



**Group H**

**Software Requirements Specification**

**<MULTIMEDIA CASE DELIVERY SYSTEM>**

Version 1.0

Date: 12-11-2019

Created By:

Anu Yadav Kambalapally

Haritha Thiagu Kumar

Pradeep Salunke

Shreyas Hastantram Nagendra

Sri Karan Reddy Prodduturi

Document Release Note			
Project Name:		MULTIMEDIA CASE DELIVERY SYSTEM	
Document Name		Final Project	
Version:		1.0	
Release Date:		12-11-2019	
Prepared by:		Anu Yadav Kambalapally, Haritha Thiagu Kumar Pradeep Salunke Shreyas Hastantram Nagendra Sri Karan Reddy Prodduturi	
Document Revision History			
S No	Revision Date	Revision Description	Release Date
1	11-25-2019	Initial Release	11-25-2019
2	11-30-2019	Initial Release	11-30-2019
3	12-04-2019	Draft Release	12-04-2019
4	12-07-2019	Draft Release	12-07-2019
5	12-11-2019	Final Project	12-11-2019

#### Reviewers

Name	Version approved	Position	Date
T. Grandon Gill	1.0		

## Table of Contents

### Contents

<b>1</b>	<b>INTRODUCTION .....</b>	<b>4</b>
1.1	PURPOSE .....	4
1.2	SCOPE .....	4
1.3	OVERVIEW .....	4
<b>2</b>	<b>OVERALL DESCRIPTION .....</b>	<b>5</b>
2.1	PRODUCT PERSPECTIVE .....	5
2.2	PRODUCT FUNCTIONS .....	6
2.3	ROLES AND RESPONSIBILITIES .....	6
2.4	OPERATING ENVIRONMENT .....	7
<b>3</b>	<b>EXTERNAL INTERFACE REQUIREMENTS .....</b>	<b>7</b>
3.1	USER INTERFACES .....	7
3.2	SOFTWARE INTERFACES .....	7
3.3	COMMUNICATIONS INTERFACES .....	7
<b>4.</b>	<b>SYSTEM FEATURES .....</b>	<b>7</b>
4.1	FUNCTIONAL REQUIREMENTS .....	8
<b>5.</b>	<b>BUSINESS RULES:.....</b>	<b>8</b>
<b>6</b>	<b>ANNOTATED DIAGRAMS .....</b>	<b>9</b>
<b>7</b>	<b>MOCKUPS .....</b>	<b>28</b>
<b>8</b>	<b>ABOUT MVP .....</b>	<b>58</b>
<b>9</b>	<b>FUTURE SCOPE.....</b>	<b>59</b>

## 1 INTRODUCTION

Introduction part of Software Requirements Specification (SRS) document enables user to understand the system by taking everything into consideration. It gives an idea on purpose, scope, definitions, acronyms, abbreviations, references and overview of the SRS. The purpose of this document is to assemble the requirements, analyze them and give a clear picture of **Multimedia Case Study in an Online Environment** by interpreting the issues and challenges in detail. It also focuses on the potential requirements of stakeholders and their needs while elaborating the high-level features of the product.

### 1.1 Purpose

The purpose of the document is to collect and analyze all assorted ideas that have come up to define the system, its requirements with respect to customer's needs. Also, we shall predict and sort out how we hope this product will be used in order to gain a better understanding of the project, outline concepts that may be developed later, and document ideas that are being considered, but may be discarded as the product develops.

In short, the purpose of this SRS document is to provide a detailed overview of our software product, its parameters and goals. This document describes the project's target audience and its user interface, hardware and software requirements. It defines how our client, team and audience see the product and its functionality. It also helps designers, developers and testers in software design, development and testing of the system.

### 1.2 Scope

Primarily, the scope pertains to the Multimedia Case Study features for making Multimedia Case Study available online. It purely focuses on bringing the Case Study online with accessibility to multimedia like videos, audios, animations etc. The main feature of the multimedia case study system is to help the students understand the case with help of videos, audios, animations etc. It also supports group projects, chat box for queries and discussions, grades can be displayed after completion of case study. In the future releases of the application we can expect it to integrate with university learning management systems like canvas.

### 1.3 Overview

The remaining sections of this document provides a general description, including characteristics of the users of this project, the functional requirements of the product. General description of the project is discussed in section 2 of this document such as Product Perspective, Product Functions, Roles and Responsibilities and Operating Environment. Section 3 is about the External Interface Requirements of the product. Section 4 is about Functional requirements and Response sequence. Session 5 is about Software Quality and Session6 is about Other Requirements.

## 2 OVERALL DESCRIPTION

### 2.1 Product Perspective

Grandon Gill, professor in the Information Systems and Decision Sciences department at the University of South Florida, had been authoring paper-based discussion case studies for mainly intended use in the classroom. The results had been very successful, both from a professional and from a student perspective.

The discussion of case studies in classroom involves student's engagement and produces lots of energy with better understanding of the case but the number of students is limited per each classroom.

In order to provide the case discussion to large number of students there emerged a need to build an online application

But the disadvantage of online case study is that students will not have same level of focus and concentration as of the classroom discussion.

So, the idea is to create an online case study application which keeps the students focused and engaged and is as interesting as the classroom discussion.

The system should be able to interact with the students and should allow group analysis of case study.

The system should be able to deliver highly interactive multimedia cases that could be analyzed by students, individually or in teams, entirely online with little or no instructor intervention.

The project is a three-tier architecture which contains the layers discussed below.

- **Presentation Layer:** It is also known as Client layer. Top most layer of an application. This is the layer the users see when they access the application. The main functionality of this layer is to communicate with Application layer. This layer passes the information which is given by the user in terms of keyboard actions, mouse clicks to the Application Layer.
- **Application Layer:** It is also known as Business Logic Layer which is also known as logical layer. In our application once user clicks on the login button, Application layer interacts with Database layer and sends required information to the Presentation layer. It controls an application's functionality by performing detailed processing. This layer acts as a mediator between the Presentation and the Database layer. Complete business logic will be written in this layer.
- **Data Layer:** The data is stored in this layer. Application layer communicates with Database layer to retrieve the data. It contains methods that connects the database and performs required action e.g.: insert, update, delete etc. The database which stores the multimedia content of the case study and user responses.

## 2.2 Product Functions

- Mentioned below are the major functionalities
  - a) This product is used for serving the below purposes
    - Students registers for the subject in Learning Management System, Admin will fetch the same from LMS and inserts into multimedia case study system
    - Students will be using the application to view the videos individually and communicate with each other on the group analysis of case study screen. The content can be in any form of the multimedia like videos, text, jpeg, hyperlink.
    - The students should be able to answer the quizzes or surveys which are maintained by author.
    - Author should be able to add, modify and delete the multimedia content.
    - Facilitator should be given access to embed logic into the system and schedule the case study for the students.
  - b) Through the UI/UX the users can interact with the application.

## 2.3 Roles and Responsibilities

- a) The user groups of this application are
  - Students
  - Facilitators
  - Authors
  - Administrators
- b) **Admin:**
  - Have the access to give permissions to the users according to their roles.
  - Based on the entitlements maintained for the user groups by the admin, the users will be able to access specific functionalities in the application.
  - Admins have the access to reset the passwords of the users upon request.
  - Admin will have access to all the student details in the Learning Management system database and inserts into Multimedia Case system.
- c) **Author:**
  - Has access to upload, delete and modify the multimedia.
  - Has the access to create, modify and delete quizzes and surveys?
- d) **Facilitator**
  - Should able to embed logic into the system
  - Grade the quizzes taken by the students
  - Form the groups of students
  - Schedule the Multimedia case study
  - Access to view the discussion
- e) **Student**
  - Should be able to view dashboard containing their videos
  - Should be able to answer the quizzes and surveys
  - Has access to participate in case discussions
  - Should be able to view the grades
  - Has access to register into Learning Management System

## 2.4 Operating Environment

- The application should be platform independent and should be accessed on any devices (mobiles, laptop, tablet, etc....)

## 3 EXTERNAL INTERFACE REQUIREMENTS

### 3.1 User Interfaces

- The MVP requires all the user interfaces to be responsively suitable for multiple screen types providing seamless experience.
- Four interfaces need to be created, one for each user and the level of simplicity depends upon the user.
- The user experience should remain the same in all the platforms and user interface should follow a common theme.
- Every upload to the system should be facilitated by the drag and drop interface.

### 3.2 Software Interfaces

- The tech stack includes UI, backend, database, and analytics system, API's need to be created for each part with swagger implementation.
- Each operation should be handled by a separate microservice which improves the modularity and interfaces should be made available between them.
- Each microservice should have the ability to interact with other learning management systems like canvas and blackboard.
- Messaging queues should be implemented to provide an interface for the analytics system for continuous data flow.
- The analytics system should provide an interface for external people to query the resources.

### 3.3 Communications Interfaces

- The application requires external communication interfaces like Gmail to communicate the messages.
- Cloud notifications need to be enabled for mobile applications

## 4. SYSTEM FEATURES

Functional specifications capture the intended behavior of the system. This behavior may be expressed as services, tasks or functions the system is required to perform. It is useful to distinguish between the baseline functionality necessary for any system to compete in that product domain, and features that differentiate the system from competitors' products.

#### 4.1 Functional Requirements

- Students should be able to register themselves in the Learning Management system.
- Only Administrator should be able to see the student details in the Learning Management System.
- Only Administrator should be able to provide permission to the users according to their roles.
- Only Administrator should be able to make the student information available in the Facilitator's Interface.
- Only Facilitator will have Privileges to Form groups, View quizzes, Embed Logics (for the flow of content), Schedule Case studies, access to Discussion boards and have Student name, id, email address.
- Students should be able to view content, participate in discussions, upload submission documents, view grades, participate in surveys.
- After the Student submission, Facilitator should be able to view the submission and grade it.
- Facilitator/Administrator won't be able to participate in the case discussions.
- Author should be able to upload, delete and edit videos and quiz content.

#### 5. BUSINESS RULES:

Students, Facilitator, Author and Administrator are the different types of people who will be using the application.

**Students:** should be given access to only

- Dashboard containing their videos,
- Answering the quizzes and surveys
- Participate in case discussions.
- View the grades
- Upload submission documents.
- Has access to register into the Learning Management System.

**Facilitator:** should be given access to only

- Conduct quizzes.
- Grade the quizzes.
- Form groups.
- Embed logic
- Schedule the Multimedia case study.
- Access to view the group discussion board.



**Admins:**

- Have the access to give permissions to the users according to their roles.
- Based on the entitlements maintained for the user groups by the admin, the users will be able to access specific functionalities in the application.
- Admins have the access to reset the passwords of the users upon request.
- Admin will have access to all the student details in the Learning Management system database and inserts into Multimedia case system.

**Author:**

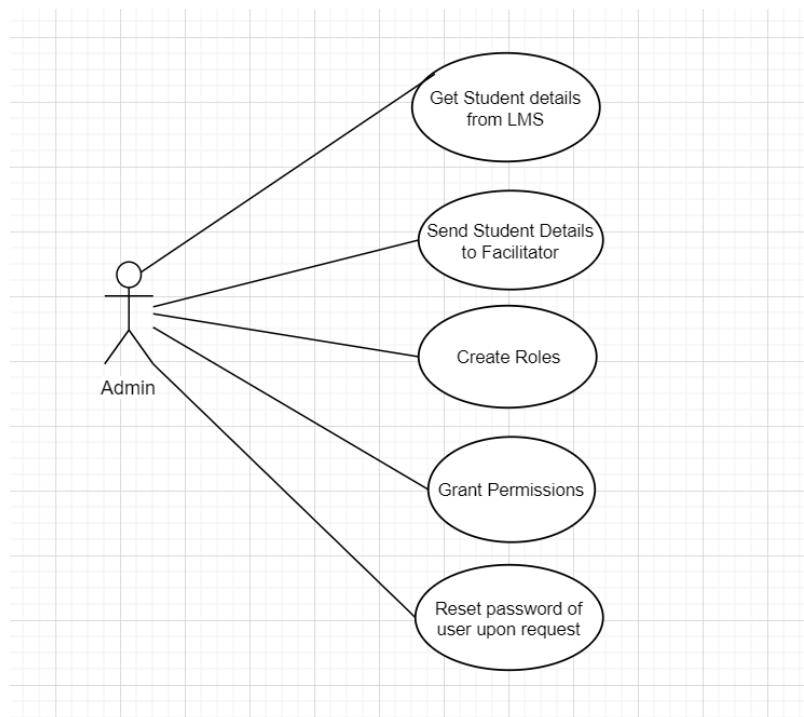
- Has been given access to upload, delete and edit the videos quiz content and surveys.

## 6 ANNOTATED DIAGRAMS

### 6.2 Use case diagrams:

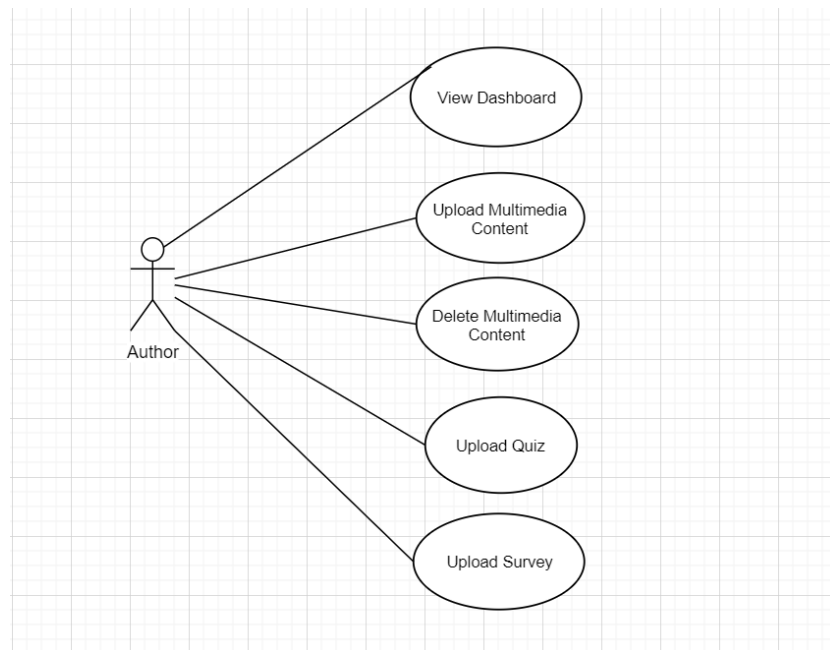
**Admin:**

Admin has the privileges to access learning management system and fetch the student details. He can also add user details and modify the user details from LMS to Multimedia system. He can create roles for all the users and grants permissions to them. He can also reset the passwords upon request from authorized users.



