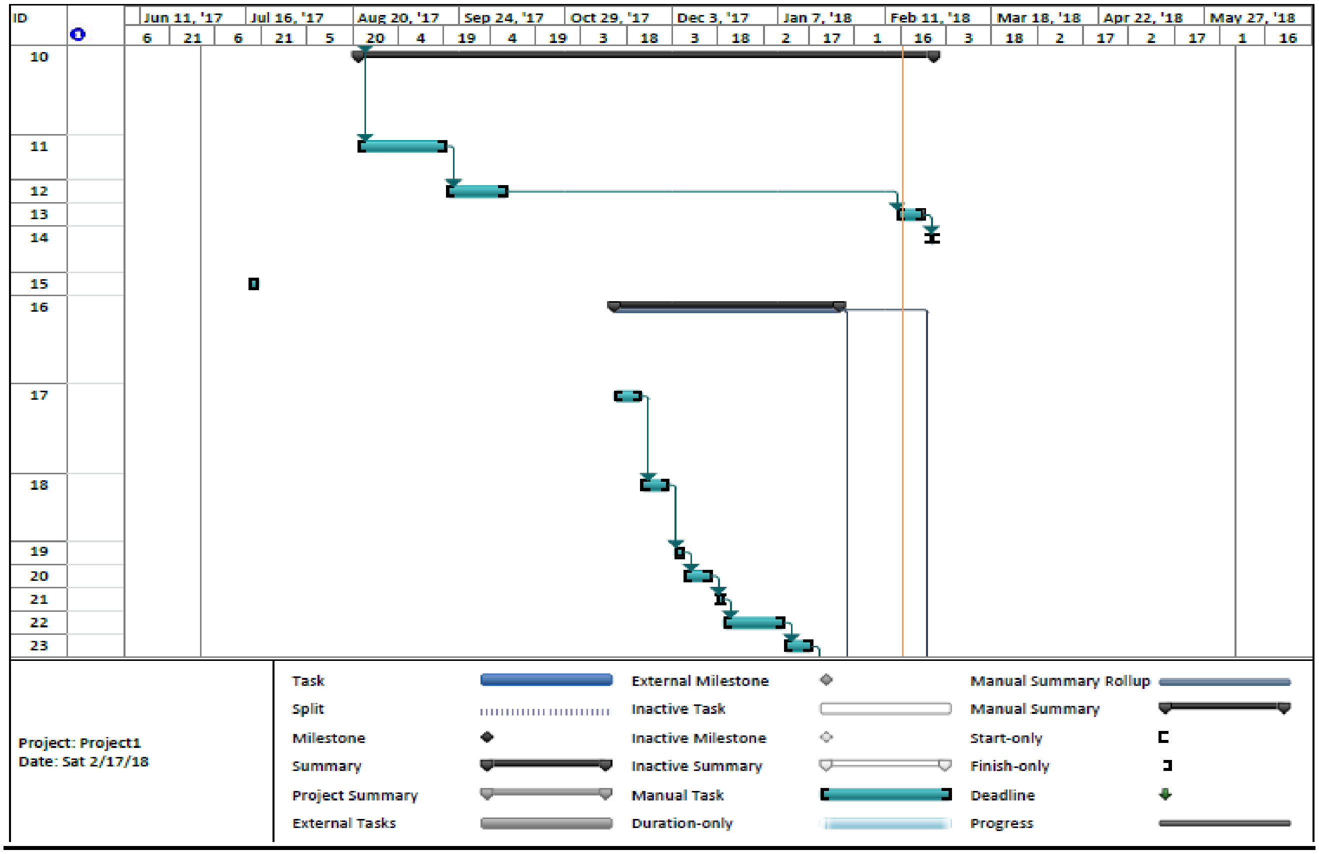
ID	ℹ	Task Mode	Task Name	Duration	Start	Finish	Predecessors	Description	Task status
36		⇨	**Testing**	**7 days**	**Tue 5/22/18**	**Thu 5/31/18**	26	**Test the system and ensure each functionality & feature does as**	**Future**
37		⇨	**Back-End**	**6 days**	**Tue 5/22/18**	**Wed 5/30/18**	26	**Test the Backend of the system**	**Future**
38		📌	Unit test	3 days	Tue 5/22/18	Fri 5/25/18			Future
39		📌	Integration test	3 days	Mon 5/28/18	Wed 5/30/18	38		Future
40		⇨	**Front-End**	**6 days**	**Tue 5/22/18**	**Wed 5/30/18**	26	**Test the Android front-end client of our system**	**Future**
41		📌	Unit test	3 days	Tue 5/22/18	Fri 5/25/18			Future
42		📌	Integration test	3 days	Mon 5/28/18	Wed 5/30/18	41		Future
43		📌	Test the whole app	1 day	Thu 5/31/18	Thu 5/31/18	42		Future
44		📌	Deployment & publishing	3 days	Fri 6/1/18	Tue 6/5/18	36	Deploy our backend system on a web hosting service as well as publishing our app on an android app store	Future

