**Analysis**

* Introduction
  + Current Setting.
  + The Old system
  + The proposed system
  + Feasibility Study
* Analysis – Functionalities and Activity
  + Introduction
  + Actors - Operational Explanation
  + Use Case Diagram
  + Sequence Diagrams
    - Scholars
    - Student
    - Admin
* Analysis – Inputs
  + Introduction : Users based on data between them
  + Class Diagrams : Attributes only
  + Class diagram : Entity Relation
  + Data Dictionary : Tables
* Analysis – Security
  + User Roles : Who can access certain Output and Inputs
  + Security Layers
  + Physical Security

1. **Introduction**

After giving an overview of what we aim to build, we seek to analyze the functionalities, input data and security aspects of our system. However, a general picture of the current setting of the system is presented. Then lightly, we touch again on the problem and how we evaluated and sought possible solutions. The points of weakness in the current method of seeking knowledge is explained as well as the strengths that we retained. We then touch on a feasibility study in its realms: Economical, Technical, Operational and Schedule.

The functional analysis is introduces the main functions in the system. Our goal here is to have a building block of activities for the different implementations. We first layout all the activities/functions/use-cases to be undertaken by the different users – now called actors. Based on this, we layout this into a visualized use-case diagram. The use-cases however are not to explain the sequential flow on the system. For this, we use a sequence diagram to explain the order of flow functions. The sequence diagrams are split into three: admin, student and Scholars.

The Inputs’ analysis focuses on the data exchange and flow between the major units of the system. We thus present a class diagram to depict how the major entities are related and their major attributes. The resulting diagram is our stepping stone to creating the entity relation diagram – a UML diagram that reflects a skeleton of our database to be. Finally we sketch tables that show the different data stores: a Data Dictionary.

A security analysis is aimed at setting layer of security for the final implementation. The user roles and different access rights is discussed. We also talk on the layers of security we wish to use from the platforms we plan to use.

1. **Current Setting**

Islamic knowledge is sought from knowledge circles in Mosques, gatherings, lectures during funerals and Memorization centers. “Knowledge is taken from the lips of Scholars” is a widespread principle. Students thus seek knowledge from as far as China and Indonesia to Africa and sometimes stay with a single scholar for more than eight years. The result is a flood of students around the countable scholars.

The students thus need guidance and organization – presented as the main problems. The public, on the other hand need this type of knowledge close in reach to use as tools for guidance. From consultations and engagements the most plausible solution lies in using a mobile application. The application is meant to curb the shortcoming of miscommunications and organization. A student should easily find the timetable of any scholar. In fact, they should see all the scholars and the courses they offer. See courses offered under different fields of Knowledge.

The Islamic Scholarly system though has been part of the Islamic culture since the beginning of Knowledge in this Nation. But different generations have integrated differently with the same knowledge with the change of immediate variables. We clearly state that this project aims to help in its delivery and not interfere with the defaults.

From this start off, we propose a system that will cover the problems – more clearly stated as, an abstraction of how we should integrate with knowledge today in the light of technology.

1. **Feasibility study**

A feasibility analysis evaluates project’s potential for success; therefore, perceived objectivity is an essential factor in the credibility of the project.

1. **Technical Feasibility.**

Our system has all its resources for development met. The software team consists of a Laravel developer, Two Android programmers and a Web developer. Software resources such as our laptops are enough to get the system going. Use of ready-made tools at our disposal cuts development cost to almost zero. However, some money is spent in sever host and Google play platforms for hosting the app for download by specific scholars.

1. **Operational Feasibility**

Our users and most students of knowledge agree to the fact and Idea of centralizing Islamic lectures are one of its kind. The project generally needs minimal supervision and may be assigned to a company to supervise most admin roles .The restrain of scholar’s registration to the admin ensures integrity and reduces operational costs in the future.