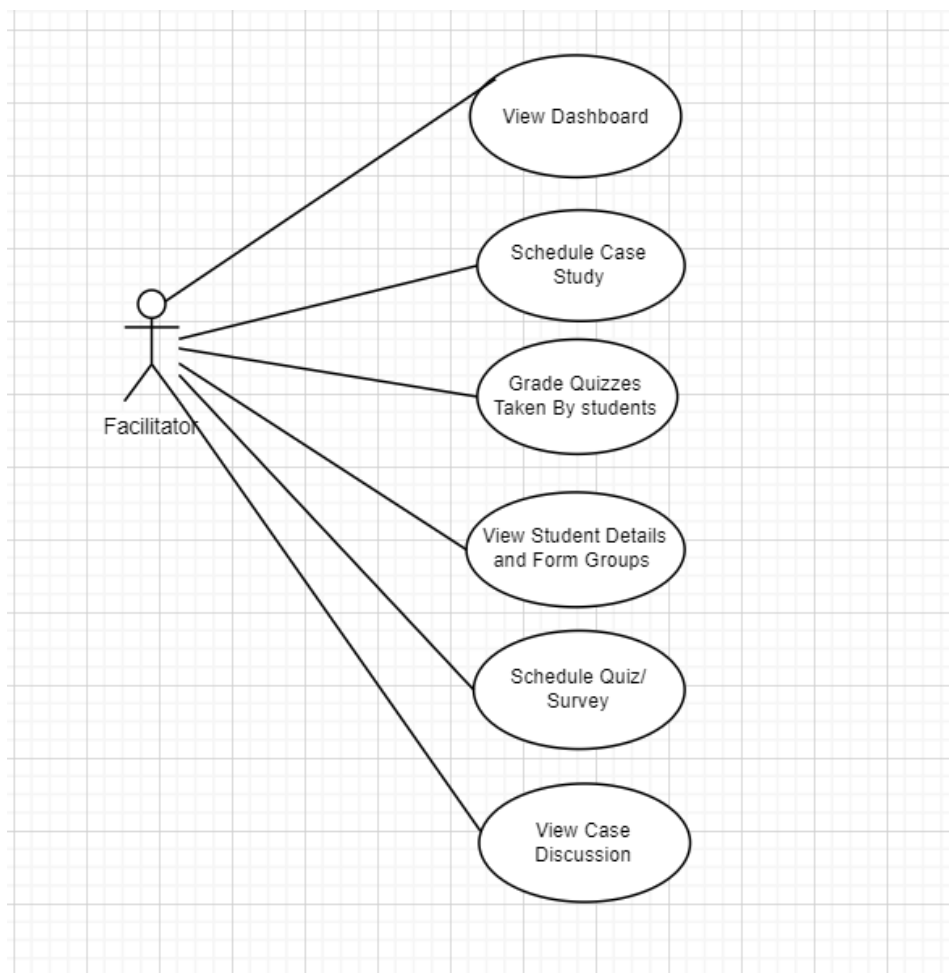
**Author:**

- Author will have privilege to login to the system and view the dashboard. He can create case and upload / delete / modify the multimedia uploaded to the system. He also has the access to create quiz and has the privilege to upload / edit / modify the quizzes for the case.



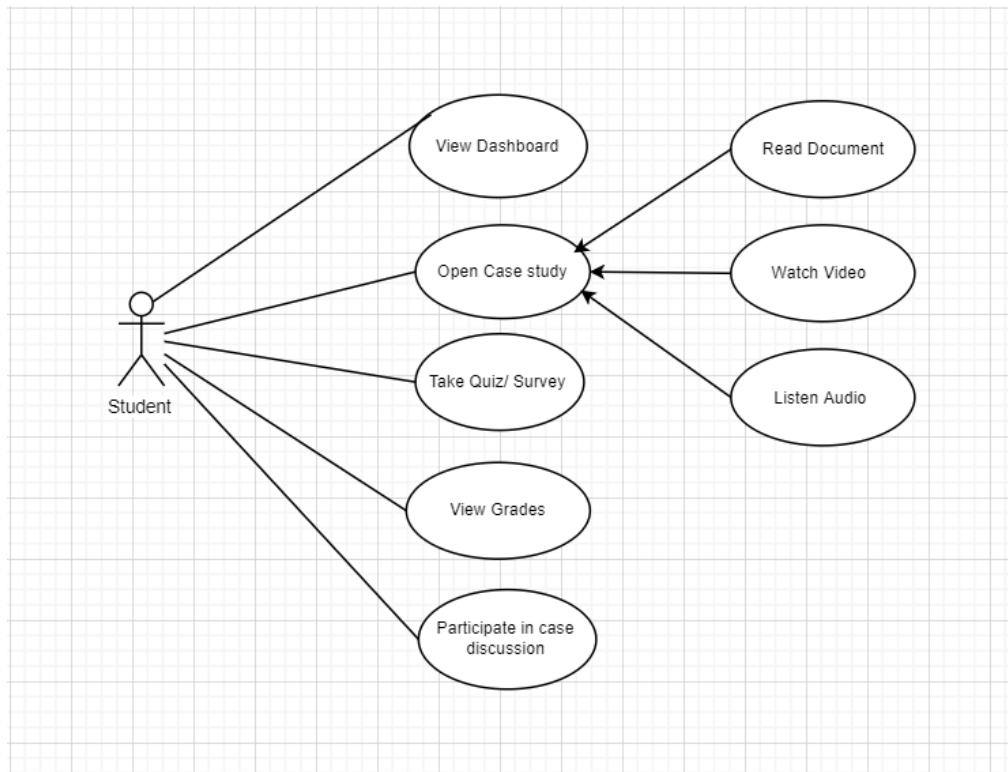
**Facilitator:**

- Facilitator has the credentials to login and access the dashboard of the system.
- Facilitator can make the quizzes and cases available for the students for the time period he wants to.
- Facilitator can embed the logic flow for the cases and quizzes.
- Facilitator will be able to grade the submitted quizzes for each student.
- Facilitator can also view all the student details, their progress and form groups.
- Facilitator will have access to the case discussions wherein he can view all the posted discussions add questions and answer them.



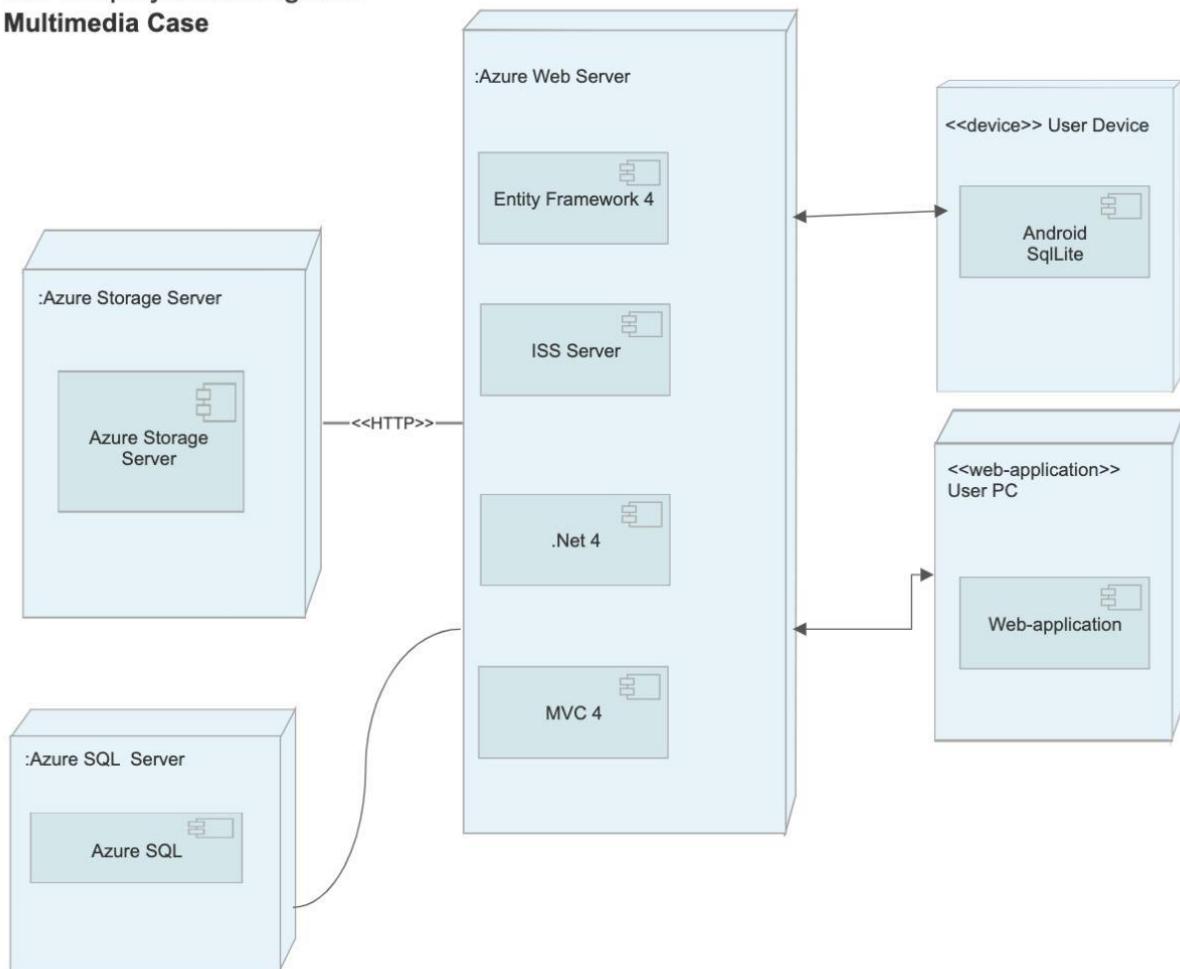
**Student:**

- Student will have the privilege to register for course in Learning management system.
- Student can login to multimedia system and view dashboard.
- Student has the access to multimedia content, and he can also attempt quizzes/surveys in an order facilitated by the instructor.
- Student can save his progress



### 6.3 Deployment diagrams:

#### UML Deployment Diagram: Multimedia Case



### 6.3 Class Diagrams:

#### User:

User class consists of properties Firstname, lastname, user\_id, password, role\_id, quiz responses, surver\_responses, courses, user \_history are of list attributes.

**CreateUser()** method allows to create a new user profile by passing the parameters like firstname, lastname etc., it stores the data of different roles to user table in database.

**ViewUser()** method allows the admin to see all the user roles that have been created.

User
User_ID: int firstname: string lastname: string password: string Role_ID: int quiz_responses: list<quiz_responses> survey_responses: list<survey_responses> courses: list<courses> user_history: list<user_history>
CreateUser(firstname: string,lastname: string,password: string,Role_ID: int,quiz_responses: list<quiz_responses>,survey_responses:list<survey_responses>,courses: list<courses>,user_history: list<user_history>)  ViewUser(USer_ID: int)

## Case:

Case class is one of the important classes in the multi-media case studies application. It has other properties like:

- CaseId for providing the id for a case, it has other properties like Case Name, Case Content, CaseBegins, CaseEnd, Case\_text and survey\_id.
- the dictionary object stores the quiz, discussion, case \_steps and binary\_data for the case.

**CreateCase()** method allows Author to create a new case by passing properties like Case Name, Case Content, CaseBegins, CaseEnd, Case\_text and survey\_id.

**UpdateCase ()** method allows Author to update an existing case by passing properties like Case Name, Case Content, CaseBegins, CaseEnd, Case\_text and survey\_id of the case if they wish to modify the contents of the case or change the priority of the content.

**DeleteCase ()** methods deletes the case that can be viewed by the Facilitator and the Students.

Cases
<pre>case_id: int Casename: string casebegins: datetime caseends: datetime survey_id: int quiz: list&lt;quiz&gt; discussion: list&lt;discussion&gt; Case_Content: list&lt;Case_Content&gt; case_steps: list&lt;case_steps&gt;</pre>
<pre>createcase(Casename: string,casebegins: datetime,caseends: datetime,survey_id: int,quiz: list&lt;quiz&gt;,discussion: list&lt;discussion&gt;,Case_Content: list&lt;Case_Content&gt;,case_steps: list&lt;case_steps&gt;)  updatecase(Casename: string,casebegins: datetime,caseends: datetime,survey_id: int,quiz: list&lt;quiz&gt;,discussion: list&lt;discussion&gt;,Case_Content: list&lt;Case_Content&gt;,case_steps: list&lt;case_steps&gt;)  deletecase(case_id: int)</pre>

## Quiz:

This class all the properties with respect to the quiz that is created by the author for further use by Facilitator and Students. Quiz class has properties like QuizId, QuizName, quiz\_begins, quiz\_ends for the quiz, questions, case\_id are present in the quiz and grades for the quiz are a list of string.