Gant chart:



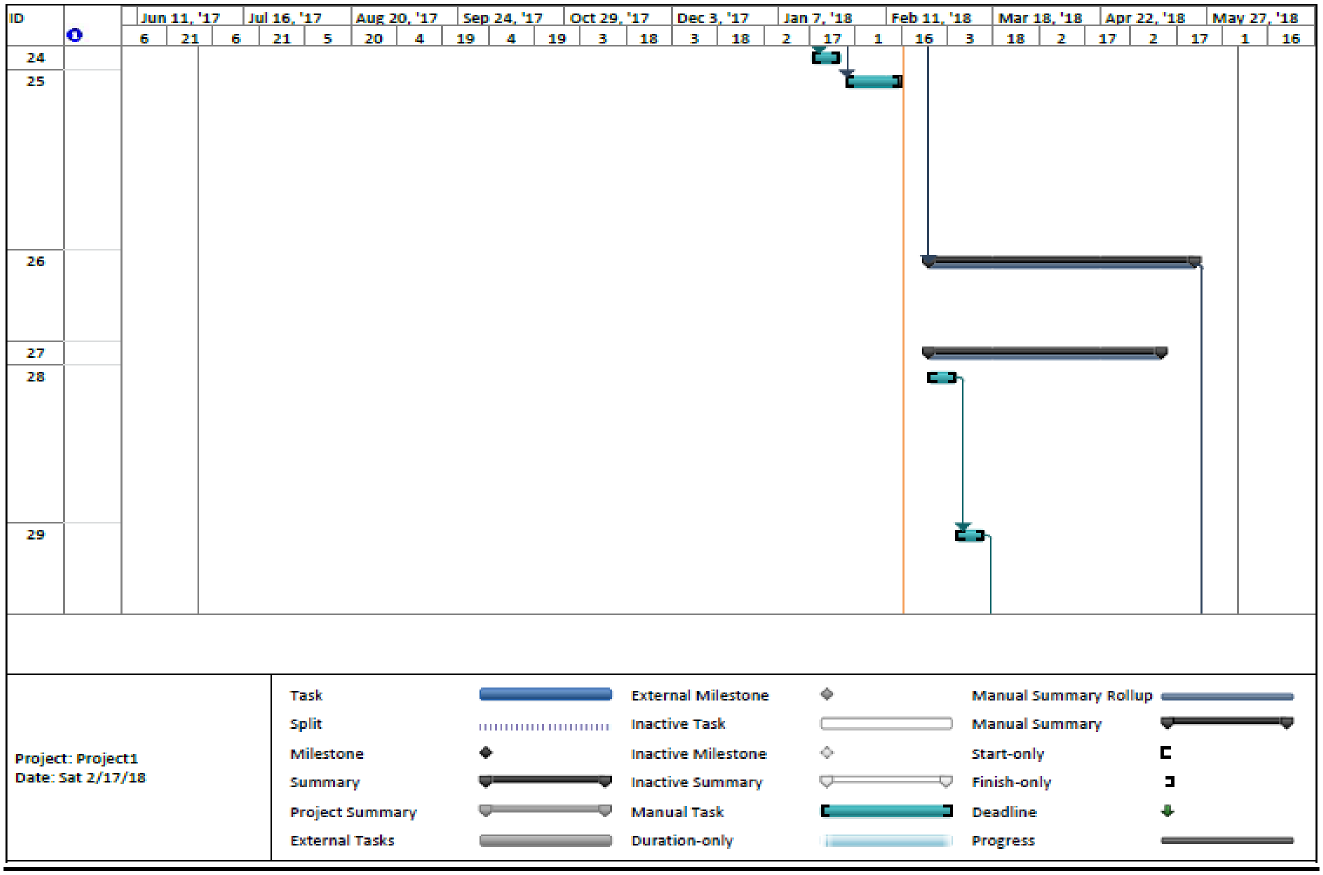
Project: Project1
Date: Sat 2/17/18

Task	▬	External Milestone	◆	Manual Summary Rollup	▬
Split	┈	Inactive Task	▭	Manual Summary	▬
Milestone	◆	Inactive Milestone	◇	Start-only	[
Summary	▬	Inactive Summary	▭	Finish-only	]
Project Summary	▬	Manual Task	▬	Deadline	⬇
External Tasks	▬	Duration-only	▬	Progress	▬

