# CHAPTER III

# RESEARCH METHODOLOGY

Each section discusses the approach to be used for the analysis and other technical specifications to help reinforce the proposal. It also includes diagrams, designs features techniques, and materials for implementing "iLearnCentral: A Cloud-Based Learning Center Platform with Mobile Technology" to fulfill the study's goals requirement.

## Software Engineering Methodology

iLearnCentral's development study uses the agile approach as the project framework for software engineering. Agile software development defines an approach to software development under which requirements and ideas progress through the collaborative effort of cross-functional self-organizing teams.

One of the benefits of the agile approach that suits this study is collaboration and open interactions with designers, advisers, and collaborators based on their feedback and any changes that occur throughout the development. It promotes flexible planning, structural growth, first conveyance, ongoing transition, and facilitates rapid and adaptable response to change.

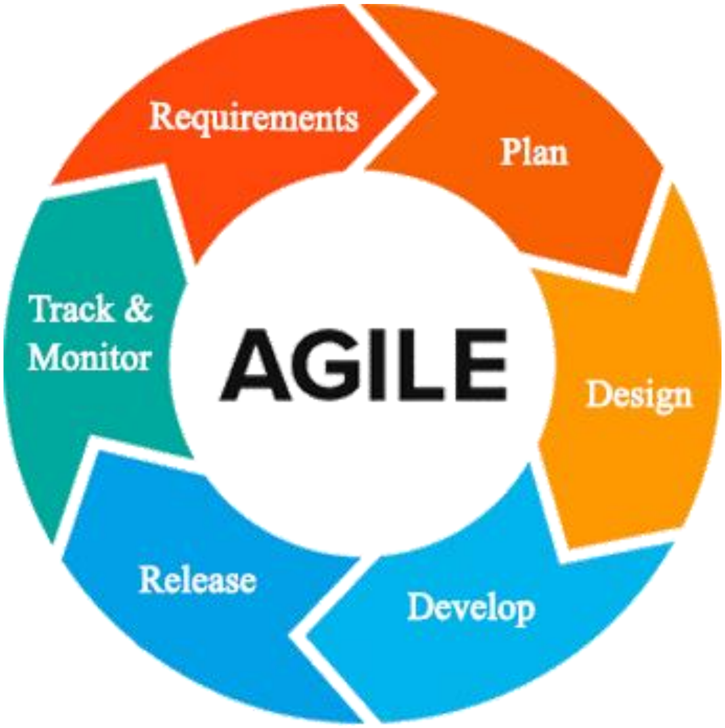


Figure 2: **Agile Development Methodology**

Figure 2 shows the representation of the framework lifecycle in an agile development methodology. The agile process requires less preparation, and the activities split into small increments. The agile process is for short-term projects with a team effort that meets the life cycle of software development (Sharma, 2012). By using customer feedback to agree on ideas, iteratively improves software This approach provides opportunities for assessing the path throughout the development lifecycle This performs by generic workflows, such as sprints or cycles to the end of which teams deliver a material increment that is potentially transmittable. This approach focuses on the replication of abbreviated work cycles and the functional yields of the product.

The developers do the following phases of the Agile Methodology:

**Requirement Analysis.** Defines the requirements for the iteration based on the product backlog, sprint backlog, customer and stakeholder feedback.

The gathered system features are from research conducted and interviews with industry experts in the related fields. The User Interfase (UI) designer and the programmer define the code specifications needed to fulfill the requirements of the project. The technical writer then takes note of changes and checks the document with all team members present. The database designer verifies if the features are compatible with the materials. The project manager reports the improvements made by the team to the team's adviser.

Initially, the team members make the primary manuscript and background researches on learning centers, educators, and job-seekers to lay out the things to do. In every iteration, the team members assigned to work on the obstacles analyze the issues and come up with a possible solution. They consult on resolutions with the other members. At the end of each day, the team members report on their progress.

**Plan Phase.** Phase of preparation involves creating a set of plans that helped guide the team through the phases of project implementation and closure. The plans produced during this process help developers manage time, cost, performance, change, risk, and issues to ensure the project is delivered on time and within budget by the developers.

The team determines schedules, preparations, and plans of actions to handle changes during the iteration. In every sprint cycle, the organizations made are directed towards the fulfillment of its intentions. Itemized priorities and time constraints are the focus of budget allocation by the project manager. The team establishes communication routes for questions and issues that may arise.

**Design Phase.** The specifications evaluated and defined by the designers are used in the design phase to make design choices using various diagrams. The UI designer assigned creates the user interface. The programmer and database designer must describe the device element interface mechanism. The project manager monitors the progress of the members' tasks. From the selected sprint backlog, the team determines which designs to tackle from the manuscript. There is a parallel development of mobile and web applications.

**Development Phase.** This step requires testing usability and reliability for all aspects of the product. The software testing checks if it meets all the specifications set out in the evaluation of requirements and if it handles the information correctly.

The developers check the software, analyze it, and identify the issues and update or modify it beyond the steps or requirements that were set up. Until deployment, all parts of the operation underwent a continuum of individual evaluation through different testing methods to ensure its efficacy and efficiency.

**Release.** Before releasing it to the market, developers carried out several activities to test the application. It allows the system to work within each operation of the deployment phase with tolerable performance and specific processes. Using the guidance given in the deployment document, developers then installed the application in the server environment.

**Track and Monitor.** This phase happens after the program is sent out to the customers/clients in this process. Developers maintain tracking, monitoring, and providing IT support services to include system and software updates and enhancements if appropriate. Feedback gathered from monitoring generates a list of improvements and bug fixes for the next iteration.

Another sprint cycle happens at the end of the previous. A sprint review with all members determines the set of activities for the next iteration. It includes adjustments from leftover unfinished tasks, additional features requested, and feedback from monitoring.

### Planning/Conception-Initiation Phase

The planning phase discusses the high-level decisions on why a project is valuable and what the requirements are. It helps the researchers keep track of assigned tasks, meeting deadlines, the progress of each requirement, and the budget for project work plans.

**Business Model Canvas**

The Business Model Canvas is a visual representation, commonly used by strategic managers, of existing and emerging business models.

Table 2

BUSINESS MODEL CANVAS

<table>
<tr><th>Key Partners</th><th>Key Activities</th><th>Value Proposition</th><th>Relationships</th><th>Customer Segment</th></tr>
<tr><td rowspan="3">-Learning centers<br>-Educators currently teaching in learning centers<br>-Job seeking educators</td><td>-Design and develop an intelligent school management software geared towards the needs of learning centers, educators, students</td><td rowspan="3">- System can be used by any type of learning center<br>- System could automate basic operations of administration with integrated artificial intelligence<br>-Recommend job vacancies to educators.<br>- Assist employed educators in classes<br>- Market learning center services and recommend courses to student</td><td>- Customer service hotlines<br>- User Feedback<br>- Email</td><td rowspan="3">- Learning center administration<br>- Educators in learning centers<br>- Students in learning centers<br>- Educators seeking employment</td></tr>
<tr><th>Key Resources</th><th>Channels</th></tr>
<tr><td>- Web domain and host<br>- Developers<br>- UI/UX designers<br>- Researchers</td><td>- On-location Visits<br>- Company Website<br>- Social Media Marketing<br>- Word of mouth</td></tr>
<tr><th colspan="3">Cost Structures</th><th colspan="2">Revenue Streams</th></tr>
<tr><td colspan="3">- Customer acquisition costs<br>- Research and Development<br>- Marketing and Advertising<br>- Hosting, Operations and Maintenance</td><td colspan="2">- Subsciption based on feature packages<br>- Ad Revenue from free or trial users</td></tr>
</table>

Table 2 illustrates the system’s Business Model Canvas. The Business Model Canvas is essential in building a flourishing business market. It gives concrete ideas to the researchers about the target market of the project and the cost of developing it. The Value Proposition shows the importance it gives to the public. Channels are a way for the group to interact simultaneously with customers and investors to sell the program. Customer relationships ensure that the entities involved are supporting our business relationship. Revenue streams demonstrates how we can earn revenue from the services provided.

## Program Workflow

Defining, managing, automating and optimizing business processes is a software workflow. Progressions of measures (tasks, events, interactions) involving a cycle of work, involving two or more individuals, and generating or adding value to the activities of the organization.

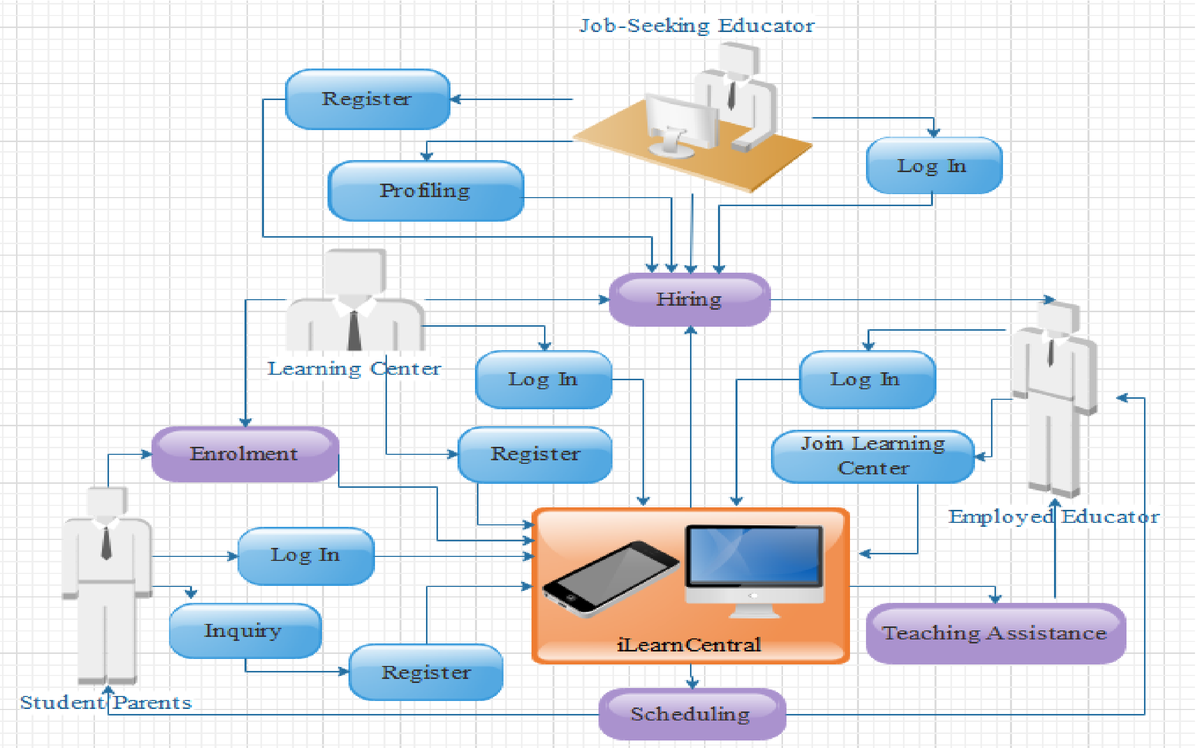


Figure 3: **User Activity Program Workflow**

Figure 3 shows the program workflow for general user activities. The administrative account creation and authentication starts with the registration of learning centers to the system. Job seekers register for an account to build their profile resume. The hiring module involves the learning center and job-seeking educator which could produce an employed educator. Only learning center and employed educator accounts can log in to most of the functionalities of iLearnCentral. Interested students/parents can inquire by creating a free account and browse through services offered by learning centers. Enrolment would involve input from both learning center and the student/parent. The scheduling is processed by iLearnCentral to produce calendars and notification to the employed educator and student/parent.