**CreateQuiz()** method helps author to create the quiz for the students by providing the quiz id, QuizName, quiz\_begins, quiz\_ends as input parameters.

**UploadQuiz()** method helps author to uploads the quiz for the facilitator as well as students by providing quiz id, QuizName, quiz\_begins, quiz\_ends as input parameters.

**DeleteQuiz()** method deletes the content of the quiz by taking quiz\_id as input parameters.

Quiz
<pre>quiz_id: int quiz_name: string quiz_begins:datetime quiz_ends:datetime case_id: int quiz_questions: list&lt;quiz_questions&gt; grades: list&lt;grades&gt;</pre>
<pre>CreateQuiz(quiz_name: string,quiz_begins:datetime,quiz_ends:datetime,case_id: int,quiz_questions: list&lt;quiz_questions&gt;,grades: list&lt;grades&gt;)  UploadQuiz(quiz_name: string,quiz_begins:datetime,quiz_ends:datetime,case_id: int,quiz_questions: list&lt;quiz_questions&gt;,grades: list&lt;grades&gt;)  DeleteQuiz(quiz_id: int)</pre>



**Quiz Questions:**

This class has all the properties with respect to the questions that is created by the author for further use by Facilitator and Students. Quiz\_questions class has properties like Quiz\_questions\_Id, quiz-question, quiz\_options, quiz\_id are present in the Quiz\_questions and quiz\_responses for the Quiz\_questions is a list of string.

**AddQuiz()** method helps author to add the quiz for the students by providing the Quiz\_questions\_Id, quiz-question, quiz\_options, quiz\_id as input parameters.

**UpdateQuiz()** method helps author to update the quiz for the facilitator as well as students by providing Quiz\_questions\_Id, quiz-question, quiz\_options, quiz\_id as input parameters.

**DeleteQuiz()** method deletes the content of the quiz by taking quiz\_questions\_id as input parameters.

quiz_questions
<pre>quiz_questions_id: int quiz_question: string quiz_options: string quiz_id: int quiz_responses: list&lt;quiz_response&gt;</pre>
<pre>addquiz(quiz_question: string,quiz_options: string,quiz_id: int,quiz_responses: list&lt;quiz_response&gt;)  updatequiz(quiz_question: string,quiz_options: string,quiz_id: intquiz_responses: list&lt;quiz_response&gt;)  deletequiz(quiz_questions_id: int)</pre>

**Quiz Response:**

This class has all the properties with respect to the quiz responses that are given by a student for a case study. This class is created by the author for further use by Facilitator and Students. Quiz\_responses class has properties like quiz\_response\_Id, quiz-response, Quiz\_questions, user\_id is present in the Quiz\_responses.

**CreateQuizResponse()** method helps students to respond to a questions which includes parameters like quiz\_response\_Id, quiz-response, Quiz\_questions, user\_id.

**ModifyQuizResponse ()** method helps students to modify the response to a question which includes parameters like quiz\_response\_Id, quiz-response, Quiz\_questions, user\_id.

**DeleteQuizResponse ()** method helps students to delete a response to a question which includes parameters like quiz\_response\_Id.

quiz_response
quiz_response_id: int quiz_response: string quiz_questions_id: int user_id: int quiz_answer: string
createquizresponse(quiz_response: string,quiz_questions_id: int,user_id: int,quiz_answer: string)  modifyquizresponse(quiz_response: string,quiz_questions_id: int,user_id: int,quiz_answer: string)  deletequizresponse(quiz_response_id: int)

**Survey:**

This class has all the properties with respect to the survey that is created by the author for further use by Facilitator and Students. Survey class has properties like Survey\_id, SurveyName, Survey timings, Survey\_questions are present in the Survey and survey\_questions are a list of string.

**CreateSurvey()** method helps author to create the Survey for the students by providing the Survey\_id, SurveyName, Survey timings, Survey\_questions as input parameters.

**DeleteSurvey()** method deletes the content of the quiz by taking survey\_id as input parameters.

Survey
survey_id: int survey_name: string survey_timings:datetime survey_questions: list<survey_questions>
Createsurvey(survey_name: string,survey_timings:datetime,survey_questions:,list<survey_questions>)  Deletesurvey(survey_id: int)

**Survey Questions:**

This class has all the properties with respect to the Survey related questions that is created by the author for further use by Facilitator and Students. Survey questions class has properties like survey\_questions\_Id, survey-question, survey\_id and surver\_responses are present in the survey\_questions and survey\_responses for the survey\_questions is a list of string.

**AddSurvey()** method helps author to add the survey for the students by providing the survey\_questions\_Id, survey-question, survey\_id and surver\_responses as input parameters.

**UpdateSurvey()** method helps author to update the quiz for the facilitator as well as students by providing survey\_questions\_Id, survey-question, survey\_id and surver\_responses as input parameters.

**DeleteSurvey()** method deletes the content of the quiz by taking survey\_questions\_Id as input parameters.

survey_questions
survey_questions_id: int survey_question: string survey_id: int survey_responses: list<survey_response>
addsurvey(survey_question: string,survey_id: int,survey_responses: list<survey_response>)  updatesurvey(survey_question: string,survey_id: int,survey_responses: list<survey_response>)  deletesurvey(survey_questions_id: int)

**Survey Response:**

This class has all the properties with respect to the survey responses that are given by a student for a particular case study. This class is created by the author for further use by Facilitator and Students. survey\_responses class has properties like survey\_response\_id, survey-response, survey\_questions, user\_id is present in the survey\_responses.

**CreateSurveyResponse()** method helps students to respond to a survey which includes parameters like survey\_response\_id, survey-response, survey\_questions, user\_id.

**ModifySurveyResponse ()** method helps students to modify the response to a survey which includes parameters like q survey\_response\_id, survey-response, survey\_questions, user\_id

**DeleteSurveyResponse ()** method helps students to delete a response to a survey which includes parameters like survey\_response\_id.

survey_response
survey_response_id: int survey_response: string survey_questions_id: int user_id: int
createsurveyresponse(survey_response: string,survey_questions_id: int,user_id: int)  modifysurveyresponse(survey_response: string,survey_questions_id: int,user_id: int)  deletesurveyresponse(survey_response_id: int)

**binary\_data**

binary\_data consist properties like resource \_id, resource, case\_id, category is of list attributes.

**SaveData()** is the method which allows admin to save data.

**RenderData ()** is the method which allows admin to render the data.

**RemoveData()** is the method which allows admin to remove the data.

Case_Content
resource_id: int resource: blob case_id: int category: string
SaveData(resource: blob,case_id: int,category: string)  RenderData(resource: blob,case_id: int,category: string)  RemoveDate(resource_id: int)

**Discussion:**

discussion consist properties like discussion \_id, discussion \_content, user\_id,case\_id,user is of list attributes.

**CreateDiscussion()** is the method which allows facilitator to Create Discussion.

**DeleteDiscussion()** is the method which allows facilitator to Delete discussion.

discussion
discussion_id: int discussion_content: string user_id: int case_id: int user: list<user> gr_id: int
CreateDiscussion(discussion_content:string,user_id: int,case_id: int,user: list<user>,gr_id: int)  DeleteDiscussion(discussion_id: int)

### Courses:

courses consist properties like course\_id,courses\_name,case\_id,user\_id, user,case is of list attributes.

Create allows facilitator to create a new course.

**Updatecourse()** allows facilitator to update the course.

**Deletecourse()** allows facilitator to delete the course.

courses
course_id: int courses_name: string case_id: int user_id: int case: list<case> user: list<user>
createcourse(courses_name: string,case_id: int,user_id: int,case: list<case>,user: list<user>)  updatecourse(courses_name: string,case_id: int,user_id: int,case: list<case>,user: list<user>)  deletecourse(course_id: int)

### Grades:

Grades consist properties like grades\_id,grade\_obtained,case\_id,quiz\_id,user\_id, user,case is of list attributes.

**viewgrades()** method allows student to view their grades.

**Creategrades()** method allows facilitator to grade the case.

grades
grades_id: int grade_obtained: string case_id: int quiz_id: int user_id: int
viewgrades(grade_obtained: string,case_id: int,quiz_id: int,user_id: int)  creategrades(grades_id: int)

### Roles:

Roles consist properties like Role\_id,Role\_name,Role\_description,user is of list attributes.

**CreateRoles()** method allows Admin to create the Roles.

**Createprivileges()** method allows Admin to grant privileges.

**DeleteRoles()** method allows Admin to delete the Roles.

Roles
Role_id: int Role_name: string Role_description: string user: list<user>
CreateRoles(Role_name: string,Role_description: string,user: list<user>)  Createprivileges(Role_name: string,Role_description: string,user: list<user>)  DeleteRoles(Role_id: int)

### Group:

group properties like gr\_id,group\_name,group\_assigned\_id,user\_id,user is of list attributes.

**Creategroups()** method allows facilitator to create the group from the pool of students.

**Editgroups()** method allows facilitator to edit the group of students.

**Deletegroups()** method allows facilitator to delete the group of students.



groups
gr_id: int group_name: string group_assigned_id: int user: list<user>
creategroups(group_name: string,group_assigned_id: int,user_id: int,user: list<user>)  editgroups(group_name: string,group_assigned_id: int,user_id: int,user: list<user>)  deletegroups(gr_id: int)

### Case\_Steps:

Case\_Steps consists of properties like case\_step\_id, step\_no, case\_id, resource\_id, category\_resource is of list attributes.

**createcasesteps()** method allows facilitator to create step for cases

**modifycasesteps()** method allow facilitator to modify the case

**deletecasesteps()** method allow facilitator to delete the cases

Case_Steps
case_step_id: int resource_id: int step_no: int case_id: int category_resource: string
createcasesteps(resource_id: int,step_no: int,case_id: int,category_resource: string)  modifycasesteps(resource_id:int,step_no: int,case_id: int,category_resource: string)  deletecasesteps(case_step_id: int)

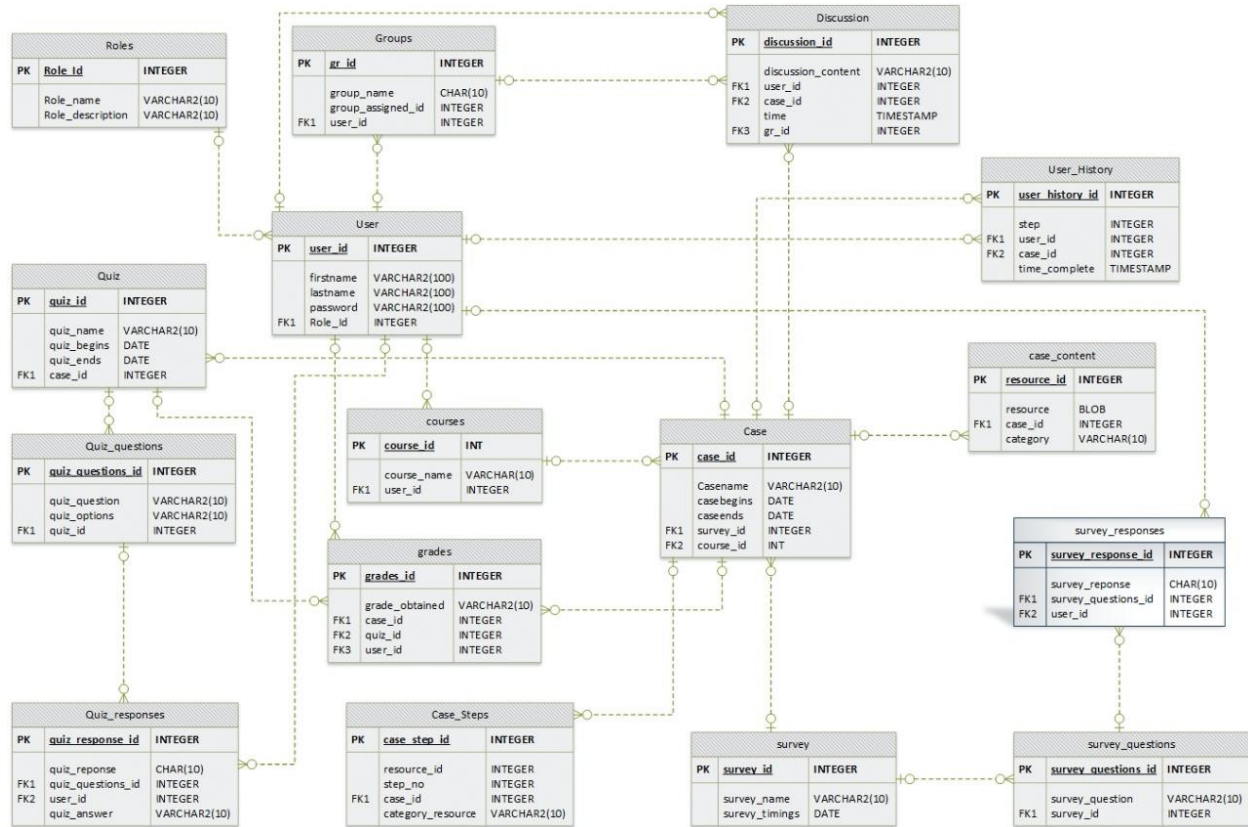
**user\_history:**

user\_history consists of properties like user\_history\_id, step, user\_id, case\_id is of list attributes.