1. **Economic Feasibility.**

The diagram shows how much it would cost the project to keep it started and going.

|  |  |  |
| --- | --- | --- |
| No. | Item Needed | Cost |
|  | Laptops | $300 \* 2 |
|  | Phones | $200 \* 1 |
|  | Hosting and Domain | $30 (1 year plan) |
|  | Play store | $ 25 |

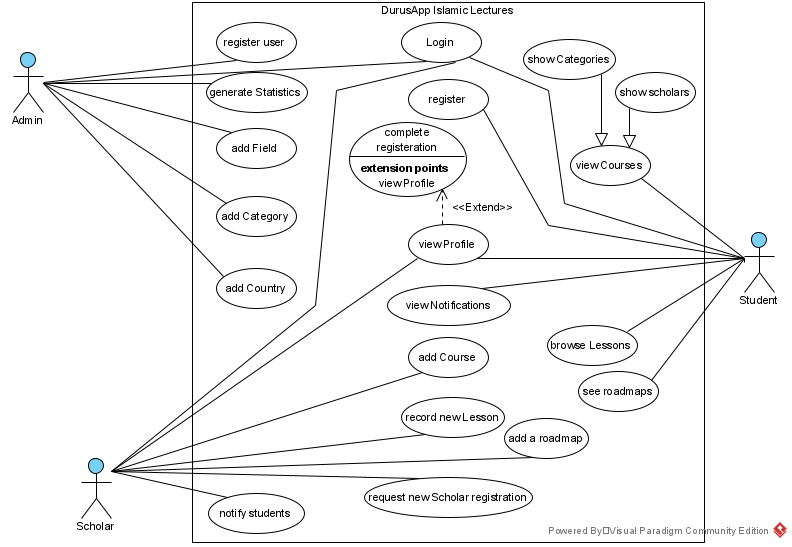
1. **Functions Analysis**

The main actors In DurusApp are: A student, scholar and Admin.

**A Scholar**: Once registered by the administration into DurusApp, a scholar logs into his/her account with the provided details. There, they are landed into a courses screen. The current timetable of courses offered by the scholar is presented. They can now either add a new course or click on a specific course to see lessons offered in these courses. The scholar will be presented by a dialog in case of adding a course. If, they proceed with the lessons, they can choose between a new lesson by recording or uploading a new file or send notification on the next lesson, whether ongoing or postponed. A scholar can alternatively choose to request another scholar’s registration. They also can complete their registrations by adding their profile pictures and other details. They can also add new roadmaps for students.

**A student** on the other hand, logs into the account with their details. if they don’t have accounts presented with register screens. If a student has no liked scholars or a particular course, they are landed onto a notifications and subscription dashboard, otherwise they are land onto a screen containing all the scholars or courses or categories. A click on any category or scholar lands the student onto a list of courses. The courses can be liked or the scholars to include it into the dashboard. The student can click on a course to view all the lectures currently offered and listen or download these audio files. Students are also expected to complete their profile. They can also view the different roadmaps given by different scholars.

**An Admin** logs into their account to register scholars, fields, categories and in rare cases students. They can discontinue courses, teachers or students. The most import part being generating user, courses and student reports.