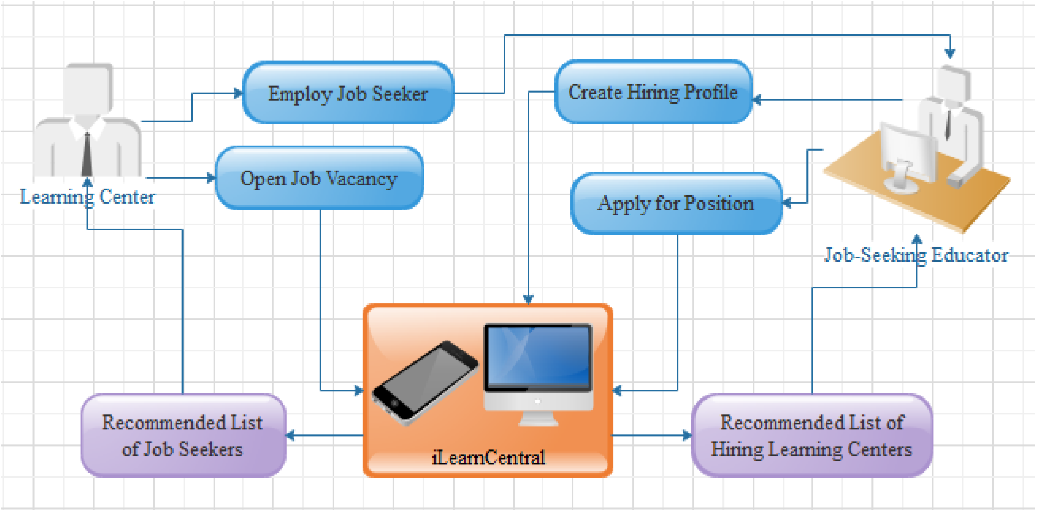


Figure 4: **Hiring Module Program Workflow**

Figure 4 details the hiring module from Figure 3. Job-seeking educators build their hiring profile or resume. After which the system processes their qualifications and determine a list of hiring learning centers from open job vacancies on which they apply for. They can also browse through other job vacancies available. On the other hand, learning centers receive recommended list of job-seeking profiles which fit their requirements.

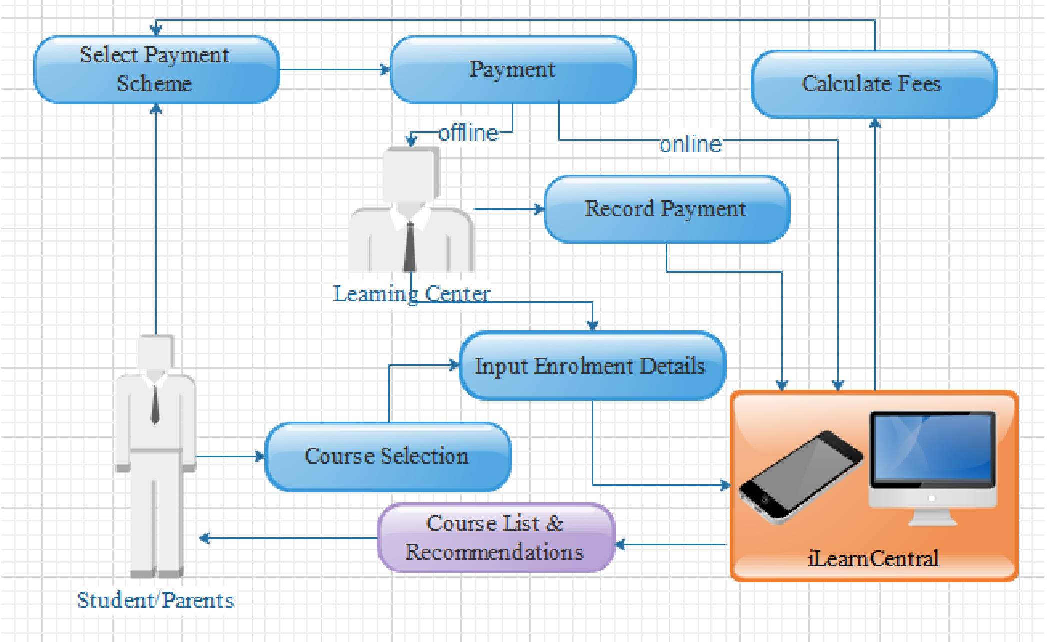


Figure 5: **Enrolment Module Program Workflow**

Figure 5 shows the program workflow for the enrolment module. The student or parent sees a list of courses from the system provided by the chosen learning center. With the selected course/s, they can process enrolment by providing the required information. The system calculates fees needed with the student/parent choosing methods of payment. Offline payment goes directly to the learning center, who would then record the transaction. Paying online is an option using various methods of online payments.

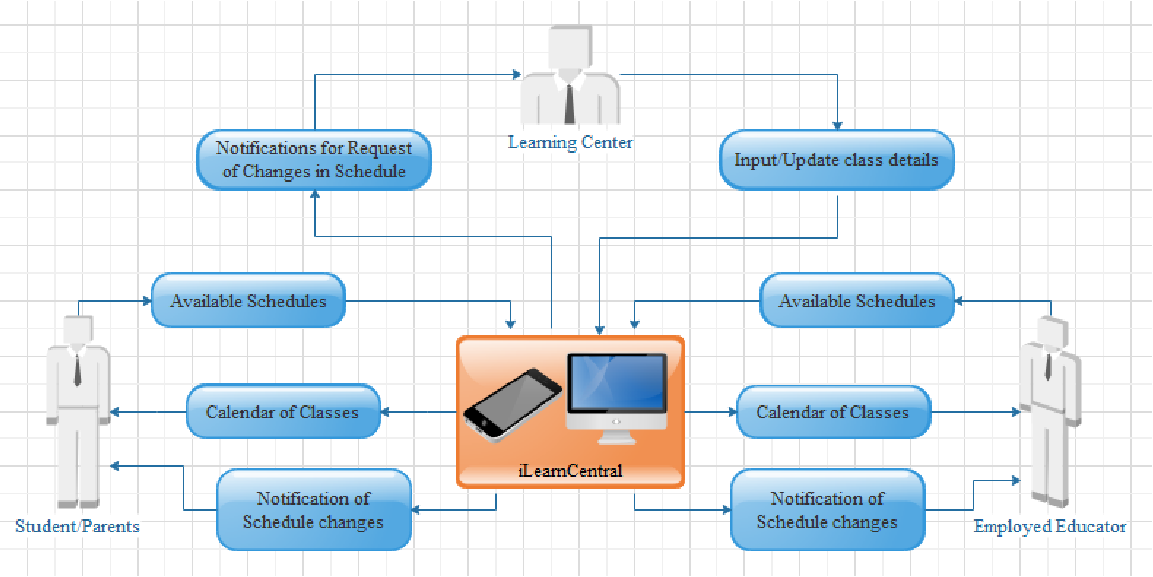

Learning Center
Notifications for Request of Changes in Schedule
Input/Update class details
Available Schedules
Available Schedules
Calendar of Classes
iLearnCentral
Calendar of Classes
Notification of Schedule changes
Notification of Schedule changes
Student/Parents
Employed Educator

Figure 6: **Scheduling Module Program Workflow**

Figure 6 shows the workflow for the scheduling module. The administrative staff would input class details for scheduling. The students and educators have time available when they can have a class. Schedules depend on matches with classes and educator's open loads. There should be a consideration for the classrooms available and the learning center's open business hours. Any changes to the schedule automatically adjusts schedules and notify all persons involved.

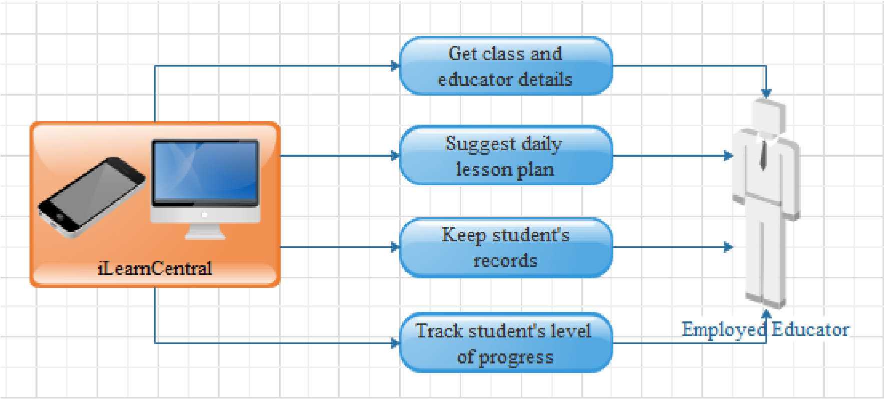

Get class and educator details
Suggest daily lesson plan
Keep student's records
Track student's level of progress
iLearnCentral
Employed Educator

Figure 7: **Teaching Assistance Module Program Workflow**

Figure 7 shows the program workflow for the teaching assistance module. It involves assisting educators with their day to day activities, including lesson plans, record keeping, and tracking student's progression.

### Validation Board (Stages 1 and 2)

Table 3 shows the different problems that our customers encountered. It also shows the solution to the problem being solved by the researcher. Table 3 also contains the most risky assumption, the methods and the criteria for success, the results and the decision, as well as the learning.

Table 3

VALIDATION BOARD

| Experiments | 1 | 2 | 3 |
|---|---|---|---|
| Customer | Learning Center Administration | Employed Educator | Job-Seeking Educators |
| Problem | Learning centers using manual transactions to support common management processes i.e. hiring, enrolment, and scheduling | Variation of lessons for different students handled, maintaining schedules, and keeping records | High turnover of educators in learning centers leading to constant demand amidst particular qualifications. |
| Solution | A dynamic learning center management system supporting different types of learning centers, i.e. day care, music, language studies | Adding a module for educators employed by a center to keep track of lessons, update schedules, and integrate records to the system. | Data pool of job-seeking educators sifted and recommended to fit learning centers' particular needs and vice versa. |
| Riskiest Assumption | Learning Center have no IT support | Learning center provide resources i.e. internet connectivity to employees | Educators uses the system to look for employment in learning centers |
| Method and Success Criteria | 60% of the respondents agree to use the system | 60% of the respondents agree to use the system | 60% of the respondents agree to use the system |

### Gantt Chart

The Gantt chart shows the scheduled work or activity completion in specific time frames in relation to the amount planned for the specified periods. The chart serves as a guide for the advocates to decide how long a project takes, classify the resources needed, and schedule the order of task completion performed by the researchers.