**deleteuserhistory()** method allows the admin to delete all the user history that have been created.

user_history
user_history_id: int step: int user_id: int case_id: int time_complete: timestamp
deleteuserhistory(user_history_id: int)

## 6.4 ER Diagram:



- User has different roles associated with him, which is stored in the roles table, user table has a foreign key to role table indicating the role of the user.
- Quiz has quiz name and questions which belonged to quiz are present in the quiz questions and each response to quiz question is recorded in the quiz responses with the user id of the user who attempted the quiz
- Survey table has the survey name and the questions in the survey is contained in survey questions.
- Discussion table holds all the discussions on the case and holds the user id who participated in the discussion, every comment is displayed based on the timestamp, we haven't followed the mechanism of posts, we treated everything as comments
- User history contains the step where user paused the case previous time, it has the step where he paused previous time.

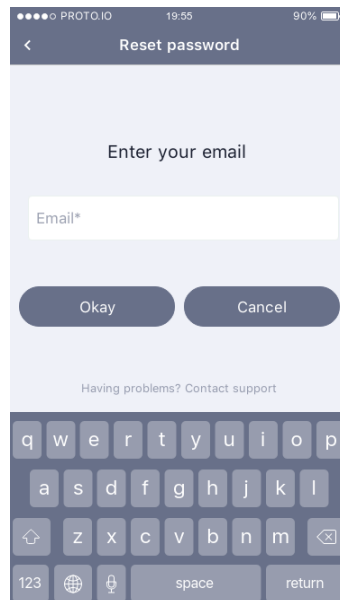
## 7 MOCKUPS

### 1.Login Screen

The mockup shows a mobile application interface for the 'Multimedia Case Study'. At the top, a dark blue header bar contains a back arrow, the title 'Multimedia Case Study', and a status bar above it showing 'PROTO.IO', '19:42', and '90%' battery. The main content area has a light blue background. It features the title 'Sign In to Multimedia Case Study' in bold. Below this are two white input fields with rounded corners, labeled 'Email\*' and 'Password\*'. A dark blue rounded button labeled 'Sign in' is positioned below the fields. Underneath the button is the word 'OR' in a small, light gray font. Below 'OR' is another dark blue rounded button labeled 'forgot password ?'. At the bottom, in a smaller gray font, is the text 'By proceeding you also agree to the Terms of Service and Privacy Policy'.

Screen that appears when the application is opened. The user will be able to enter the credentials and sign in into the application. If the user forgets his/her password ,they can click on forgot password button so that the admin sends a link to reset the password.

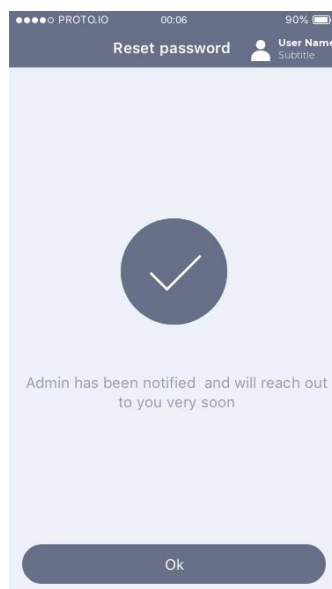
## 2. Password Reset Screen



A mobile app screenshot of the 'Reset password' screen. The status bar at the top shows 'PROTO.IO', '19:55', and '90%' battery. The screen has a light blue background. At the top, there's a dark blue header with a back arrow and the text 'Reset password'. Below the header, the text 'Enter your email' is centered. Underneath is a white text input field with the placeholder 'Email\*'. Below the input field are two dark blue buttons: 'Okay' and 'Cancel'. At the bottom, there's a link that says 'Having problems? Contact support'. A virtual keyboard is visible at the bottom of the screen.

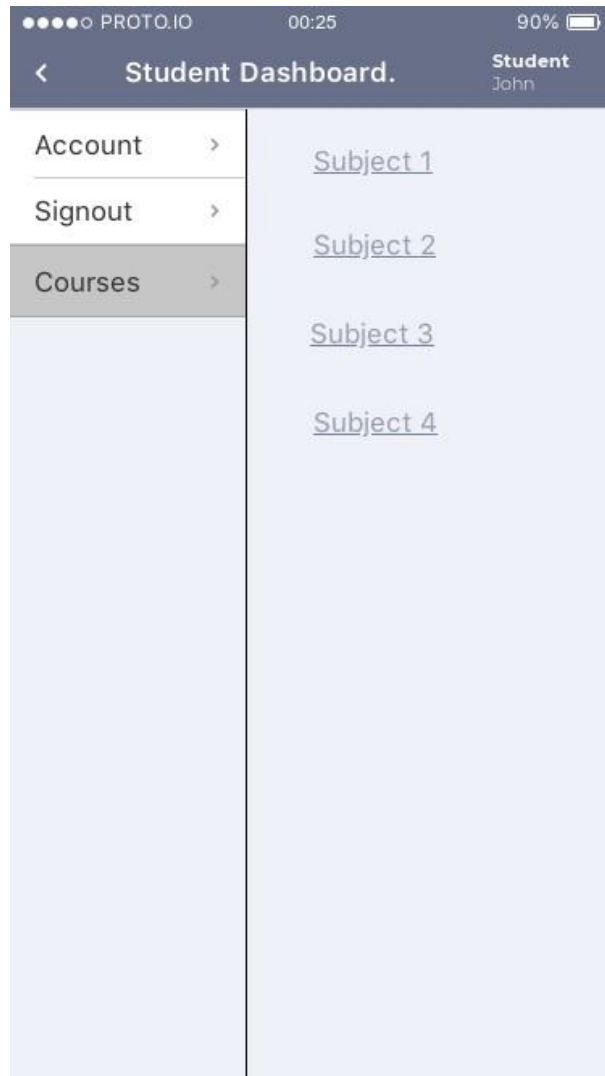
This Screen appears when user clicks on "forgot password" button .This page allows the user to enter the email id to which a link is sent to reset the password.

Screen that appears when the admin has been notified to reset the password . We have not included the automatic reset system as we want to reduce the administration involved with login and registration.



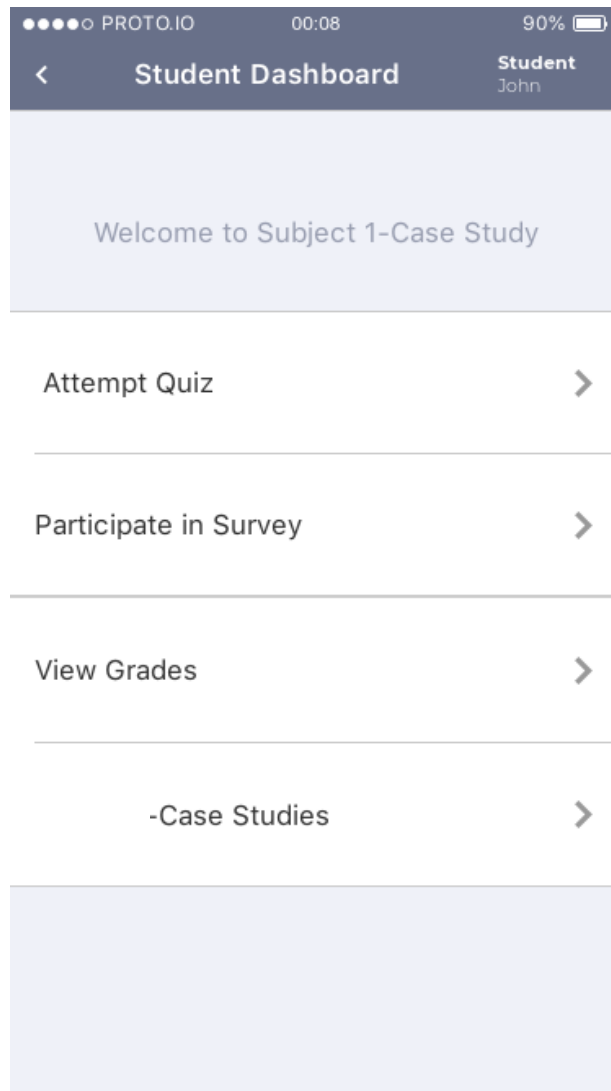
A mobile app screenshot of the 'Reset password' screen showing a confirmation message. The status bar at the top shows 'PROTO.IO', '00:06', and '90%' battery. The screen has a light blue background. At the top, there's a dark blue header with a back arrow, the text 'Reset password', and a user profile icon with the text 'User Name' and 'Sub2010'. In the center, there's a large dark blue circle with a white checkmark. Below the circle, the text 'Admin has been notified, and will reach out to you very soon' is centered. At the bottom, there's a dark blue button with the text 'Ok'.

#### 4.Student Dashboard



This screen is the student dashboard and is displayed to student upon login  
Courses: Displays all the subjects that a Student is taking for the Semester  
Sign out : The button allows the student to log out from the application  
Account : On clicking this button all the student details will be displayed .

## 5. Student Dashboard



This screen appears when the student clicks on any of the subjects in the Courses-Student Dashboard, Student has to do each and every step as prescribed by the facilitator. If the student violates any step, they will be redirected to the right step that needs to be followed.

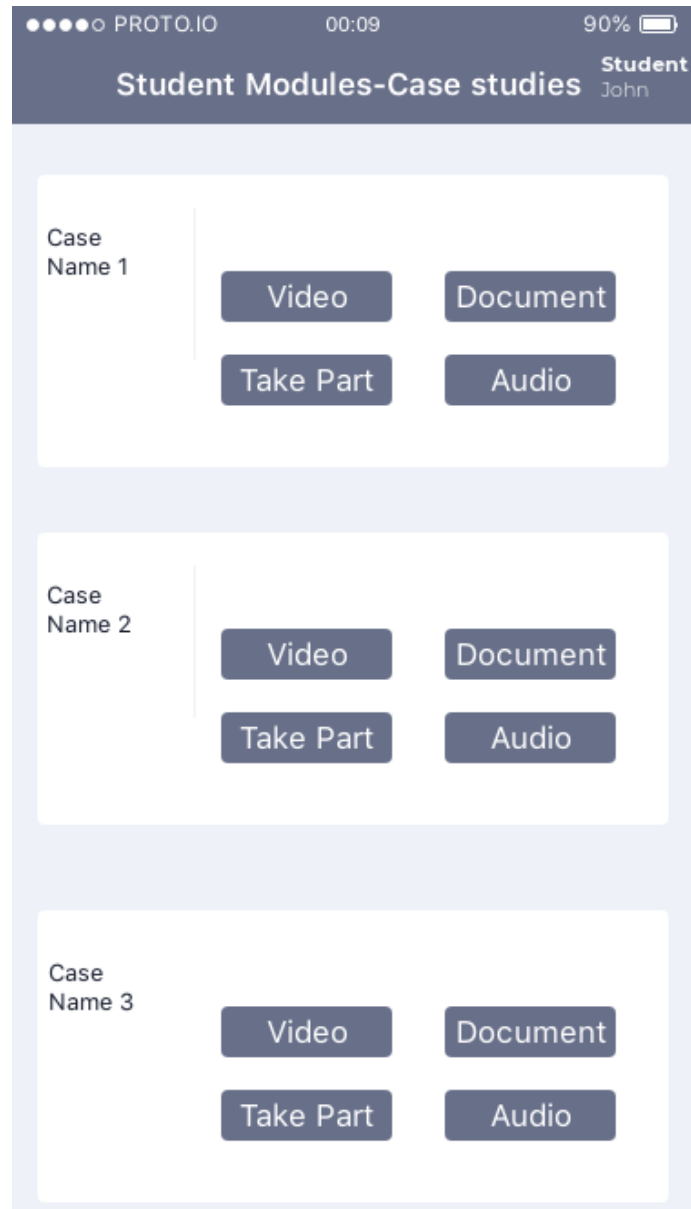
**Attempt Quiz:** Takes the student to a page where he/she can attempt quiz

**Participate in Survey:** Takes the student to a page where he/she can participate in a survey

**View Grades :** Takes the student to a page where he/she can view the grades.

**Modules-Case Studies:** Takes the student to a page where he/she can view go through the Case studies.

## 6. Student Modules- Case Studies Screen

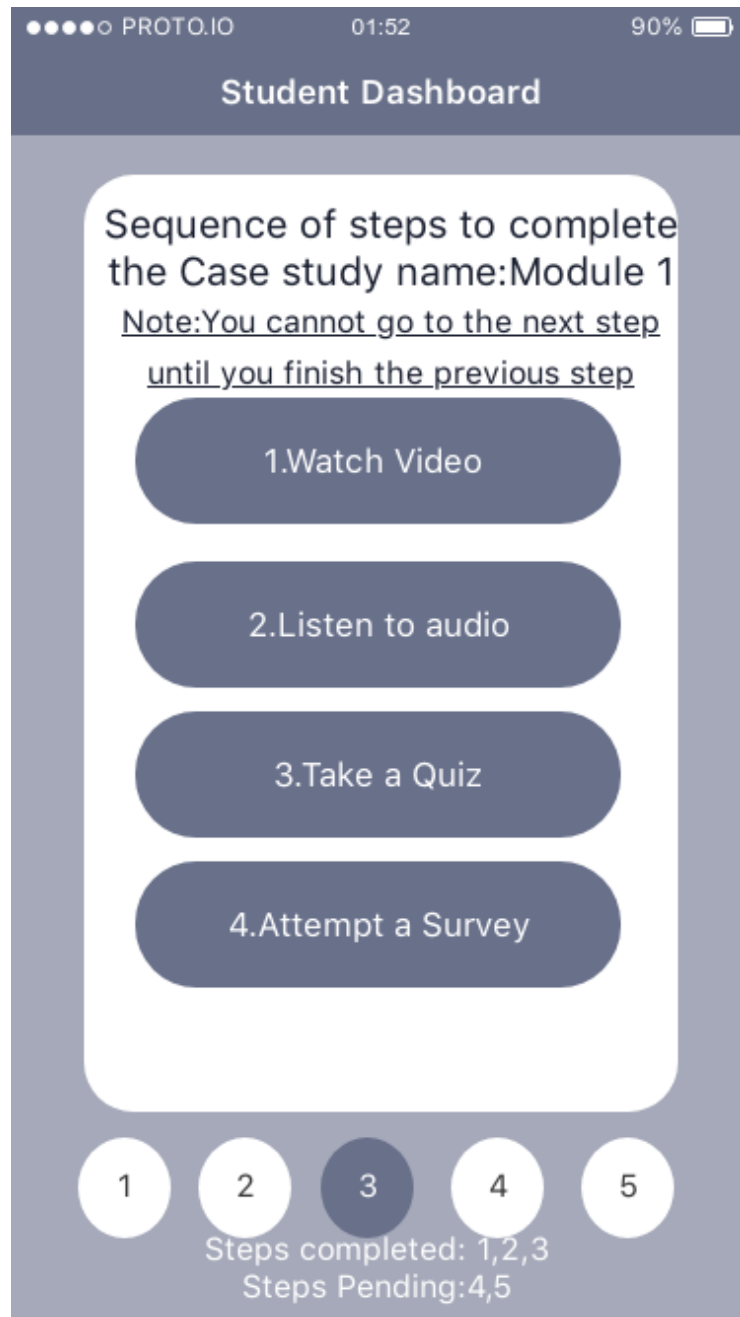


Student will click on the video/document/audio button and the respective multimedia will be displayed for the student for the selected case.