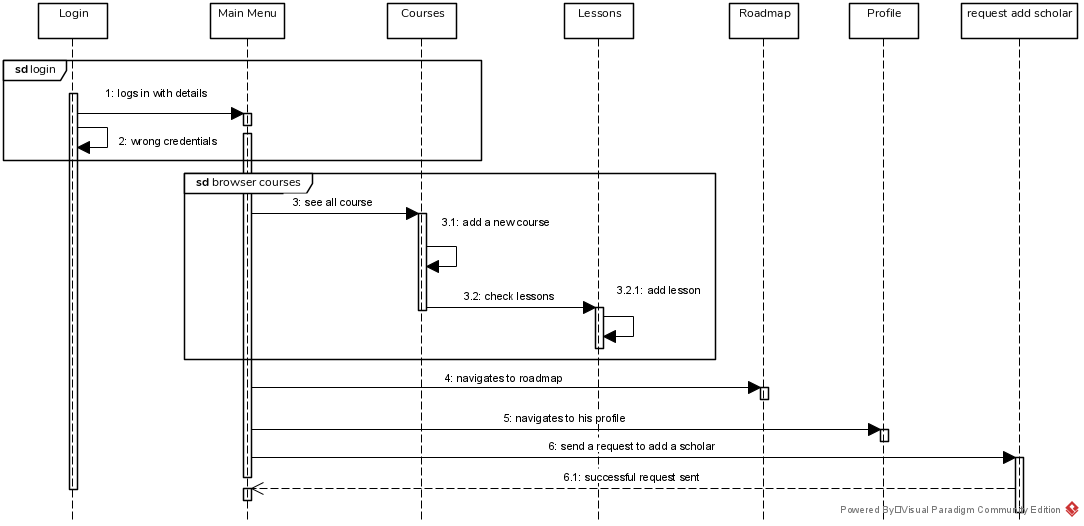


**Fig 4.0:** A use-case diagrams showing the different activities undertaken by users

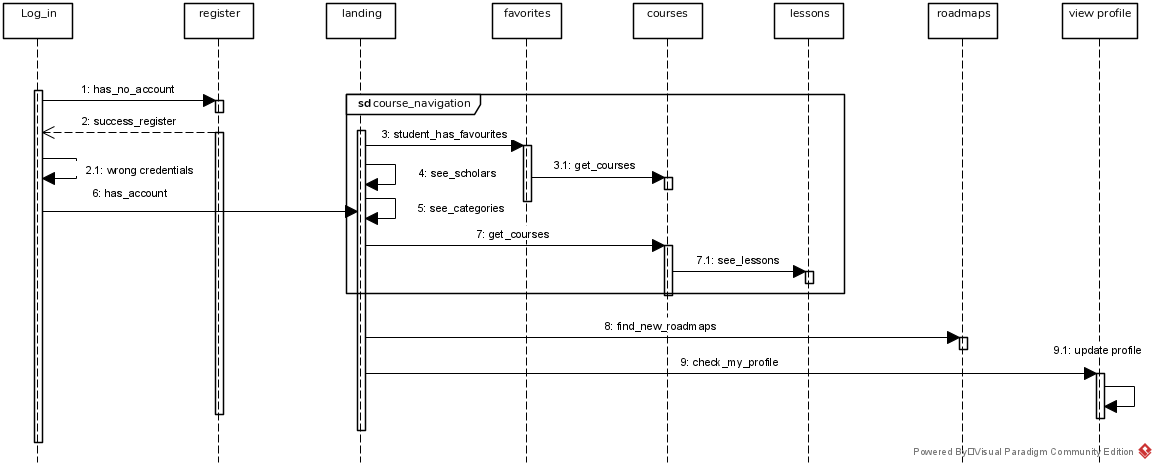
**Sequence Diagrams**

The activities presented above do not show the sequence of events. We thus use a Sequence diagram to explain the order over which the activities occur. Each entity will have its own sequence diagram.

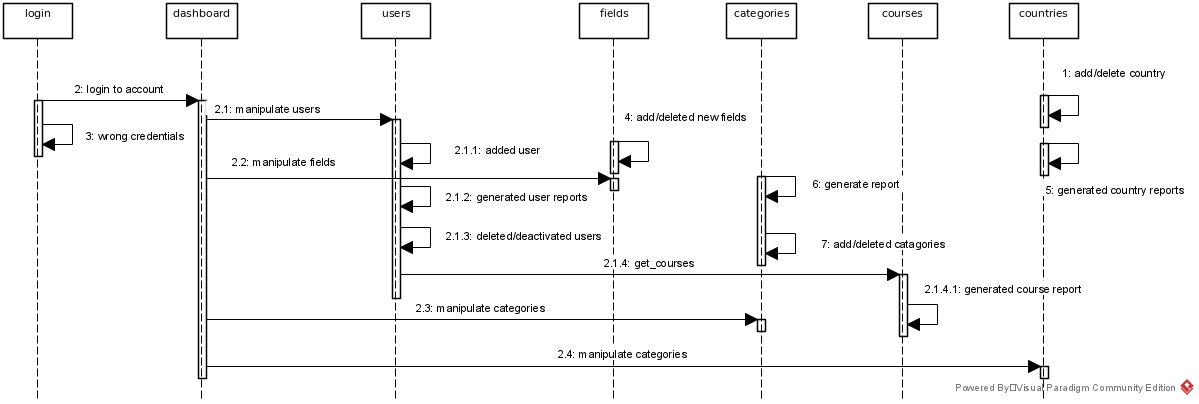
**Fig 4.1: Scholar Sequence**



**Fig 4.2: Student Sequence**



**Fig 4.3: Admin sequence**

****

1. **Data Input and flow Analysis**

When focusing on data, we try to analyze the data flow between the three main actors, now entities. A scholar is the main player here.