Table 4

GANTT CHART

| Task ID | Task Name | Task Lead | Start Date | End Date | August 2019 | | | | September 2019 | | | | October 2019 | | | | | Nov 2019 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 5 | 1 | 2 |
| 1 | Title Consultation | | Aug 16 | Sep 16 | 🟥 | 🟥 | 🟨 | 🟨 | 🟨 | 🟦 | | | | | | | | | |
| 2 | Project Proposal recommended | | Sep 16 | Sep 16 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟦 | | | | | | | | | |
| 3 | Discussion with adviser | Jephunneh | Sep 19 | Sep 23 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟨 | 🟦 | | | | | | | | |
| 4 | Preliminary research of the project | Cristian | Sep 23 | Sep 26 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟨 | 🟦 | | | | | | | |
| 5 | Writing and Compiling of Initial and Final Pages | Jephunneh | Sep 23 | Oct 26 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟨 | 🟨 | 🟨 | 🟨 | 🟨 | 🟨 | 🟦 | | |
| 6 | Writing of Chapter 1 | Cristian | Sep 23 | Sep 26 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟨 | 🟦 | | | | | | | |
| 7 | Consultation with adviser on Chapter 1 | Jephunneh | Sep 27 | Sep 27 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟦 | | | | | | | |
| 8 | Preparing of Transmittal Letter/s | Rhea Shane | Sep 16 | Sep 23 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟨 | 🟦 | | | | | | | | |
| 9 | Preparing of Questionnaires | Rhea Shane | Sep 16 | Sep 23 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟨 | 🟦 | | | | | | | | |
| 10 | Interviews/Surveys | Rhea Shane | Sep 23 | Oct 11 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟨 | 🟨 | 🟨 | 🟦 | | | | | |
| 11 | Finalizing of Chapter 1 | Cristian | Sep 27 | Sep 28 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟦 | | | | | | | | |

| | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 12 | Writing of Chapter 2 | Cristian | Sep 29 | Oct 2 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟨 | 🟦 | | | | | | |
| 13 | Consultation with adviser on Chapters 1 and 2 | Jephunneh | Oct 2 | Oct 2 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟦 | | | | | | |
| 14 | Finalizing of Chapter 2 | Cristian | Oct 2 | Oct 4 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟦 | | | | | | |
| 15 | Designing User Interfaces | Cristian | Oct 7 | Oct 17 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟨 | 🟦 | | | | |
| 16 | Writing of Chapter 3 | Rhea Shane | Oct 7 | Oct 17 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟨 | 🟦 | | | | |
| 17 | Consultation with adviser on Chapters 1, 2, and 3 | Jephunneh | Oct 18 | Oct 18 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟦 | | | | |
| 19 | Consultation with Technical Editor | Rhea Shane | Oct 22 | Oct 25 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟦 | | | |
| 20 | Finalizing of Chapter 3 | Rhea Shane | Oct 24 | Oct 26 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟦 | | | |
| 21 | Concept Video & Presentation preparation | Jephunneh | Oct 21 | Oct 30 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟨 | 🟦 | | |
| 22 | Final consultation with adviser | Jephunneh | Oct 24 | Oct 25 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟦 | | | |
| 23 | Compiling and finalizing of Manuscript | Rhea Shane | Oct 24 | Oct 26 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟦 | | | |
| 24 | Poster design & preparation | Cristian | Oct 17 | Oct 20 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟨 | 🟦 | | | |
| 25 | Proposal presentation | | Nov 6 | Nov 9 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 | 🟦 |

Table 4 is the Gantt chart for the development of iLearnCentral's proposal manuscript. Every activity is performed in three different colors: red means that the activity is still incomplete, yellow means that the activity is still on the way, and blue means that the activity is already finished.

### Functional Decomposition Diagram

The functional decomposition diagram demonstrates the operative relationship between the various components of the project into critical modules to clearly illustrate and simplify various activities.

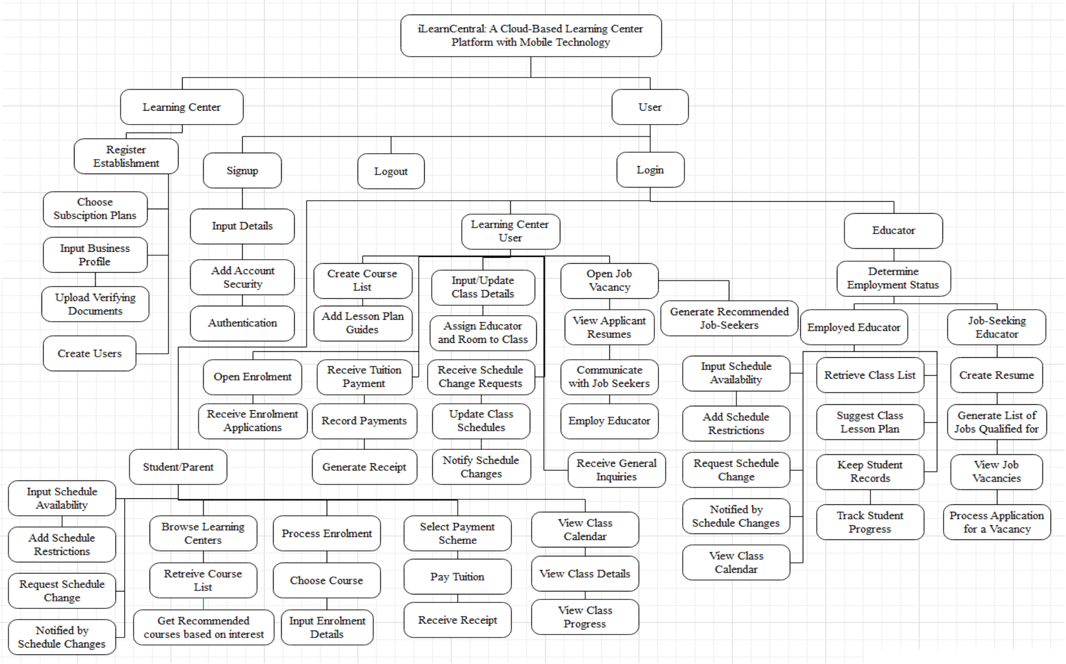


Figure 8: **Functional Decomposition Diagram**

Figure 8 shows the functional decomposition diagram of the system. It contains the five modules to complete and sub-functions for each module concerning the user type needing the functionality.

### Analysis / Design Phase

The stage of analysis includes the concept of the specifications needed to accomplish the method. Each step determines the problem to be solved by the customer.

## Use Case Diagram

Use case diagram shows the graphic representation of the mechanism of iLearnCentral and potential sequences of interactions between systems and users in a specific environment related to a specific target.

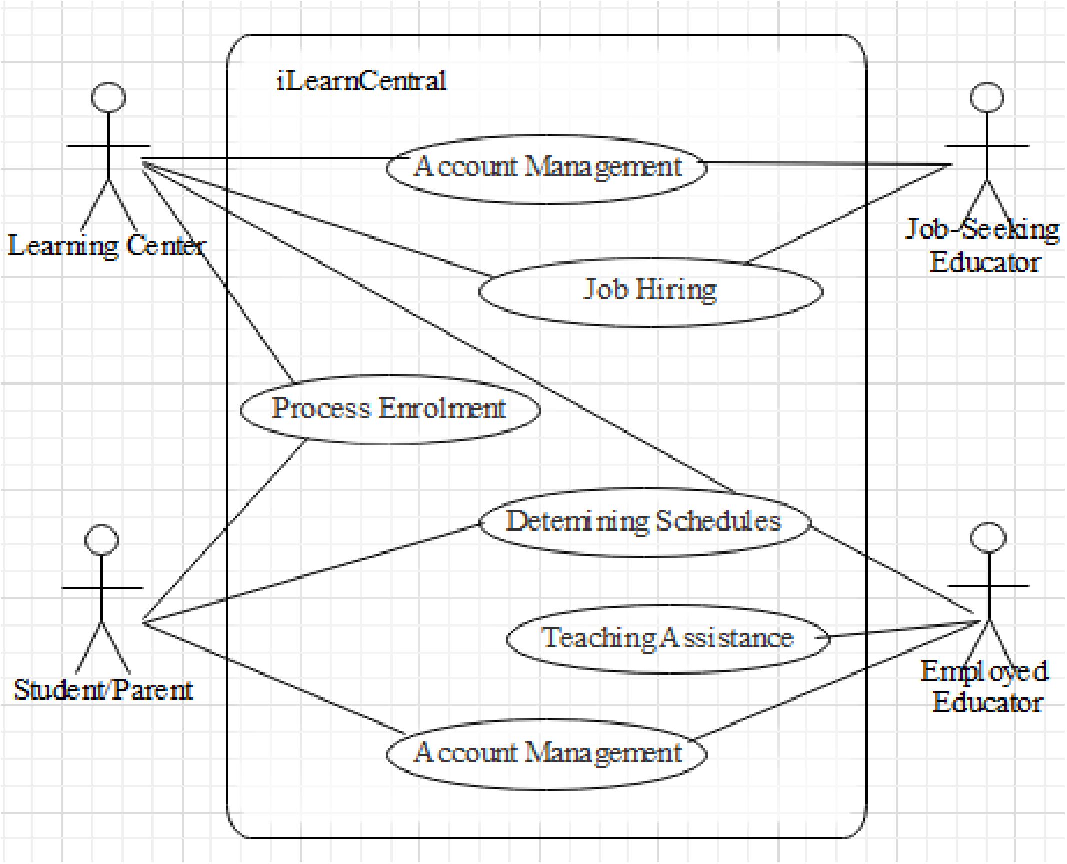


Figure 9: **Use Case Diagram**

Figure 9 shows the use case diagram for iLearnCentral. It shows the outside view of the system and the requirements needed. It identifies the system's influencing external and internal factors and their interactions.

The learning center is a factor in most of the internal modules. Account management involves all actors with varying degrees of complexity for each actor. Job hiring only concerns with the learning center and the job-seeking applicant. Enrolment processing is between the learning center and the students/parents. Determining schedules need the interaction between the learning center, student/parent, and assigned educator. Lastly, teaching assistance is a module for employed educators to utilize.