When clicked on Take Part button , the student will be directed to a screen with questions for survey where he can select his answers for the survey.

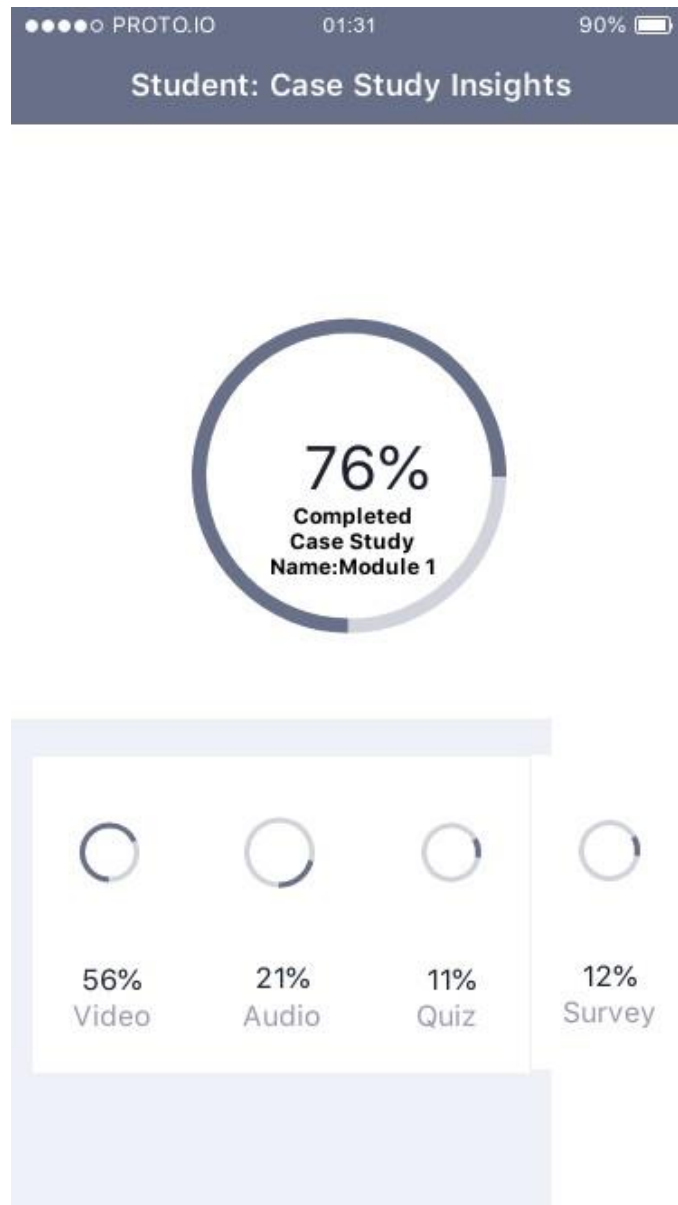


## 7. Student Dashboard



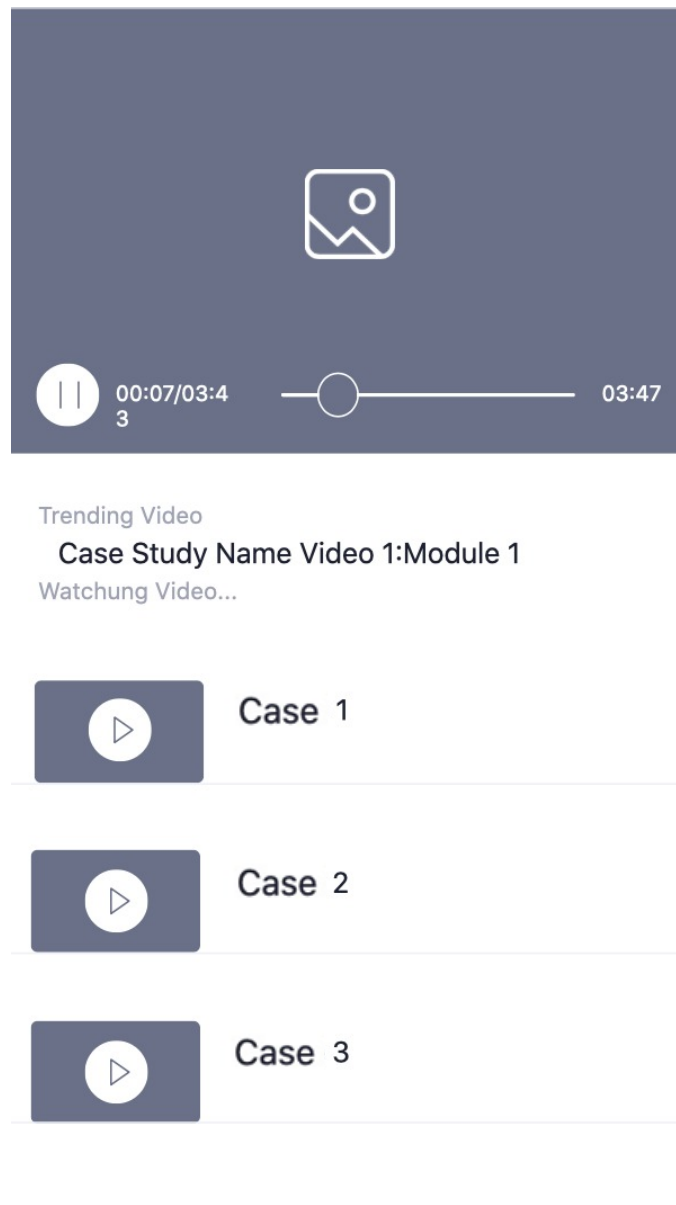
In this screen, student will be able to view the sequence of events the student should follow and also displays the status of each event.

## 8. Student: Case Study Insights screen



In this screen, the student will be able to view his insights on particular case study and all its details.

## 9. Video pop up screen



When the user clicks on the watch a video button, this screen will pop up with the video playing.

## 10. Student Modules View Grades screen



Student will be able to view his/her grades for each quiz they have participated all in a single screen

## 11. Module Case Screen

PROTO.IO 00:09 90%

< Module Case Student John

Post your Queries/Answers

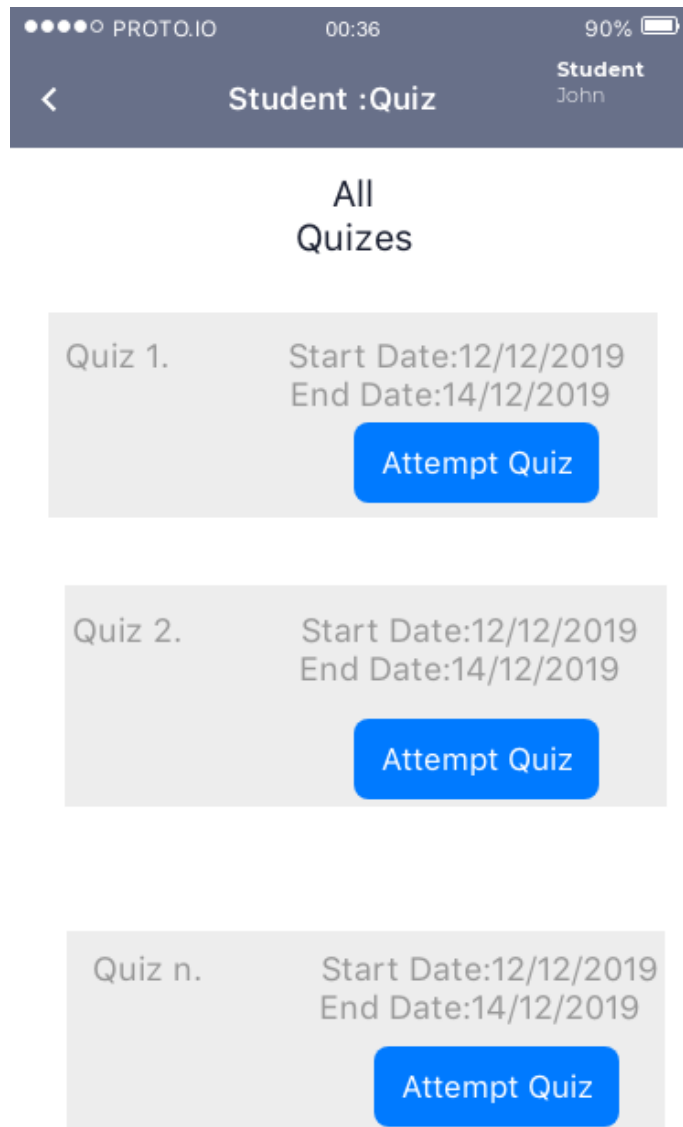
Post your comments here.....

+ Write message Click to Add

q w e r t y u i o p  
a s d f g h j k l  
↑ z x c v b n m ↵  
123 globe mic space return

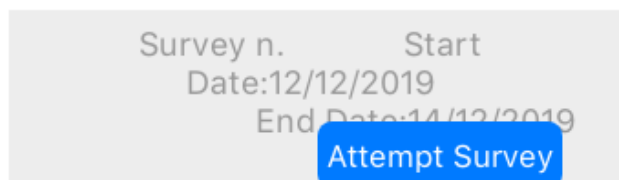
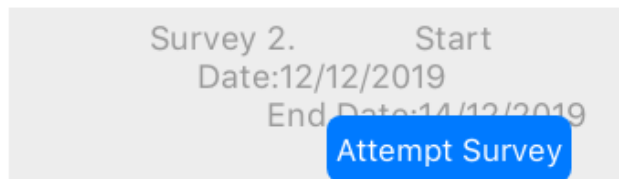
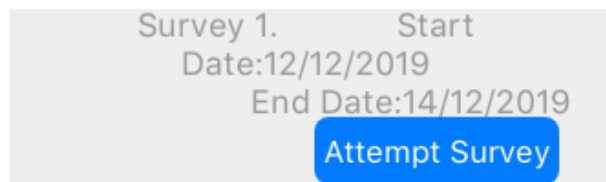
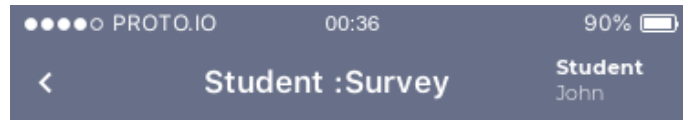
The students will be able to post their questions and answers for the respective case study they are participating.

## 12. Quiz screen for Student



The student will be able to see Quizzes available for them and Start Date and End Date of the quiz so that they will be aware of the deadlines. When clicked on Attempt Quiz screen, the system redirects to a screen with questions and options for the student to take the quiz.

### 13. Survey Screen



The students will be able to view all the surveys available for them with the start and end date of the surveys so that the students will be aware of the deadlines. When clicked on Attempt Survey button, the students will be redirected to screen containing the survey questions.

#### 14. Quiz Screen

●●●●○ PROTO.IO

00:10

90%

Quiz: Case Name

Student  
John

Question 1

What is the answer for the question ?

A

B

C

D

Question 2

What is the answer for the question ?

A

B

C

D

Question 3

What is the answer for the question ?

A

B

C

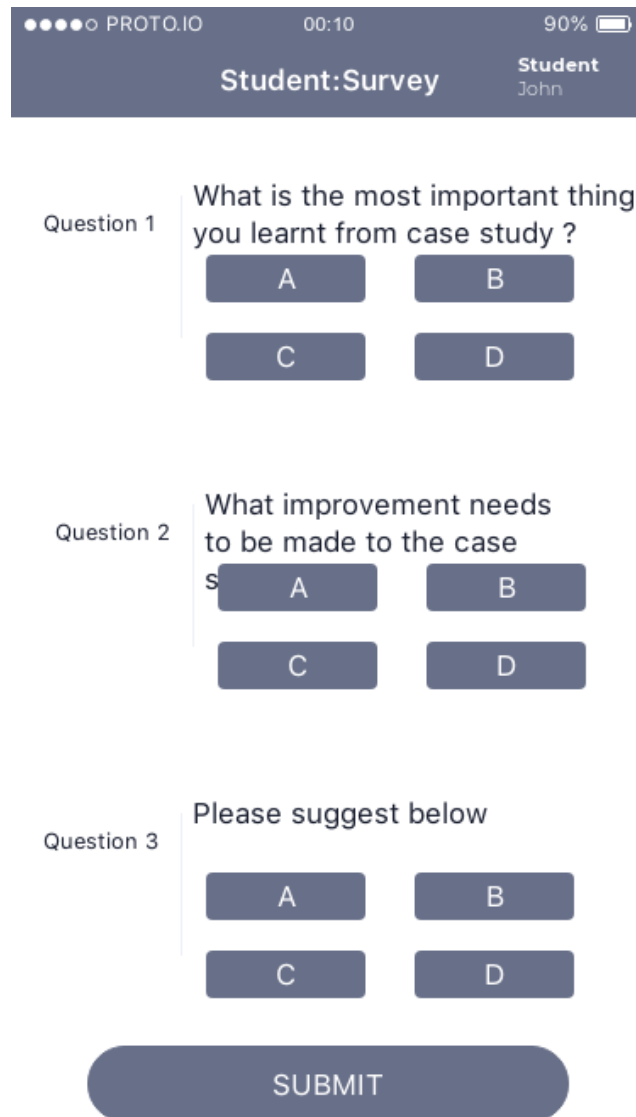
D

SUBMIT

The quiz screen contains the questions and options for the students to choose the options for the questions and when clicked on Submit button, the answers will be submitted.