**Admin feeds into the system**

* Categories
* Students Fields
* Scholars Account
* Different countries.

**Scholars feed into the system**

* New courses
* New Lectures
* Update on existing lectures
* Complete their details
* Roadmaps

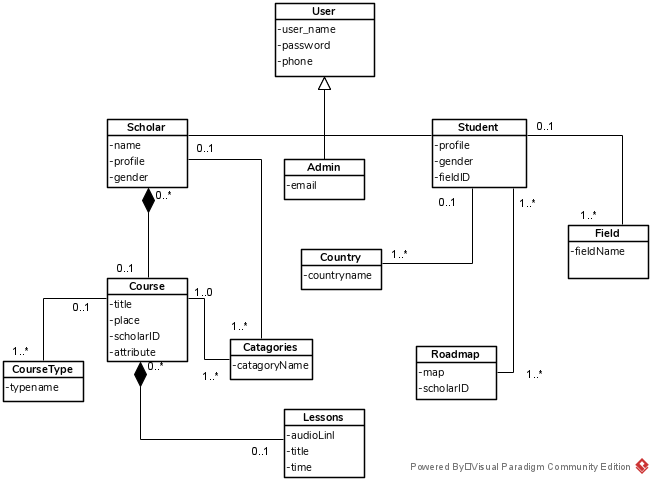
**Students feed into the system**

* Their details
* Scholars they follow
* Lectures they follow.

A close look into the data feed by everyone reveals the main entities in the application

* Users
  + Student
  + Scholar
  + Admin
* Courses
* Lectures/Lessons
* Roadmaps

The skeletal class diagram based on this entities is as follows:



The above diagram and relationships define the resulting skeleton for the database analysis. We use an Entity relation diagram to complete the tables or fields that may result from the data normalizations. Finally, we end up with a dynamic table that can summarize our database analysis:



A final ERD that shows different relationships between the data tables

**Data Tables**

1. **Users Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| F/P | Field name | Data type | Constraint | description |
| P | user\_id | Integer (10) | - | Auto incremented user identification |
|  | user\_name | Varchar (255) | **Unique** | Username, to login to accounts, cannot be duplicated |
|  | password | varchar (255) | - | Login password, saved as an encrypted string |
| F | user\_group\_id | Integer (10) | - | References the id in UserGroup table. Identifies user type: Scholar, student or Admin |

1. **UserGroup Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| F/P | Field name | Data type | Constraint | description |
| P | User\_group\_id | Integer (10) |  | Incremental identification for different types of users |
|  | User\_group\_name | Varchar (255) | unique | User group names |

1. **Scholars Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| F/P | Field name | Data type (Length) | Constraint | description |
| P | scholar\_id | Integer (10) | - | Auto incremented scholar identification |
|  | scholar\_name | Varchar(255) | - | The scholars full names. |
| F | user\_id | Integer(10) | - | References Users table. Shows the remaining details |
|  | email | Varchar (255) | - | Email address , any. For contact keeping |
|  | active | Integer (10) | - | Status of scholar: Active (1) and inactive (0) |
|  | title | Varchar(255) | - | The scholar’s title/specialization. |
|  | image | Varchar (255) | - | A link to the profile picture on the server |
|  | address | Varchar (255) | - | Scholars most common location |
| F | country\_id | Integer (10) | - | References the Country table. Tracks the scholar’s country. |

1. Students table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| F/P | Field name | Data type (Length) | Constraint | description |
| P | student\_id | Integer (10) | - | Auto incremented student identification |
|  | student\_name | Varchar(255) | Nullable | The students full names. |
| F | user\_id | Integer(10) | - | References Users table. Shows the remaining details |
|  | email | Varchar (255) | Nullable | Email address, any. For contact keeping |
|  | gender | Integer (10) | Nullable | Status of sudent: Active (1) and inactive (0) |
| F | field\_id | Varchar(255) | - | References Fields table. The students faculty/occupation |
|  | image | Varchar (255) | - | A link to the profile picture on the server |
| F | country\_id | Integer (10) | - | References the Country table. Tracks the students country. |

//and continue like this

**Security Analysis**

User Roles:

Admin can

* Access all the data
* Can alter the database directly
* Deactivate some users or delete them

Scholars

* See