## Storyboard

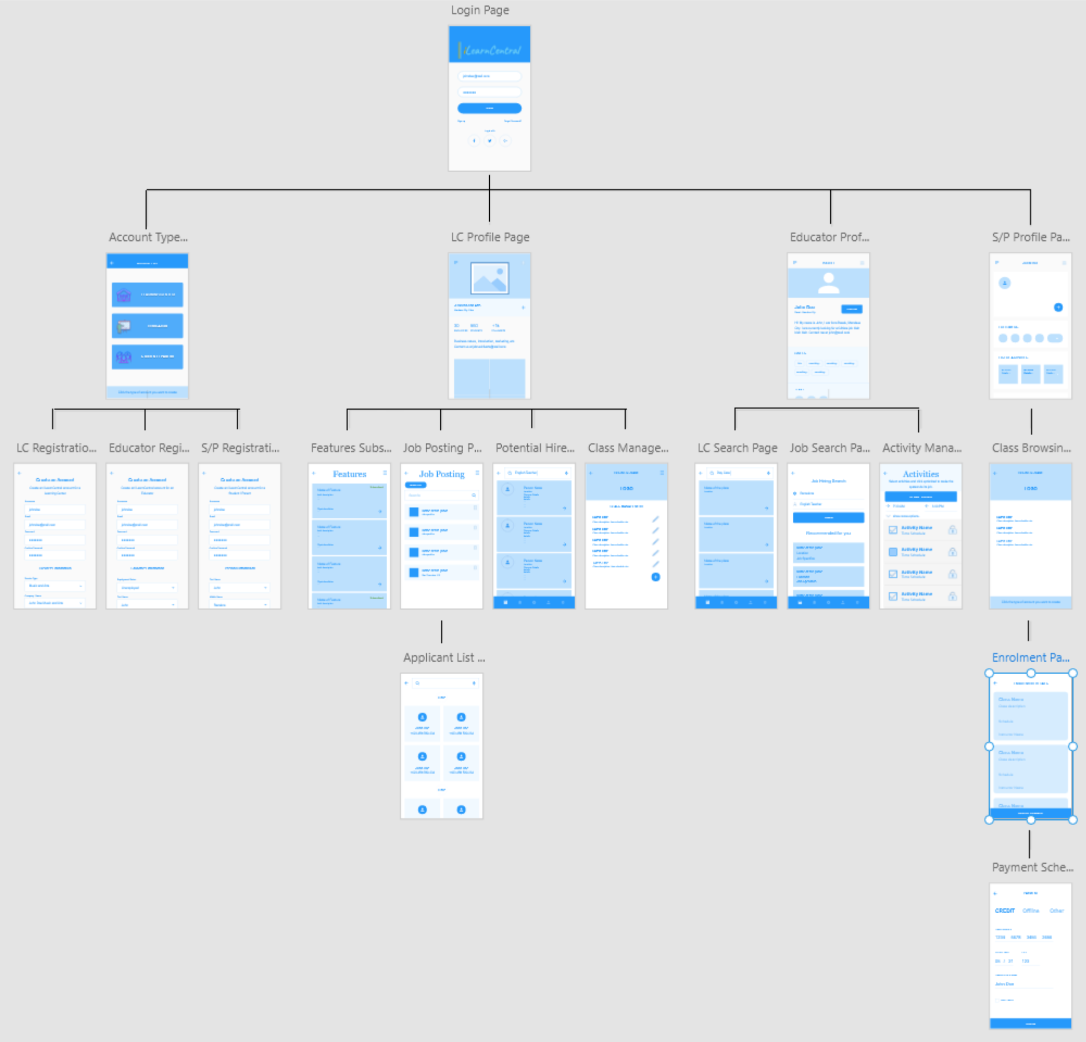


Figure 10: **iLearnCentral Storyboard**

Figure 10 shows the graphical representation of the usage of the app. New users have the option to register as a Learning Center, Educator or Student/Parent and each type of registered user has their own designated profile pages and features.

Learning Centers are able to check their subscriptions, post new jobs, search and hire educator and manage classes. Educators are also able to search for learning centers and jobs, and manage their activities or classes. Lastly, Students/Parents can search for available classes that interest them, view enrolled classes and pay classes on multiple options.

### User Interface Diagram

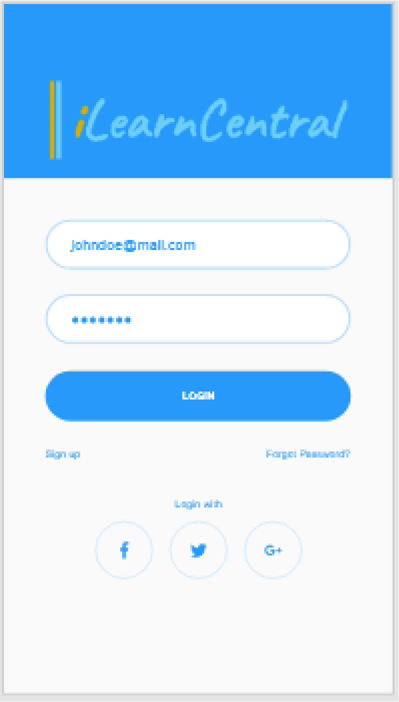
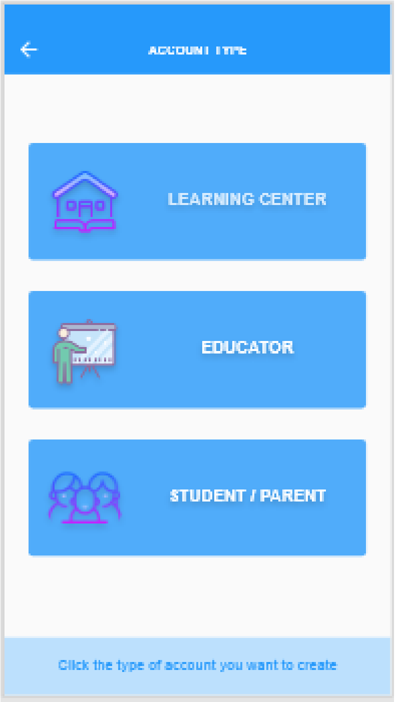


Figure 11: **Login Page**

Figure 11 shows the Login Page. The user can enter their credentials to login. This page also provides links to the registration page and forgot password support page.



ACCOUNT TYPE

LEARNING CENTER

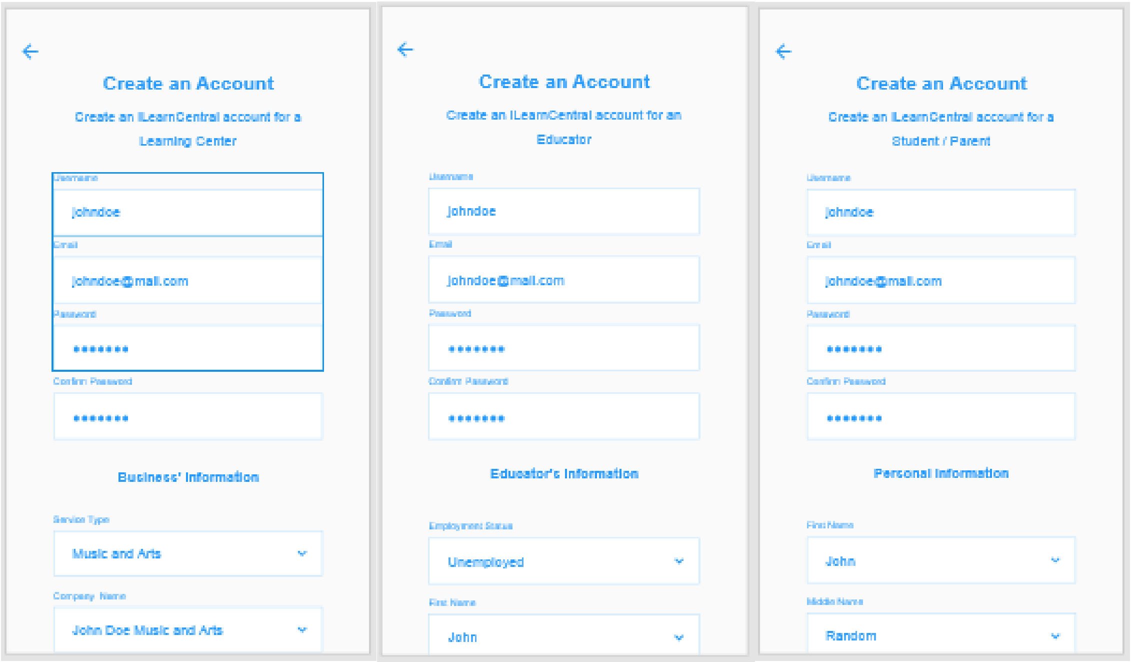
EDUCATOR

STUDENT / PARENT

Click the type of account you want to create

Figure 12: **<u>Account Type Selection Page</u>**

There are three type of users – educator, student/parent and learning center. Users can select the type of account they would like to create.



Create an Account

Create an ILearnCentral account for a Learning Center

Username: johndoe

Email: johndoe@mail.com

Password: •••••••

Confirm Password: •••••••

Business' Information

Service Type: Music and Arts

Company Name: John Doe Music and Arts

Create an Account

Create an ILearnCentral account for an Educator

Username: johndoe

Email: johndoe@mail.com

Password: •••••••

Confirm Password: •••••••

Educator's Information

Employment Status: Unemployed

First Name: John

Create an Account

Create an ILearnCentral account for a Student / Parent

Username: johndoe

Email: johndoe@mail.com

Password: •••••••

Confirm Password: •••••••

Personal Information

First Name: John

Middle Name: Random

Figure 13: **<u>Sign up Page</u>**

Figure 13 shows the different pages for each of the user sign up types. The sign up page for learning centers is different from the educator and student/parent because the sign up for learning centers require them to specify the type of learning center that they have. The pages show required information for the registration (e.g. First Name, Middle Name, Last Name, Username and Password). Once filled out, users can click on 'Register' button to complete the registration or to cancel by clicking the 'Cancel' button.

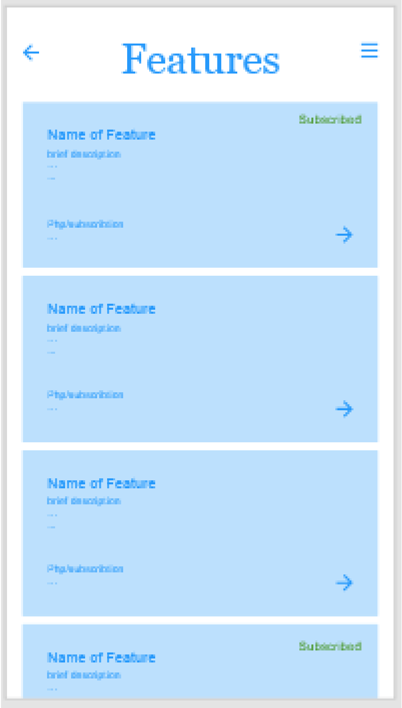
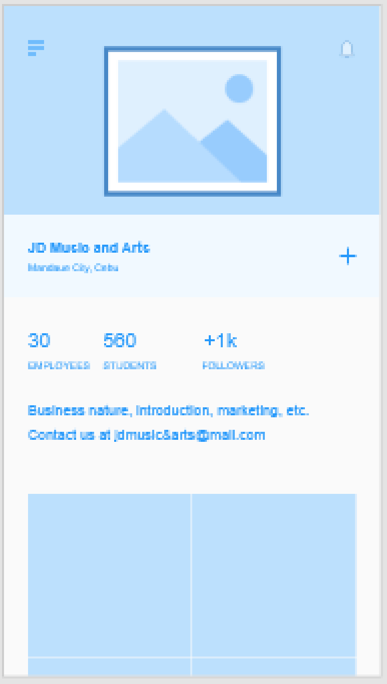


Figure 14: **<u>Feature Subscription Page</u>**

Figure 14 shows the pre-defined features the app offers. After logging in, users select features on this page and install them as an add-on. A feature is either free or subscription-based.