## 15. Survey Screen



The image shows a mobile app interface for a survey. At the top, there is a status bar with signal strength, the text 'PROTO.IO', a timer '00:10', and a battery level '90%'. Below this is a dark blue header with the text 'Student:Survey' and 'Student John'. The main content area contains three questions, each with four options (A, B, C, D) in dark blue buttons. Question 1 asks 'What is the most important thing you learnt from case study?'. Question 2 asks 'What improvement needs to be made to the case study?'. Question 3 asks 'Please suggest below'. At the bottom is a large dark blue button labeled 'SUBMIT'.

Question 1 What is the most important thing you learnt from case study ?

A B

C D

Question 2 What improvement needs to be made to the case study ?

A B

C D

Question 3 Please suggest below

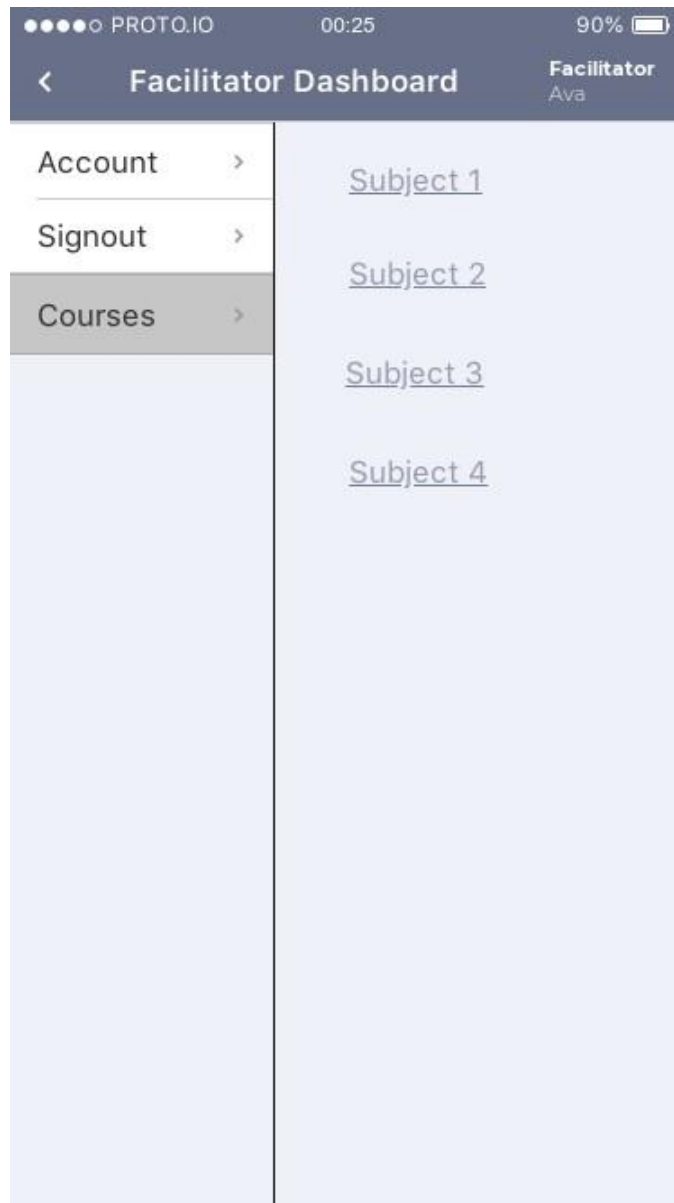
A B

C D

SUBMIT

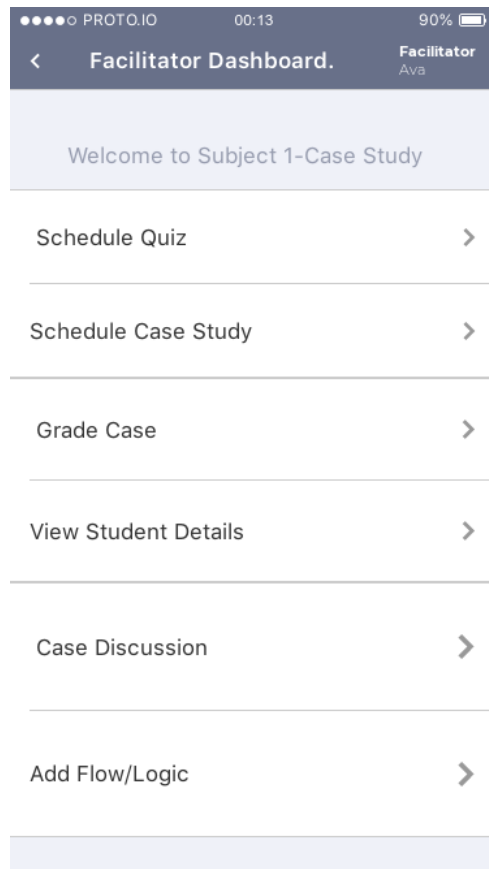
The survey screen contains the questions and options for the students to choose the options for the questions and when clicked on Submit button, the answers will be submitted .

## 16. Facilitator Dashboard screen



This screen is the facilitator dashboard and is displayed to facilitator upon login  
Courses: Displays all the subjects that a Facilitator is teaching for the Semester  
Sign out: The button allows the facilitator to log out from the application  
Account: On clicking this button all the facilitator details will be displayed .

## 17. Facilitator Dashboard



This screen appears when the student has clicked on a particular subject in the courses-Facilitator dashboard.

**Schedule Quiz:** Takes the facilitator to a page where he/she can set a date for the quiz to be taken by the students.

**Schedule Case study:** Takes the facilitator to a page where he/she can schedule a case study.

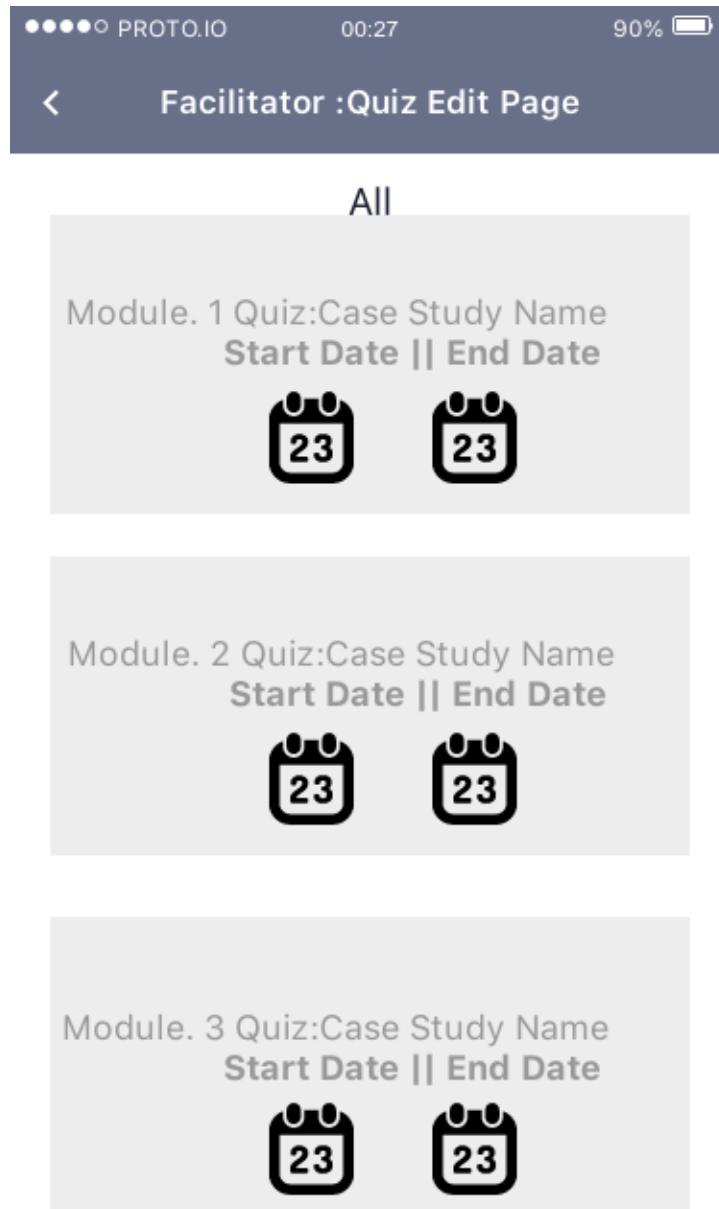
**Grade case:** Takes the facilitator to a page where he/she can grade each quiz for a module.

**View Student Details:** Takes the facilitator to a page where student details are available.

**Case Discussion:** Upon clicking on the case discussion button, the facilitator can view and respond to the student's comments and queries.

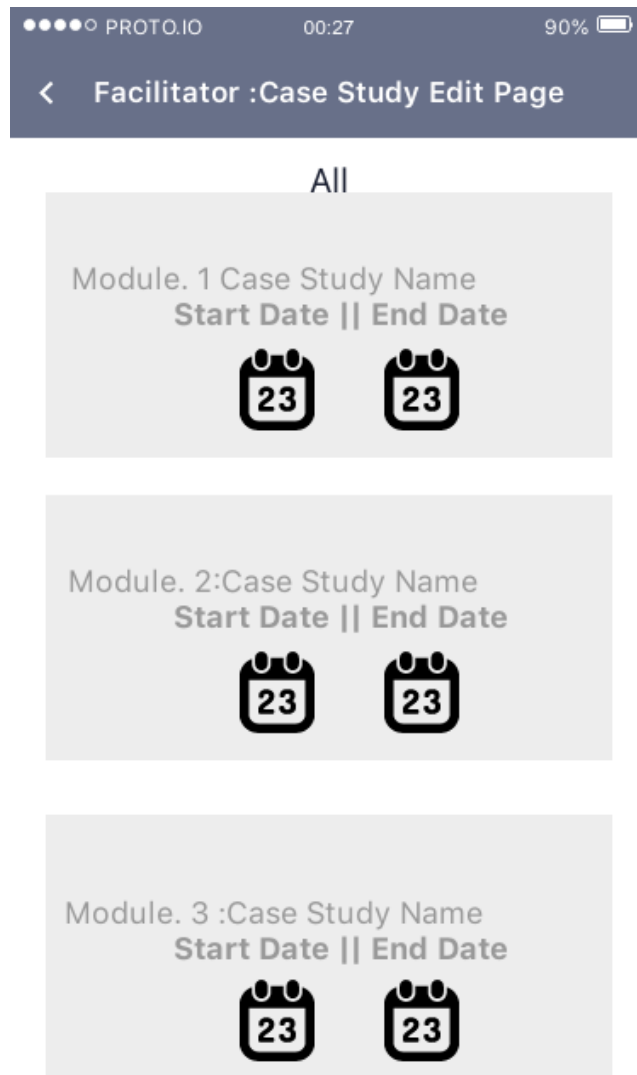
**Add Flow/Logic:** Upon clicking on this, the facilitator can control the flow of quizzes, surveys and case studies.