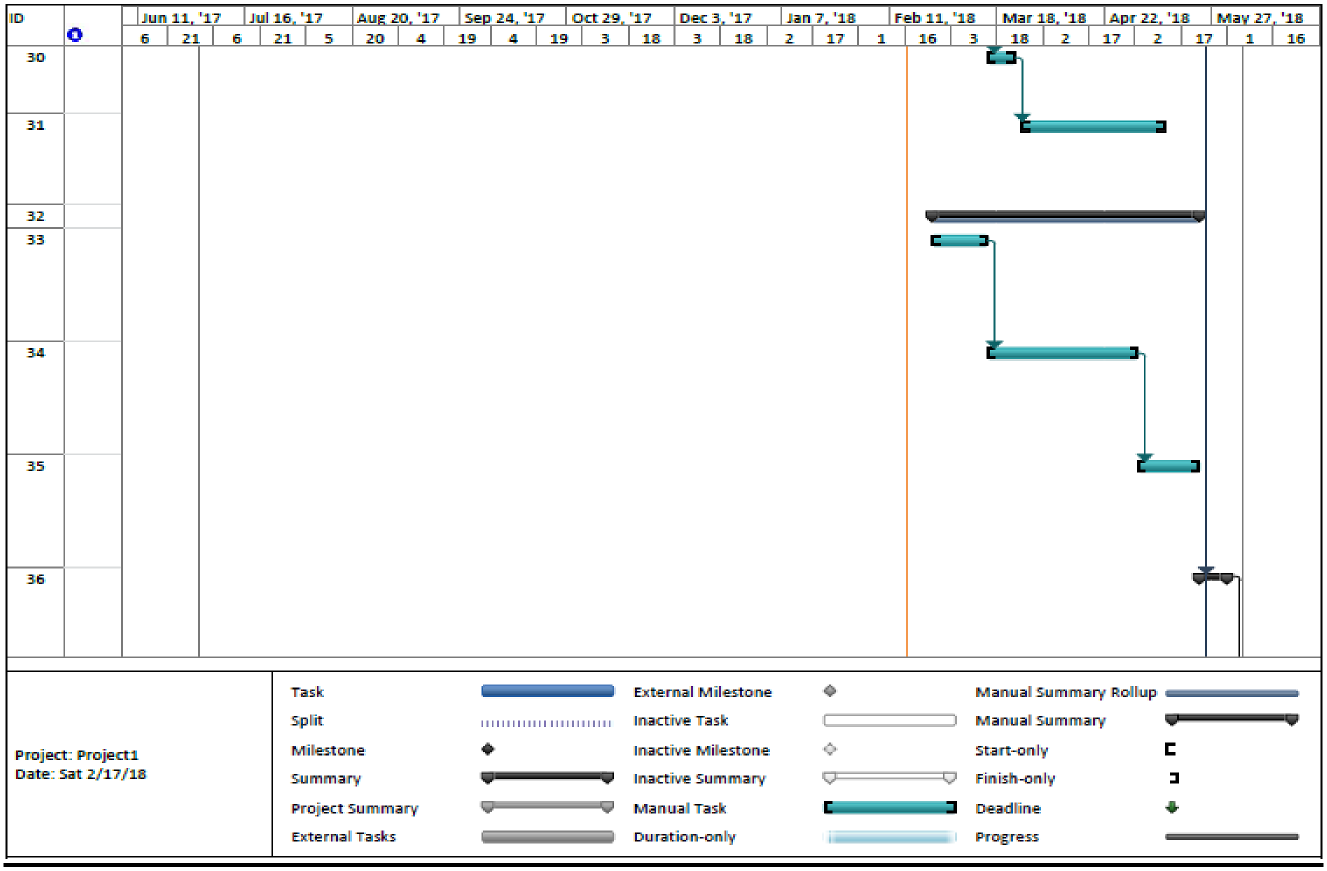


Project: Project1
Date: Sat 2/17/18

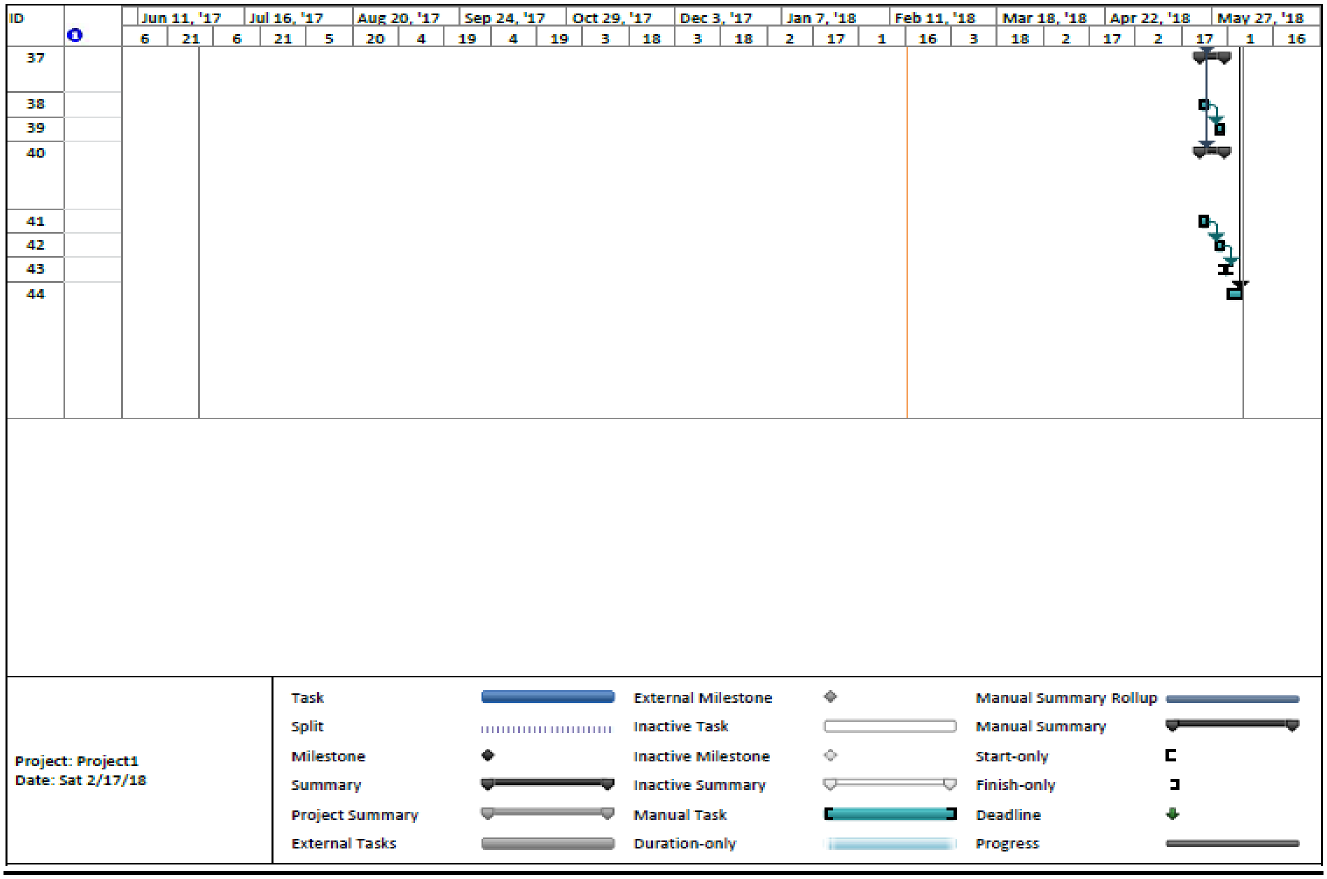
Task	▬	External Milestone	◆	Manual Summary Rollup	▬
Split	┈	Inactive Task	▭	Manual Summary	▬
Milestone	◆	Inactive Milestone	◇	Start-only	[
Summary	▬	Inactive Summary	▭	Finish-only	]
Project Summary	▬	Manual Task	▬	Deadline	⬇
External Tasks	▬	Duration-only	▬	Progress	▬



Project: Project1
Date: Sat 2/17/18



Project: Project1
Date: Sat 2/17/18



Project: Project1
Date: Sat 2/17/18

Task	▬	External Milestone	◆	Manual Summary Rollup	▬
Split	┆┆┆┆┆┆	Inactive Task	▭	Manual Summary	▬
Milestone	◆	Inactive Milestone	◇	Start-only	[
Summary	▬	Inactive Summary	▭	Finish-only	]
Project Summary	▬	Manual Task	▬	Deadline	⬇
External Tasks	▬	Duration-only	▭	Progress	▬