JD Music and Arts

Mandaue City, Cebu

Business nature, introduction, marketing, etc.

Contact us at jdmusic&arts@mail.com

Figure 15: **<u>Learning Center Profile Page</u>**

Figure 15 shows the profile of a learning center. This includes the number of employees, students, followers and contact information.

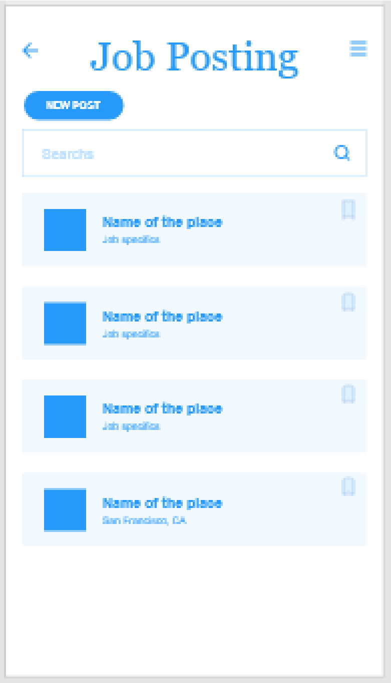


Figure 16: **<u>Job Posting Page</u>**

Figure 16 shows the jobs posted by a learning center. Learning centers also have the option to add new job posting by clicking on the 'New Post' button.

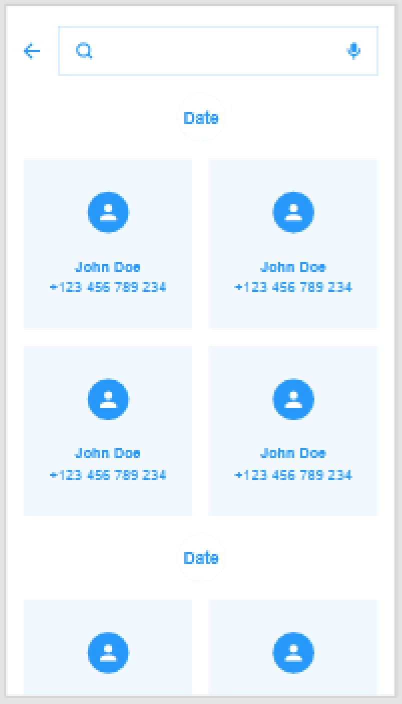


Figure 17: **<u>Applicants List Page</u>**

Figure 17 shows the list of applicants on a posted job. Learning centers have the option to select and hire the educator of their choice.

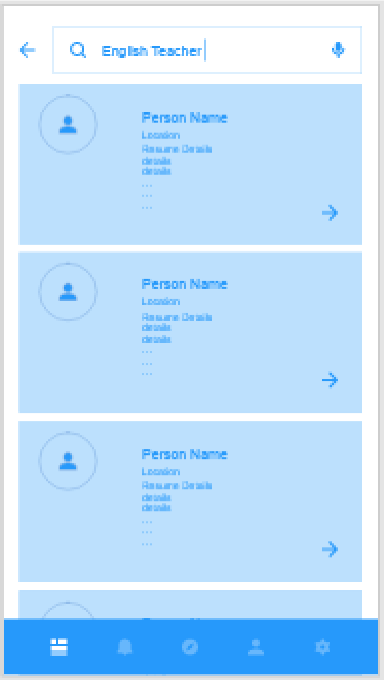
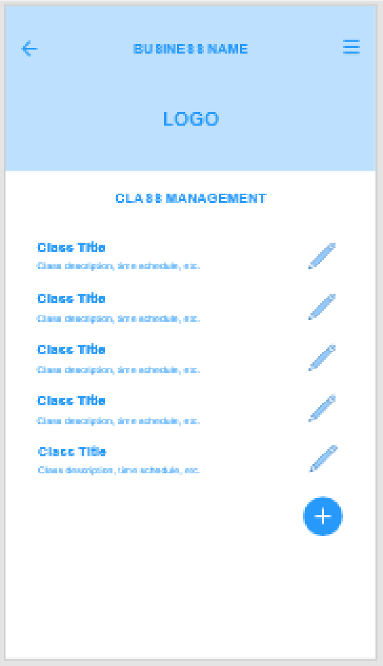

English Teacher

Figure 18: **<u>Potential Hire Search Page</u>**

Figure 18 shows the list of the best applicants determined by the app. Learning centers can search for available educators by use of keywords. Through its Machine Learning, it is able to suggest qualified educators to the posted jobs.



Class Title
Class description, time schedule, etc.

Class Title
Class description, time schedule, etc.

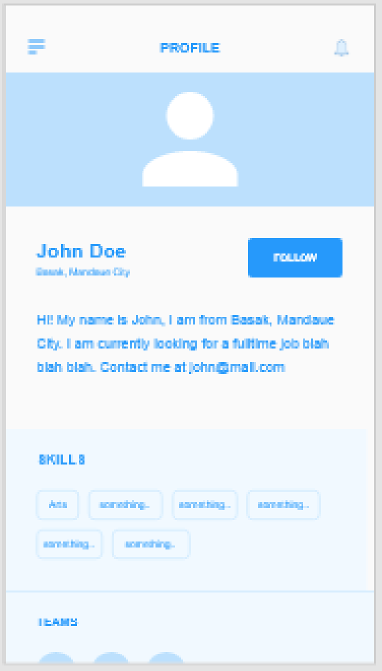
Class Title
Class description, time schedule, etc.

Class Title
Class description, time schedule, etc.

Class Title
Class description, time schedule, etc.

Figure 19: **Class Management Page**

From the Class Management Page, learning centers are able to view all its offered classes. They are also able to add new classes and schedules.

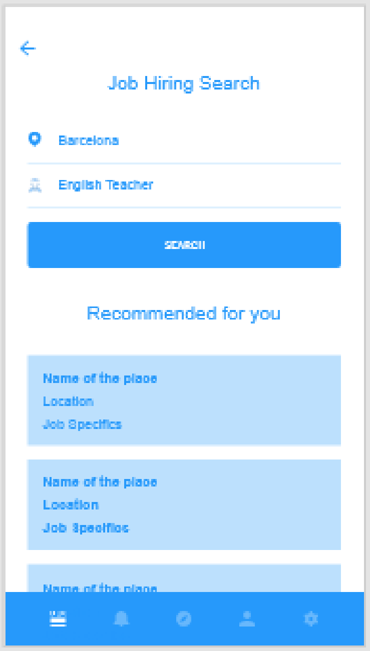


John Doe
Basak, Mandaue City

Hi! My name is John, I am from Basak, Mandaue City. I am currently looking for a fulltime job blah blah blah. Contact me at john@mail.com

Figure 20: **Educator's Profile Page**

After registration, they can edit/modify their profile by adding a short description of themselves, skills, etc. from the Educator's Profile Page. This page also shows whether or not an educator is currently employed to specific learning center/s.

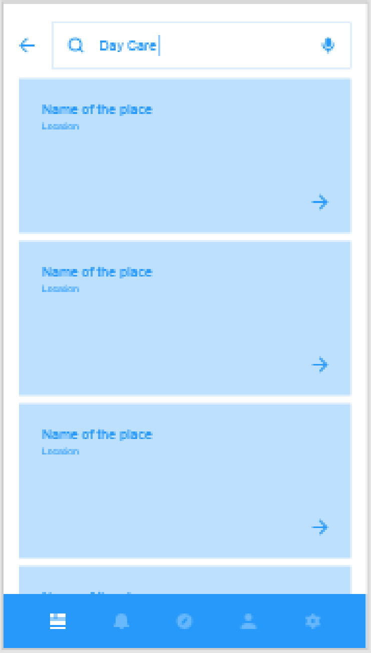


Barcelona

English Teacher

Figure 21: **<u>Job Search Page</u>**

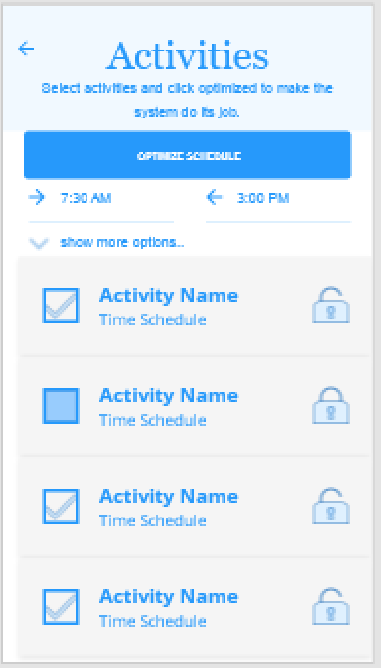
The Job Search Page allows educators to look for posted jobs within their location. They can specify a job search by using keywords. There is a list of recommendations which educators can view and apply.



Day Care

Figure 22: **<u>Learning Center Search Page</u>**

Learning Center Search page allows educators to search for specific type of learning centers.



Select activities and click optimized to make the system do its job.

Figure 23: **<u>Activity Management Page</u>**

The Activity Manage Page allows educators to monitor, administer classes and specify activities for the day. This ensures that there is no overlapping of tasks or classes and educators are on top of their schedules.

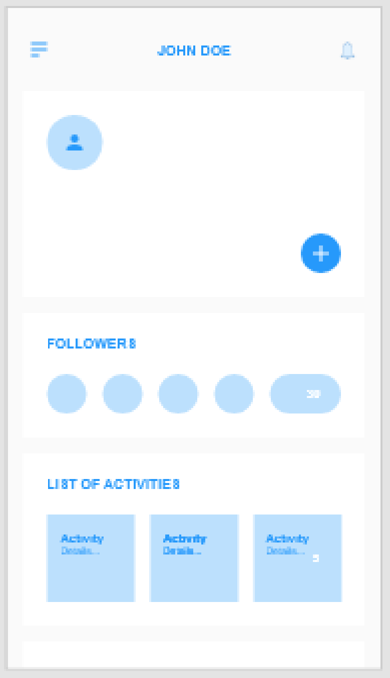


Figure 24: **<u>Student/Parent Profile Page</u>**

The Student/Parent Profile Page mainly shows the classes/activities and basic information about the user.