## 18. Facilitator: Edit Quiz Screen



In this screen, the facilitator will be able to edit the start and end date of the particular quiz

## 19. Facilitator Case Study Edit Page



In this screen, the facilitator will be able to edit the start and end date of the particular case study


## 20. Facilitator Grade Case Screen

<

Facilitator :Grade Case

Facilitator  
Ava


Student 1 Grades



Case 1 :A

Case 2: A


Student 2 Grades



Case 1 :A

Case 2: A

Student 3 Grades

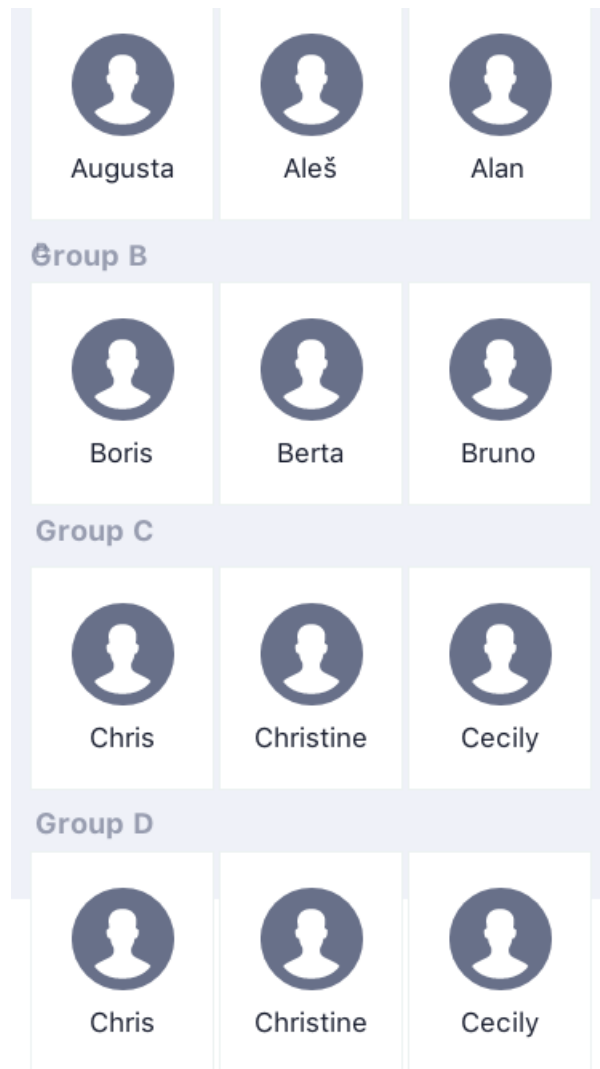


Case 1 :A

Case 2: A

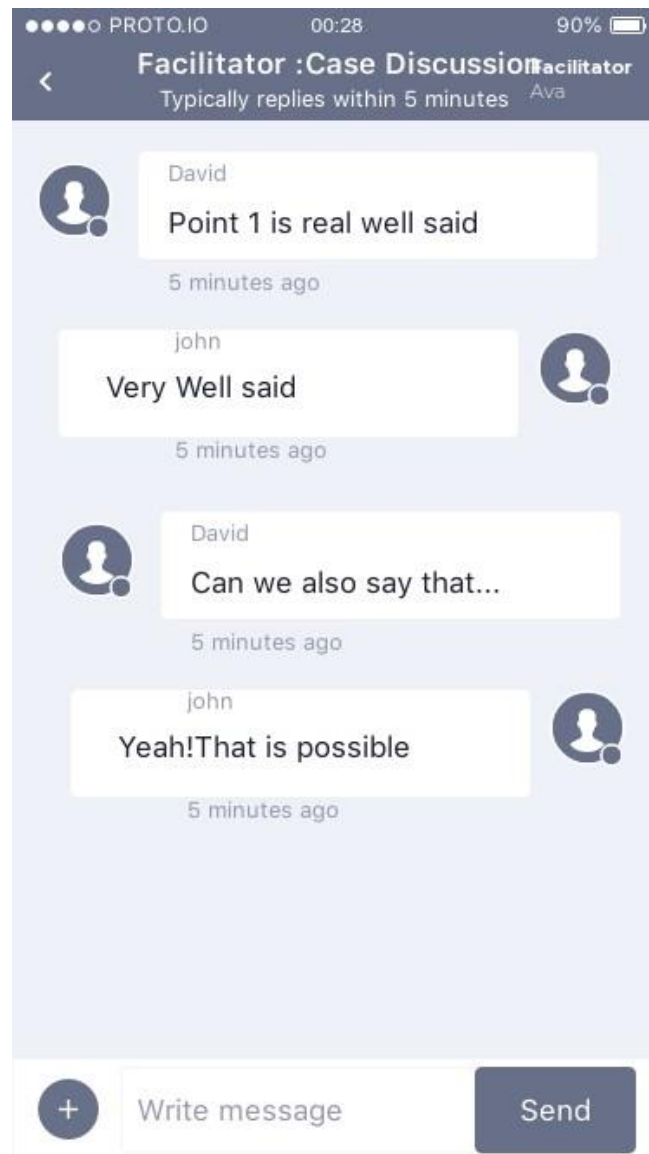
In this screen, the facilitator will be able to view the answers for the quiz's students have submitted and will be able to assign grade.

## 21. Facilitator- Students Group Details Screen



In this screen, the facilitator will be able to form the students into group.

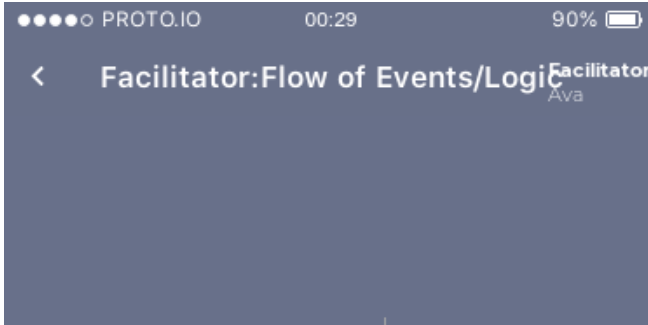




## 22. Facilitator: Case Discussion Screen



The screen will display the discussions happening for that case to the facilitator so that he can view / participate in the discussion. We are assuming this as a comments thread rather than posts thread.

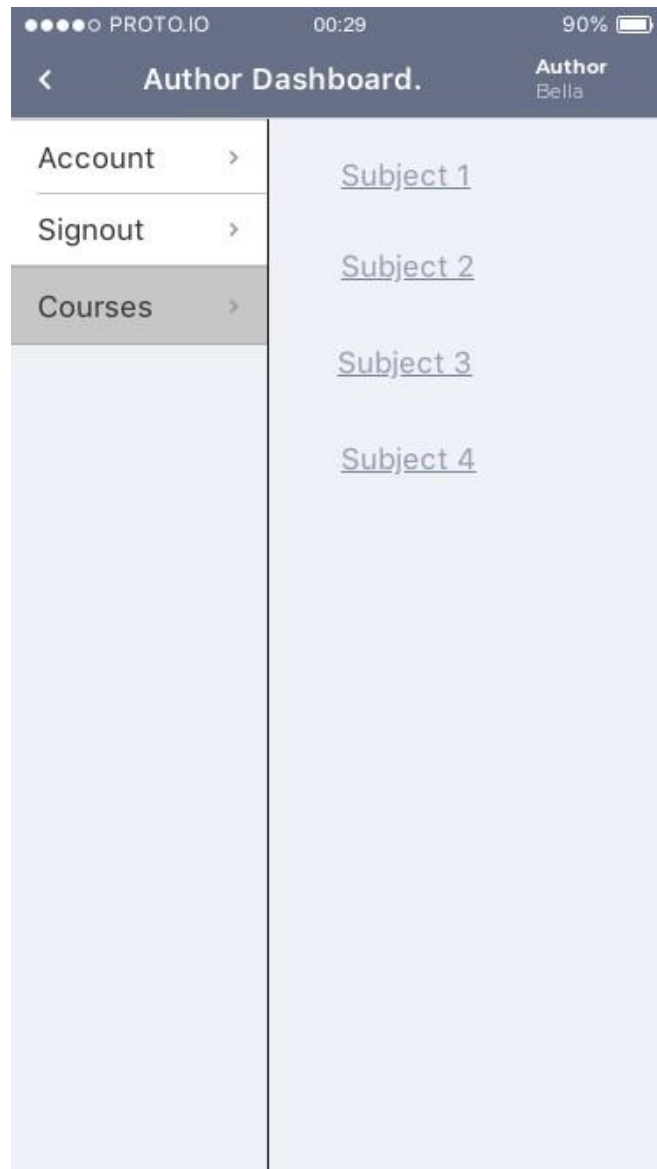


### 23. Facilitator: Flow of Events / Logic Screen

		
	Quiz 1	Order of event:2
	Quiz 2	Order of event:4
	Video	Order of event:1
	Survey	Order of event:3

The facilitator will be able to design the logic of events.

## 24. Author Dashboard



This screen is the author's dashboard and is displayed to author upon login

Courses: Displays all the subjects that an author is writing the cases.

Sign out: The button allows the author to log out from the application

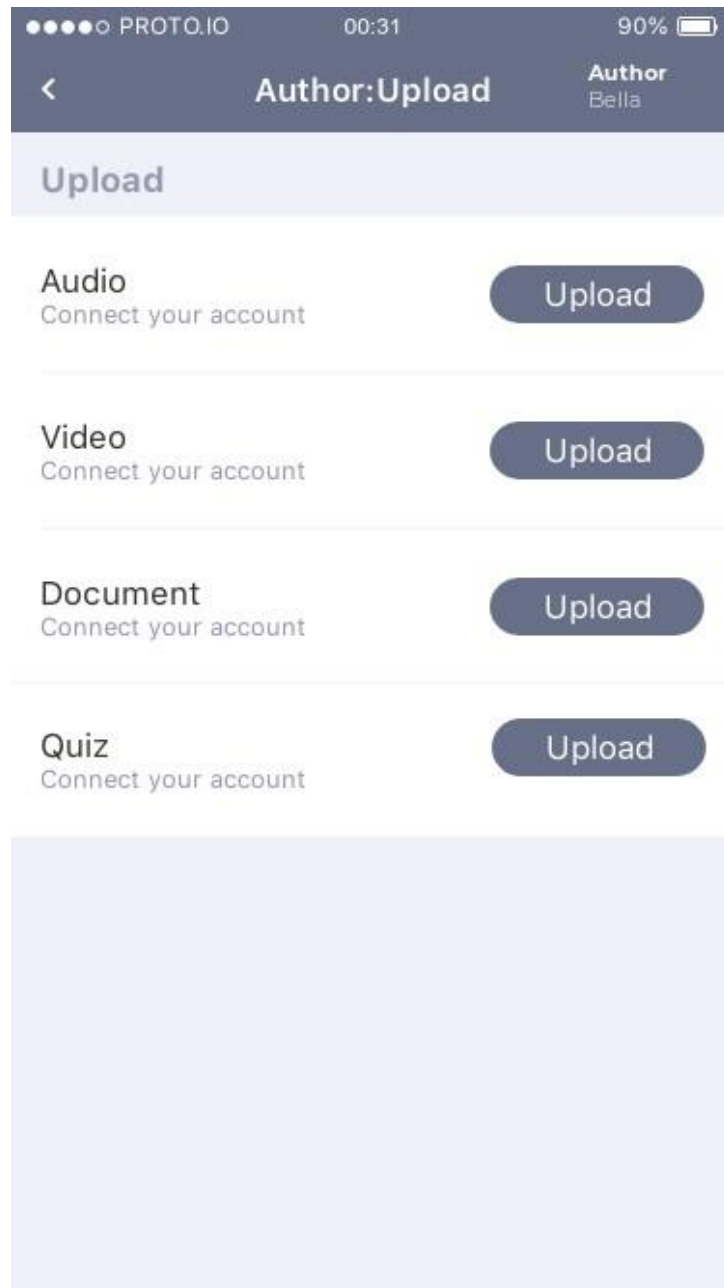
Account: On clicking this button all the author details will be displayed.

## 25. Author Dashboard



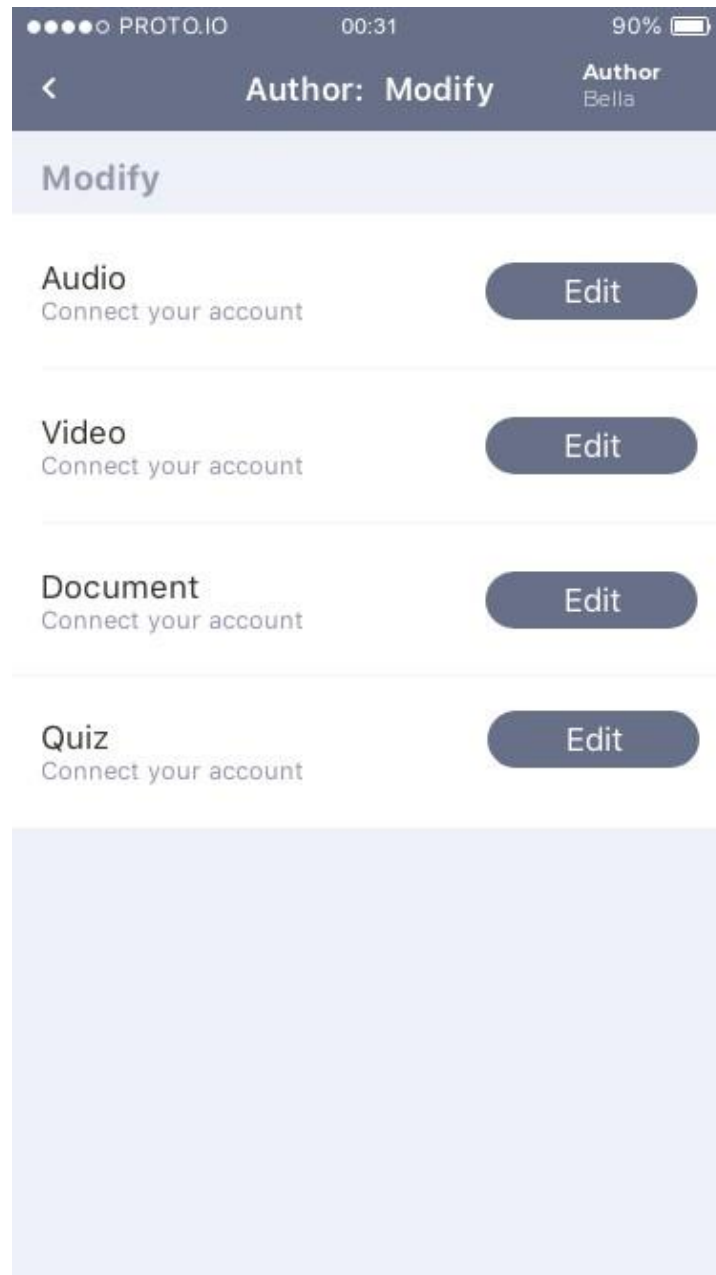
This screen the author will be able to upload, modify or delete the case studies.

## 26. Author Upload Screen



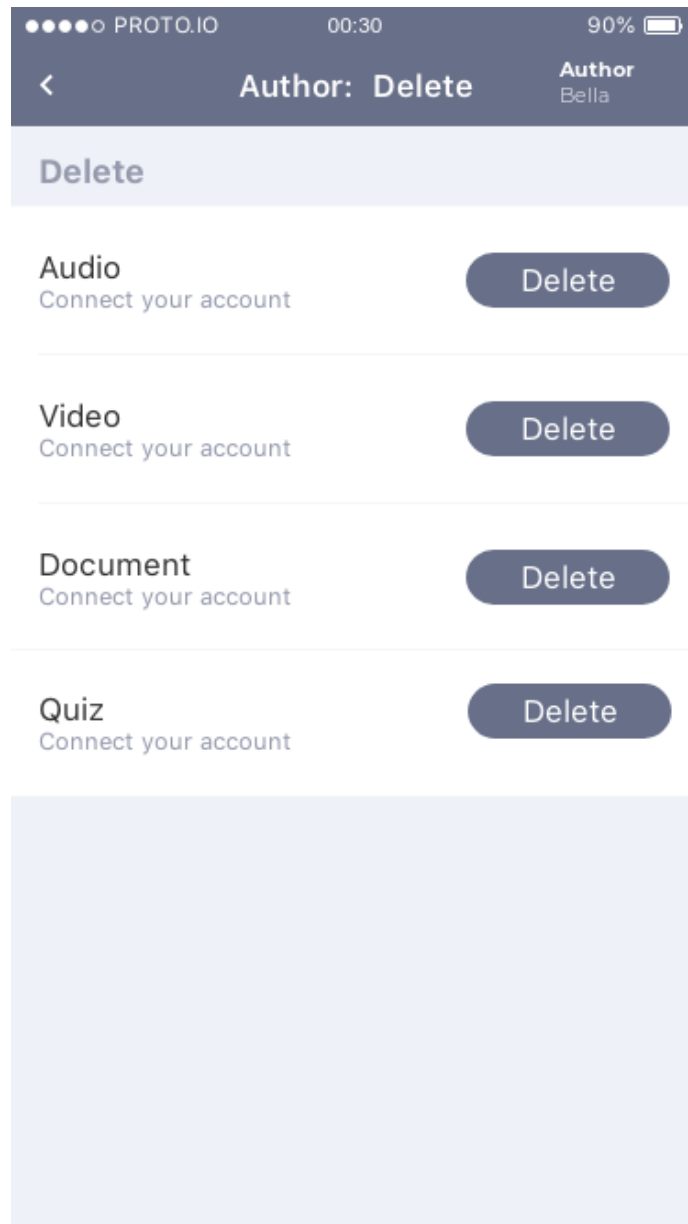
In this screen, the author will be able to upload different types of multimedia such as Audio, Video, Document and Quiz so that they are available for the students.

## 27. Author Modify screen



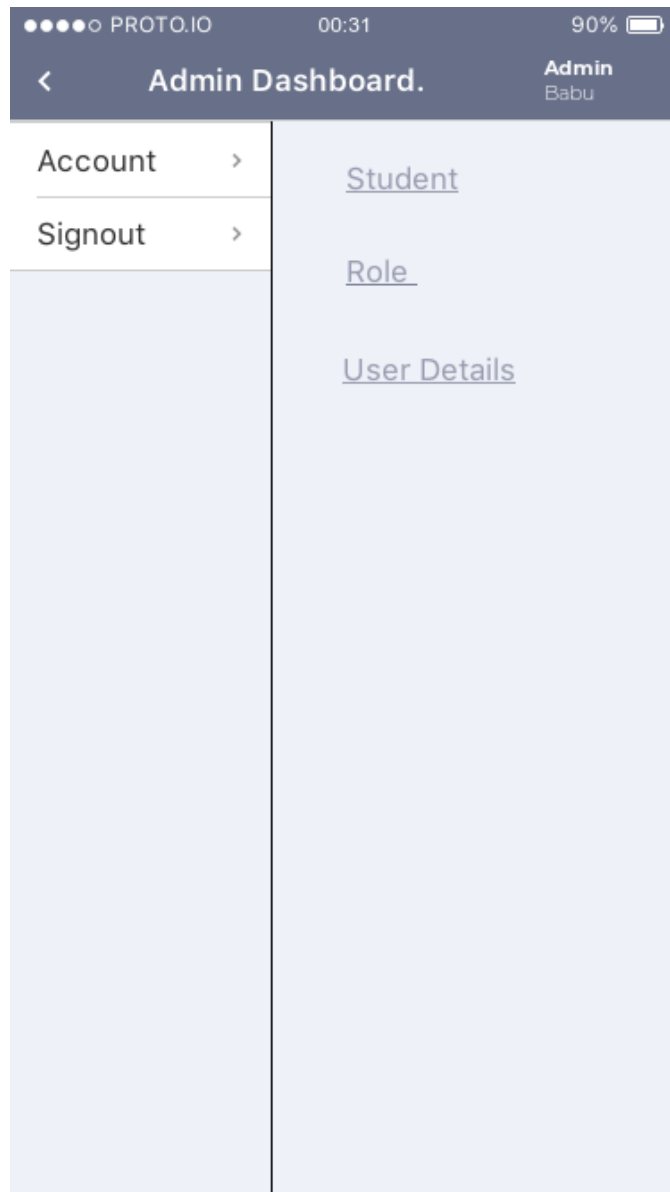
In this screen, the author will be able to edit different types of multimedia such as Audio, Video, Document and Quiz so that the modified multimedia are available for the students.

## 28. Author Delete Screen



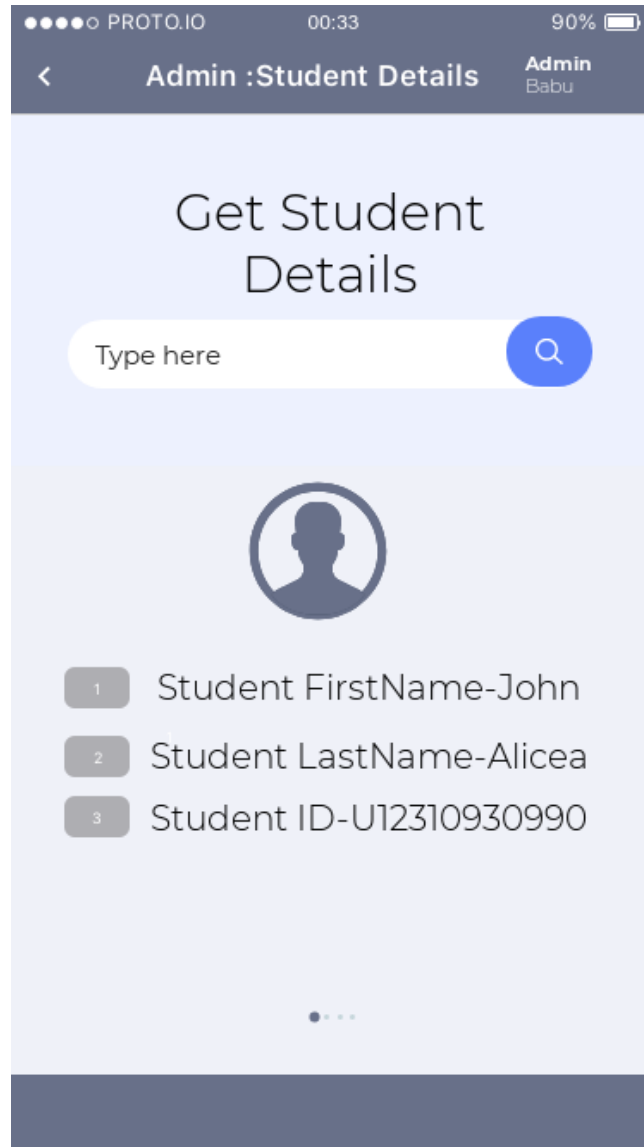
In this screen, the author will be able to delete different types of multimedia such as Audio, Video, Document and Quiz .

## 29. Admin Dashboard



This screen is the admin dashboard and is displayed to admin upon login  
Sign out: The button allows the admin to log out from the application  
Account: On clicking this button all the admin details will be displayed .

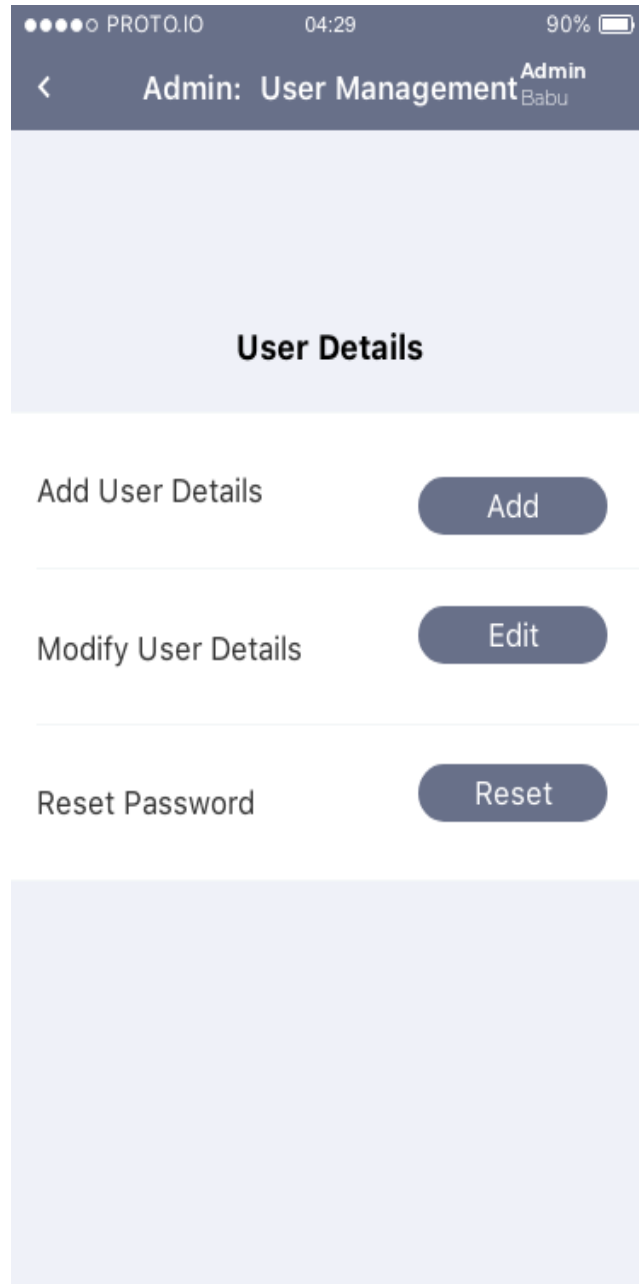
### 30. Admin: Student Details Screen



Through this screen, the admin will be able to view the students who have registered.

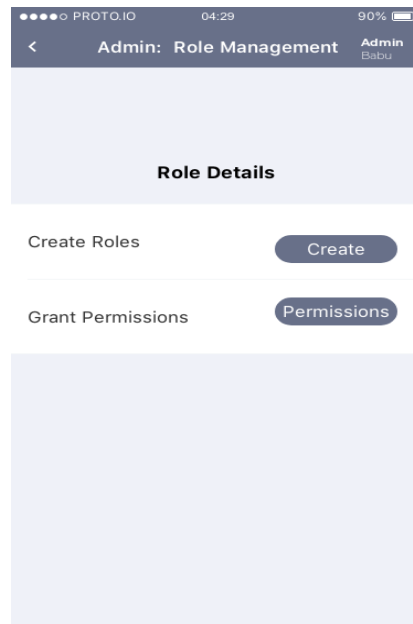


### 31. Admin: User Management



In this screen, the admin will be able to add and modify the user details and will be able to Reset password of the user upon request.

## 32. Admin: Role Management Screen



The admin will be able to create roles for the application and grant permissions to the users according to their roles assigned to them.

## 8 ABOUT MVP

We wanted to test MVP with all the features which we will be including in the future, hence some of our assumptions might be way more than what's expected out of an MVP, some of our crucial assumptions are listed below.

- Both Authors and instructors will have the capacity to upload quizzes and assignments, when this product scale, we assumed flexibility in this aspect makes lot of impact.
- We added flow to each case study, this will be helpful in adding logic in the future, we have also presented the details in future scope
- We wanted system to be organized and envisioned clean user interface hence we included the cases under the course, this will be used by the instructor to grade the student easily for a course, student will be presented with a clean organized interface.

## 9 FUTURE SCOPE

In this part, we will emphasize of the things which we are looking to improve in the future updates

### Discussion:

We have followed the principle of comments thread; future updates will have the principle of posts where each people can write a post and others can react to it.

### Logic implementation

We can add more logic to the current application with the specification of logic, we have designed our own language to implement the logic, since we consider every part in case is a step and step numbers are assigned by the facilitators for students to follow.

We have condition statement in condition column, specification column has the value that we use for comparison.

### Condition elements

==: equals comparison

ge: greater than

le: lesser than

+, -, /: Plus, minus and division

### Specification

Any number

### Actions elements

g: go to mentioned step

r: repeat the present step

id	quiz_id	condition	specification	action
1	2	2ge	45	G4
2	2	2==	50	G5

We have to implement the business logic in way that, after completing each step, variable known as result needs to populate and stored in the database for that respective step, this value will be compared with the specification in the table.

The above data in the row translates to

If the result from the step 2 > 45

Goto step 4

If the result from the step 2 == 50

Goto step 5

id	quiz_id	condition	specification	action
1	3	2+3ge	100	G5

If the addition of result from step 2 and step 3 is greater than 100, proceed to step 5, we can add complex logic with the current constructs, and we can make add intelligence to the case studies.

In initial phases, we can just give the author the interface to add the logic later on new ui elements can be added.

Database schema

Logic_id	Int PK
Quiz_id	int
Condition	String
Specification	int
action	G5

### **Interface:**

The future products will have better interface with conditional logic interface for the authors to write down the logic.