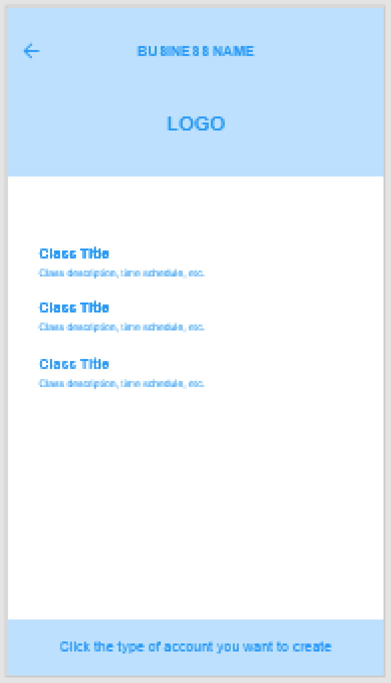

BUSINESS NAME
LOGO
Class Title
Class description, time schedule, etc.
Class Title
Class description, time schedule, etc.
Class Title
Class description, time schedule, etc.
Click the type of account you want to create

Figure 25: **Class Browsing Page**

In figure 25, students/parents are able to search for classes and schedules from the Class Browsing Page.

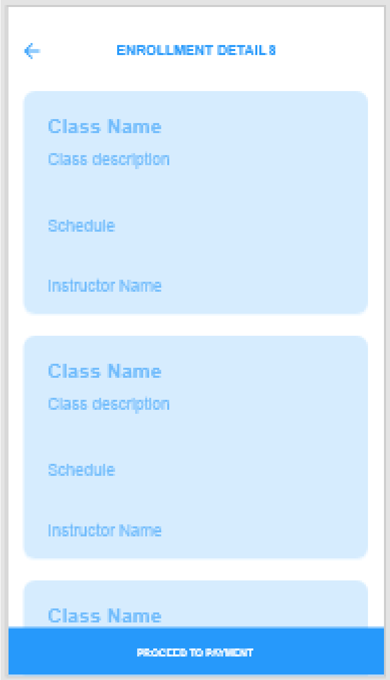
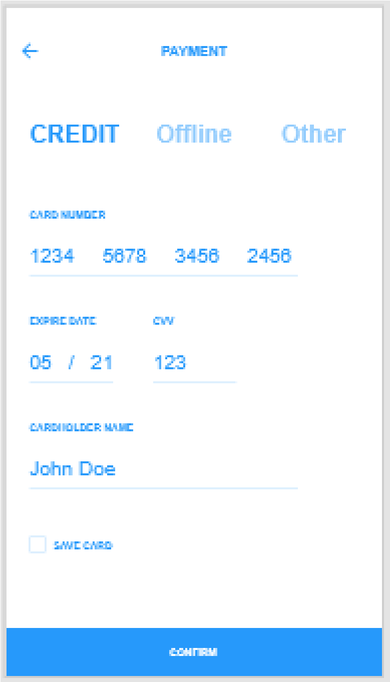


Figure 26: **Enrolment Page**

The Enrolment Page is a digitized form where students process the enrolment of the classes they selected.



CARD NUMBER

1234 5678 3456 2456

EXPIRE DATE

05 / 21

CVV

123

CARDHOLDER NAME

John Doe

□ SAVE CARD

Figure 27: **Payment Scheme Page**

The Payment Scheme Page shows the multiple options for students/parents to pay their classes.

## Database Design

The database to use is NoSQL due to the advantages it provides with data volume, velocity, and variety. It allows for better adaptability to changes in schema when using agile development. It is scalable and accessible to multitudes of users, which is necessary to a cloud-based system.

This section shows the designed NoSQL schema. The designing process follows the Query Driven Design that optimizes access instead of storage. It is by no means the final structure of the schema as changes may arise during the development process.

A document-oriented database, one of the main categories of NoSQL databases, is a computer program designed to store, retrieve, and handle document-oriented information, also known as semi-structured data. It is inherently a subclass of the key-value store and relies on an internal structure in the document to extract metadata that the database engine uses for further optimization. The current list of features in the documents presented in this section are basic details and more can be added or altered depending on the progress during development phase.

Table 5

USER DOCUMENT

| User | | |
|---|---|---|
| PK | username | |
| | password | |
| | emailAddress | |
| | accountType | |
| | securityQuestions [ ] | |
| | | question |
| | | answer |
| | accountStatus | |

Table 5 is the document database design for all user accounts. The collection of users is solely for account management. Depending on the type of account type, the system proceeds differently. The security questions are the means to provide validation in the event of resetting or retrieving forgotten passwords.

Table 6

LEARNING CENTER DOCUMENT

| LearningCenter | | |
|---|---|---|
| PK | centerID | |
| | accounts [ ] | |
| FK | | username |
| | | accessLevel |
| | | status |
| | businessName | |
| | serviceType | |
| | businessAddress | |
| | | buildingNo |
| | | buildingName |
| | | streetName |
| | | subdivision |
| | | barangay |
| | | district |
| | | city |

| | | |
|---|---|---|
| | | province |
| | | country |
| | | zipCode |
| | contactEmail | |
| | contactNumber [ ] | |
| | companyWebsite | |
| | operatingDays [ ] | |
| | openingTime | |
| | closingTime | |
| | coursesOffered [ ] | |
| | subscriptionType | |
| | subsciptionEndDate | |

Table 6 is the document database design for learning center entities. It records the information about learning centers, including data on identity, operating hours, and subscription to the system. The address is necessary to have segmented documentation for easier processing by the recommendation system in the hiring module.

Table 7

EDUCATOR DOCUMENT

| Educator | | |
|---|---|---|
| PK | educatorID | |
| FK | username | |
| | name | |
| | | firstName |
| | | middleName |
| | | lastName |
| | | extension |
| | birthday | |
| | employmentStatus | |
| FK | centerID | |
| | position | |
| | address [] | |
| | | houseNo |
| | | streetName |
| | | subdivision |
| | | barangay |
| | | district |
| | | city |
| | | province |

| | |
|---|---|
| | country |
| | zipCode |
| | currentAddress |
| | email |
| | phoneNo |
| | gender |
| | maritalStatus |
| | religion |
| | citizenship |

Table 7 is the document database design for educator entities. It holds the primary information of an educator and represents both employed or job-seeking educators. The employment status and accompanying centerID determines the state of an educator.

Table 8

RESUME DOCUMENT

| Resume | |
|---|---|
| PK | resumeID |
| FK | educatorID |
| | careerObjective |
| | educationalHistory [ ] |
| | educationLevel |
| | schoolName |
| | schoolAddress |
| | course |
| | major |
| | yearStart |
| | yearEnd |
| | graduated |
| | employmentHistory [ ] |
| | companyName |
| | companyAddress |
| | position |
| | dateStart |
| | dateEnd |
| | skills [ ] |
| | qualities [ ] |
| | interests [ ] |
| | awards [ ] |
| | references [ ] |

| | | |
|---|---|---|
| | | referenceName |
| | | affiliation |
| | | position |
| | | contactInfo |

Table 8 is the document database design for resume entries. It represents the accompanying resume of an educator account and provides the usual information about a job seeker.

Table 9

STUDENT DOCUMENT

| Student | | |
|---|---|---|
| PK | studentID | |
| FK | username | |
| | name | |
| | | firstName |
| | | middleName |
| | | lastName |
| | | extension |
| | birthday | |
| | address [] | |
| | | houseNo |
| | | streetName |
| | | subdivision |
| | | barangay |
| | | district |
| | | city |
| | | province |
| | | country |
| | | zipCode |
| | | currentAddress |
| | email | |
| | phoneNo | |
| | gender | |
| | maritalStatus | |
| | religion | |
| | citizenship | |
| FK | centerID | |
| | enrolmentStatus | |
| | enrolmentHistory [ ] | |

Table 9 is the document database design for student entities. Parents and students get one account in our system as they do not have a difference in functionalities directed to them. The expectation is for parents to handle the account for minor students. The document also contains the enrolment history of the student.

Table 10

JOB VACANCY DOCUMENT

| JobVacancy | |
|---|---|
| PK | vacancyID |
| FK | centerID<br>status |
| | position |
| | jobDescription |
| | jobType |
| | educationalRequirements |
| | educationalLevel |
| | degrees [ ] |
| | majors [ ] |
| | minimumUnits |
| | qualifications [ ] |
| | skills [ ] |
| | incentives [ ] |
| | responsibilities [ ] |
| | requirements [ ] |
| | applicationMethod [ ] |

Table 10 is the document database design for job vacancy events. The job vacancy has to be made by a learning center. It has data on the position to be filled and all pertinent information required to qualify a job-seeker to the job.

Table 11

JOB APPLICATION DOCUMENT

| Job Application | |
|---|---|
| PK | jobApplicationID |
| FK | educatorID |
| FK | vacancyID |
| | applicationDate |
| | applicationStatus |
| | preferredMethod |
| | message |

Table 11 is the document database design for job application events. A job application happens when a job seeker applies for an available job vacancy. The learning center receives a list of recommended applicants as well as job-seekers who manually applied.

Table 12

COURSE DOCUMENT

| Course | |
|---|---|
| PK | courseID |
| FK | centerID |
| | courseName |
| | courseDescription |
| | tuition |

Table 12 is the document database design for course entities. The courses are services offered by a learning center and the basis for enrolment and classes.

Table 13

ENROLMENT DOCUMENT

| Enrolment | |
|---|---|
| PK | enrolmentID |
| FK | centerID |
| FK | studentID |
| | enrolmentDate |
| | dateClassStart |
| | dateClassEnds |
| | noOfHours |
| | enrolmentStatus |

Table 13 is the document database design for enrolment events. Details of an enrolment process are stored here. Information about the learning center and student involved retrieves from their document store via foreign keys.

Table 14

PAYMENT DOCUMENT

| Payment | |
|---|---|
| PK | paymentID |
| FK | enrolmentID |

| | |
|---|---|
| | tuition |
| | additionalFees |
| | balance |
| | payments [ ] |
| | paymentDate |
| | amount |
| | paymentMethod |
| | validated |
| | status |

Table 14 is the document database design for a payment plan. An entry of the payment document is a counterpart of an enrolment. It records the progress of payments made, be it one-time full payment or each staggering pay. The record also contains the details of the fees needed.

Table 15

CLASS SESSION DOCUMENT

| ClassSession | |
|---|---|
| PK | classSessionID |
| FK | courseID |
| FK | enrolmentID |
| FK | educatorID |
| | date |
| | timeStart |
| | timeEnd |
| | roomNo |

Table 15 is the document database design for a class session. Class sessions contain details of meetups between students and educators. Learning centers are tasked to set up the classes.

Table 16

SCHEDULE REQUEST DOCUMENT

| ScheduleRequest | |
|---|---|
| PK | scheduleRequestID |
| | restrictionType |
| FK | educatorID |
| FK | studentID |
| FK | centerID |
| FK | classSessionID |

| | |
|---|---|
| | scheduleDate |
| | timeStart |
| | timeEnd |

Table 16 is the document database design for schedule requests. The requests are days and times a student or educator scheduled to open or restrict and help the scheduling module determine when to place classes.

Table 17

LESSON PLAN DOCUMENT

| LessonPlan | |
|---|---|
| PK | lessonID |
| FK | courseID |
| | topic |
| | overview |
| | objective [ ] |
| | materials [ ] |
| | activities [ ] |
| | procedures [ ] |

Table 17 is the document database design for lesson plans. It contains the different sections in building lesson plans. An educator may add multiple instances of each part. Lesson plans are reusable and shareable across educators within the learning center.

Table 18

STUDENT RECORD DOCUMENT

| StudentRecord | | |
|---|---|---|
| PK | studentRecordID | |
| FK | enrolmentID | |
| | lessonPlansCovered [ ] | |
| | classSession [ ] | |
| FK | | classSessionID |
| | | remarks |
| | studentScores [ ] | |
| | | testTitle |
| | | score |
| | | perfectscore |
| | | testDate |

Table 18 is the document database design for student records. It means to keep track of student progress and data. It links to lesson plans and histories of sessions attended. It records test scores, if available.

### Entity-Relationship Diagram

The entity-relationship diagram graphically demonstrates the interactions of entities, activities, events, and relationships across all modules of the system.

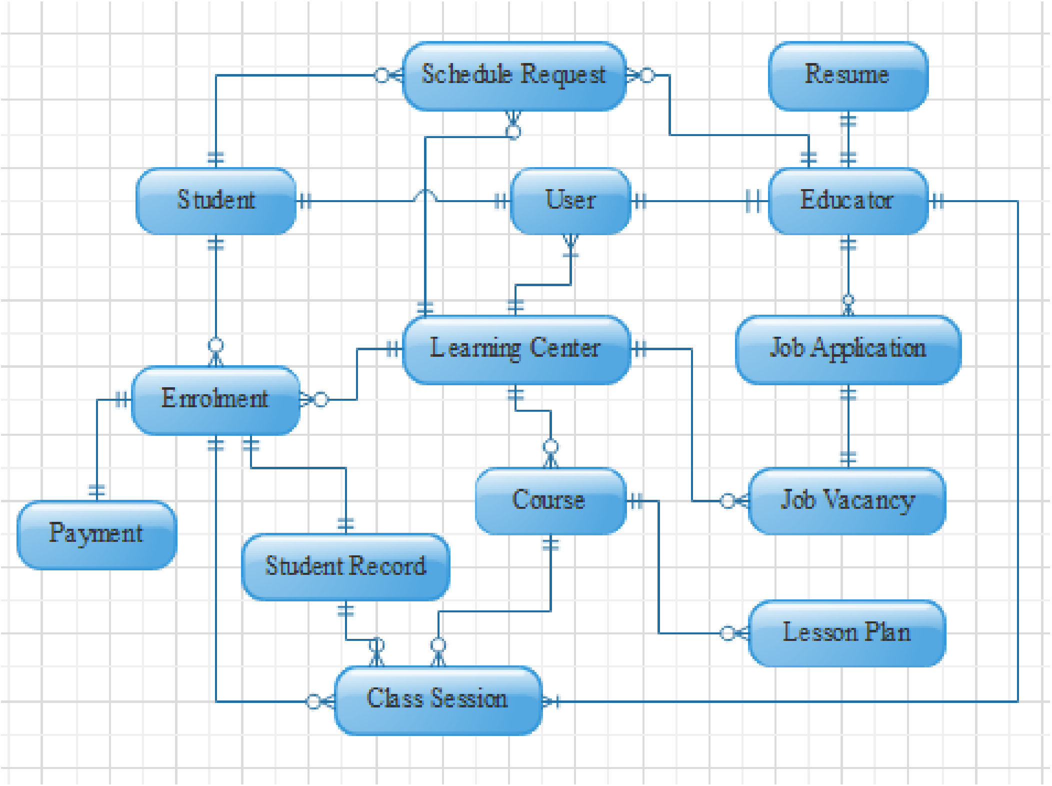


Figure 28: **Entity Relationship Diagram**

Figure 28 shows the entity-relationship diagram of iLearnCentral's database. The user is an entity that holds account management information used for login, password recovery, registration, and verification. Multiple user accounts are within a learning center with different access levels, while one user account per student and educator. The account management module handles user accounts.

The resume, job application, and job vacancy are document stores for profiling and hiring. Each educator is allowed to have one and only one resume. Meanwhile, learning centers can make multiple job vacancies for which educators can apply.

The enrolment module utilizes the course list and creates enrolment entries with payment instances. A single payment instance records the information for an enrolment's payment scheme and progress of installments.

The schedule request is the basis for scheduling classes. Class scheduling depends on the restrictions from students, educators, and learning centers. A student has classes from an enrolled course with many sessions assigned to one or different educators.

The teaching assistance involves the lesson plan and student record documents. The lesson plan segregates by course, while student records by enrolment.

### Data Dictionary

The data dictionary describes the types of data, properties and field sizes shown in the tables in the previous section. The tables below are data dictionaries for each table in the database.

Table 19

DATABASE DATA DICTIONARY

| Table | Key Name | Data Type | Field Size | Null | Description |
|---|---|---|---|---|---|
| User | username | varchar | 20 | not null | the name the user uses to login |
| User | password | varchar | 20 | not null | value used to verify the identity of a user |
| User | emailAddress | varchar | 30 | not null | valid email address for account verification |
| User | accountType | varchar | 20 | not null | determines the user account designation |
| User | securityQuestions | list |  | not null | array of security questions used for validating user identity |
| User | question | varchar | 50 | not null | single security question |
| User | answer | varchar | 20 | not null | answer to a security question |
| User | accountStatus | varchar | 20 | not null | the state of the user if they are validated, etc. |
| Learning Center | centerID | long | 20 | not null | primary key for learning center document |
| Learning Center | accounts | list |  | not null | array of user accounts in a learning center entry |
| Learning Center | username | varchar | 20 | not null | foreign key for the name user uses to login |

| | | | | | |
|---|---|---|---|---|---|
| Learning Center | accessLevel | varchar | 10 | not null | access levels to determine how a user can use the learning center's features |
| Learning Center | status | varchar | 10 | not null | status of a user account in learning center |
| Learning Center | businessName | varchar | 100 | not null | complete business name of a learning center |
| Learning Center | serviceType | varchar | 50 | not null | type of service provided by learning center |
| Learning Center | businessAddress | | | | address of business |
| Learning Center | buildingNo | varchar | 10 | null | building number part of the address |
| Learning Center | buildingName | varchar | 20 | null | building name part of the address |
| Learning Center | streetName | varchar | 20 | null | streetName part of the address |
| Learning Center | subdivision | varchar | 20 | null | subdivision part of the address |
| Learning Center | barangay | varchar | 20 | not null | barangay part of the address |
| Learning Center | district | varchar | 10 | null | district part of the address |
| Learning Center | city | varchar | 20 | not null | city part of the address |
| Learning Center | province | varchar | 20 | not null | province part of the address |
| Learning Center | country | varchar | 20 | not null | country part of the address |
| Learning Center | zipCode | int | 4 | not null | zip code part of the address |
| Learning Center | contactEmail | varchar | 30 | not null | official learning center email address |
| Learning Center | contactNumber | varchar | 20 | not null | contact numbers for learning center |
| Learning Center | companyWebsite | varchar | 30 | null | website to visit and learn more about learning center |
| Learning Center | operatingDays | list | | not null | days the learning center is open |
| Learning Center | openingTime | time | | not null | time the learning center opens |
| Learning Center | closingTime | time | | not null | time the learning center closes |
| Learning Center | coursesOffered | list | | | list of courses offered by the learning center |
| Learning Center | subscriptionType | varchar | 10 | not null | determines the current subscription |
| Learning Center | subscriptionEndDate | date | | null | date when a subscription ends and reverts to limited |

| | | | | | |
|---|---|---|---|---|---|
| Educator | educatorID | long | 20 | not null | primary key for educator |
| Educator | username | varchar | 20 | not null | foreign key for name of user used to log in |
| Educator | name | | | | name of educator |
| Educator | firstName | varchar | 50 | not null | first name of person |
| Educator | middleName | varchar | 20 | null | middle name of person |
| Educator | lastName | varchar | 20 | not null | last name of person |
| Educator | extension | varchar | 10 | null | extensions to name such as Sr., Jr., III, IV, etc. |
| Educator | birthday | date | | not null | birthdate of educator |
| Educator | address | list | | | addresses of an educator |
| Educator | houseNo | int | 10 | null | house number part of the address |
| Educator | streetName | varchar | 20 | null | street number part of the address |
| Educator | subdivision | varchar | 20 | null | subdivision part of the address |
| Educator | district | varchar | 10 | null | district part of the address |
| Educator | city | varchar | 20 | not null | city part of the address |
| Educator | province | varchar | 20 | not null | province part of the address |
| Educator | country | varchar | 20 | not null | country part of the address |
| Educator | zipCode | int | 4 | not null | zip code part of the address |
| Educator | currentAddress | boolean | | not null | tag determining if the address is the current one |
| Educator | email | varchar | 30 | not null | email address of educator |
| Educator | phoneNo | varchar | 15 | not null | contact no of educator |
| Educator | gender | char | 1 | not null | gender of educator (F, M) |
| Educator | maritalStatus | varchar | 10 | not null | marital status of an educator |
| Educator | religion | varchar | 30 | null | religion of the educator |
| Educator | citizenship | varchar | 30 | null | citizenship of the educator |
| Educator | employmentStatus | varchar | 10 | not null | status of employment in respect to learning centers in the system |
| Educator | centerID | int | 10 | null | foreign key for centerID employing this educator |
| Educator | position | varchar | 20 | null | position for employed educators in a learning center |
| Resume | resumeID | long | 20 | not null | primary key for resume document |
| Resume | educatorID | long | 20 | not null | foreign key to distinguish the owner of resume document |
| Resume | careerObjective | varchar | 500 | null | short description for career |

| | | | | | |
|---|---|---|---|---|---|
| | | | | | objectives in a resume |
| Resume | educationalHistory | list | | | list of educational history of an educator |
| Resume | educationLevel | varchar | 20 | not null | determines the level of education i.e. elementary, college |
| Resume | schoolName | varchar | 100 | not null | school name of previous education |
| Resume | schoolAddress | varchar | 300 | not null | address of the school |
| Resume | course | varchar | 100 | null | course taken |
| Resume | major | varchar | 50 | null | major taken during the course |
| Resume | yearStart | int | 4 | not null | starting year in this school |
| Resume | yearEnd | int | 4 | not null | ending year in this school |
| Resume | graduated | boolean | | not null | true if graduated, false if undergraduate |
| Resume | employmentHistory | list | | | list of employment history of an educator |
| Resume | companyName | varchar | 100 | not null | name of previous company |
| Resume | companyName | varchar | 300 | not null | address of previous company |
| Resume | position | varchar | 100 | not null | position or job description of previous company |
| Resume | dateStart | date | | not null | date started with previous employment |
| Resume | dateEnd | date | | not null | date ended with previous employment |
| Resume | skills | list | | null | list of skills in a resume |
| Resume | qualities | list | | null | list of qualities in a resume |
| Resume | interests | list | | null | list of interests in a resume |
| Resume | awards | list | | null | list of awards in a resume |
| Resume | references | list | | null | list of references for individual |
| Resume | referenceName | varchar | 50 | not null | name of reference |
| Resume | affiliation | varchar | 100 | not null | company of the reference |
| Resume | position | varchar | 50 | not null | position of the reference in their company |
| Resume | contactInfo | varchar | 30 | not null | contact information of the reference |
| Student | studentID | long | 20 | not null | primary key for the student document |
| Student | username | varchar | 20 | not null | foreign key for name of user used to log in |
| Student | name | | | | name of student |
| Student | firstName | varchar | 50 | not null | first name of person |
| Student | middleName | varchar | 20 | null | middle name of person |

| | | | | | |
|---|---|---|---|---|---|
| Student | lastName | varchar | 20 | not null | last name of person |
| Student | extension | varchar | 10 | null | extensions to name such as Sr., Jr., III, IV, etc. |
| Student | birthday | date | | not null | birthdate of educator |
| Student | address | list | | | addresses of an educator |
| Student | houseNo | int | 10 | null | house number part of the address |
| Student | streetName | varchar | 20 | null | street number part of the address |
| Student | subdivision | varchar | 20 | null | subdivision part of the address |
| Student | district | varchar | 10 | null | district part of the address |
| Student | city | varchar | 20 | not null | city part of the address |
| Student | province | varchar | 20 | not null | province part of the address |
| Student | country | varchar | 20 | not null | country part of the address |
| Student | zipCode | int | 4 | not null | zip code part of the address |
| Student | currentAddress | boolean | | not null | tag determining if the address is the current one |
| Student | email | varchar | 30 | not null | email address of educator |
| Student | phoneNo | varchar | 15 | not null | contact no of educator |
| Student | gender | char | 1 | not null | gender of educator (F, M) |
| Student | maritalStatus | varchar | 10 | not null | marital status of an educator |
| Student | religion | varchar | 30 | null | religion of the educator |
| Student | citizenship | varchar | 30 | null | citizenship of the educator |
| Student | centerID | int | 10 | null | centerID for the current learning center enrolled in |
| Student | enrolmentStatus | varchar | 10 | null | status of enrolment |
| Student | enrolmentHistory | list | | | list of enrolmentIDs, foreign key, of instances of enrolment made by the student |
| Job vacancy | vacancyID | long | 20 | not null | primary key for job vacancy entries |
| Job vacancy | centerID | long | 20 | not null | foreign key for Learning center creator of job vacancy |
| Job vacancy | status | varchar | 10 | not null | status of the job vacancy i.e. active, cancelled, filled |
| Job vacancy | position | varchar | 30 | not null | position to be filled |
| Job vacancy | jobDescription | varchar | 400 | null | description of the job position |
| Job vacancy | jobType | varchar | 25 | not null | type of job i.e. full-time, part-time, full-time or part-time |

| | | | | | |
|---|---|---|---|---|---|
| Job vacancy | educationalRequirements | list | | | requirements based on educational attainment |
| Job vacancy | educationalLevel | varchar | 200 | not null | educational attainment needed i.e. high school graduate, college level |
| Job vacancy | degrees | list | | null | degrees earn from school i.e. bachelor of Secondary Education |
| Job vacancy | majors | list | | null | major taken during from the degrees |
| Job vacancy | minimunUnits | int | | null | minimum number of units required |
| Job vacancy | qualifications | list | | not null | list of qualifications needed |
| Job vacancy | skills | list | | null | list of skills needed |
| Job vacancy | incentives | list | | null | list possible incentives to entice applicants |
| Job vacancy | responsibilities | list | | null | list of possible responsibilities |
| Job vacancy | requirements | list | | null | list of what requirements applicants need to give |
| Job vacancy | applicationMethod | varchar | | null | list of ways to apply |
| Jobapplication | jobApplicationID | long | 20 | not null | primary key for job application |
| Jobapplication | educatorID | long | 20 | not null | foreign key to the educator making the job application |
| Jobapplication | vacancyID | long | 20 | not null | foreign key for the vacancy applied for |
| Jobapplication | applicationDate | date | | not null | date the job was applied to |
| Jobapplication | applicationStatus | varchar | | not null | status of the application i.e. pending, accepted, rejected |
| Jobapplication | preferredMethod | varchar | | null | what way the application was done i.e. walk-in |
| Jobapplication | message | varchar | 10000 | null | optional message to the learning center |
| Course | courseID | long | 20 | not null | primary key for the course |
| Course | centerID | long | 20 | not null | foreign key for the center offering the course |
| Course | courseName | varchar | 100 | not null | name of course or class offered |
| Course | courseDescription | varchar | 500 | not null | description of the course or class offered |
| Course | courseType | varchar | 30 | null | if any, the course type |
| Course | tuition | float | | null | tuition amount for a course |
| Enrolment | enrolmentID | long | 20 | not null | primary key for enrolment |
| Enrolment | centerID | long | 20 | not null | foreign key to which center |

| | | | | | |
|---|---|---|---|---|---|
| Enrolment | studentID | long | 20 | not null | foreign key to which student |
| Enrolment | courseID | long | 20 | not null | foreign key to course enrolled |
| Enrolment | enrolmentDate | date | | not null | date enrolment occurred |
| Enrolment | dateClassStart | date | | null | date for start of classes |
| Enrolment | dateClassEnds | date | | null | date for end of classes |
| Enrolment | noOfHours | int | 5 | null | number of hours for the course |
| Enrolment | enrolmentStatus | varchar | 20 | not null | status of the enrolment |
| Payment | paymentID | long | 20 | not null | primary key for payment |
| Payment | enrolmentID | long | 20 | not null | foreign key for enrolment |
| Payment | tuition | float | | not null | tuition amount |
| Payment | additionalFees | float | | null | amount of additional fees |
| Payment | balance | float | | not null | current balance |
| Payment | payments | list | | null | list of payments made |
| Payment | paymentDate | date | | not null | date a payment is made |
| Payment | amount | float | | not null | amount, partial or full for payment |
| Payment | paymentMethod | varchar | 40 | not null | method the payment is made |
| Payment | validated | boolean | | not null | validation for payment |
| Payment | status | varchar | 20 | not null | status of payment, complete or with balance |
| Class session | classSessionID | long | 20 | not null | primary key for the class instance |
| Class session | enrolmentID | long | 20 | not null | foreign key of the enrolment basis of the class |
| Class session | educatorID | long | 20 | null | educator assigned to the class |
| Class session | date | date | | not null | the date of the class |
| Class session | timeStart | time | | not null | the time the class start |
| Class session | timeEnd | time | | not null | the time it should end |
| Class session | roomNo | varchar | | not null | the room number assigned to the class |
| ScheduleRequest | scheduleRequestID | long | | not null | primary key for schedule restriction request |
| ScheduleRequest | restrictionType | varchar | 50 | not null | type of restriction i.e. available, restricted |
| ScheduleRequest | educatorID | long | 20 | null | foreign key to educator requesting the schedule |
| ScheduleRequest | studentID | long | 20 | null | foreign key to student requesting the schedule |
| ScheduleRequest | centerID | long | 20 | not null | foreign key to learning center requested to |

| | | | | | |
|---|---|---|---|---|---|
| ScheduleRequest | classSessionID | long | 20 | null | foreign key for the specific class if necessary |
| ScheduleRequest | scheduleDate | date | | not null | the date a schedule is requested |
| ScheduleRequest | timeStart | time | | null | a possible start time of schedule request |
| ScheduleRequest | timeEnd | time | | null | a possible end time of schedule request |
| Lesson Plan | lessonID | long | 20 | not null | primary key for lesson plan |
| Lesson Plan | courseID | long | 20 | not null | foreign key for learning center |
| Lesson Plan | topic | varchar | 100 | not null | topic of the lesson plan |
| Lesson Plan | overview | varchar | 500 | null | short description of the topic to plan for |
| Lesson Plan | objective | list | | null | a list of objectives for the lesson plan |
| Lesson Plan | materials | list | | null | a list of materials for the lesson plan |
| Lesson Plan | activities | list | | null | a list of activities for the lesson plan |
| Lesson Plan | procedures | list | | null | a list of procedures for the lesson plan |
| StudentRecord | studentRecordID | long | 20 | not null | primary key for student record |
| StudentRecord | enrolmentID | long | 20 | not null | foreign key to enrolment id associated with record |
| StudentRecord | lessonPlanCovered | list | | null | the lesson plans tackled by the student |
| StudentRecord | classSession | list | | null | the class sessions the student showed up to |
| StudentRecord | classSessionID | long | 20 | not null | foreign key to the class session ID |
| StudentRecord | remarks | varchar | 500 | null | remarks an educator can give to the student |
| StudentRecord | studentScores | list | | null | possible scores for student when tests occur |
| StudentRecord | testTitle | varchar | 200 | not null | title for the test |
| StudentRecord | score | int | 20 | not null | score the student got |
| StudentRecord | perfectscore | int | 20 | not null | perfect score the student can get |
| StudentRecord | testDate | date | | not null | date the score was taken |

Table 19 displays the data dictionary of all documents in the database. It contains the description for each detail in the records. For some NoSQL servers, the Varchar data type may be String. To find the primary and foreign keys refer to the database design section.

### Network Model

The model of the network shows how the system components communicate via the internet. The diagram shows that the user is able to check and monitor their account through application for possible breaches or errors.

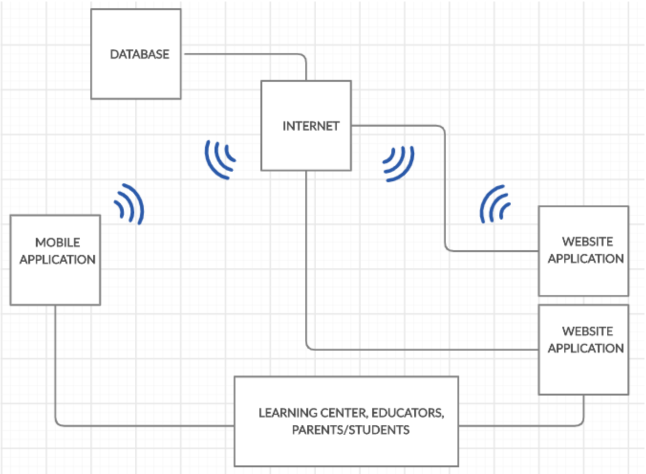


Figure 29: **Network Model**

Figure 29 shows the network model of the system. Internet is used for both web and mobile app to interact with the database.

### Network Topology

The network topology illustrates how the system's component work in conjunction with the use of internet connection to access the user's access database.

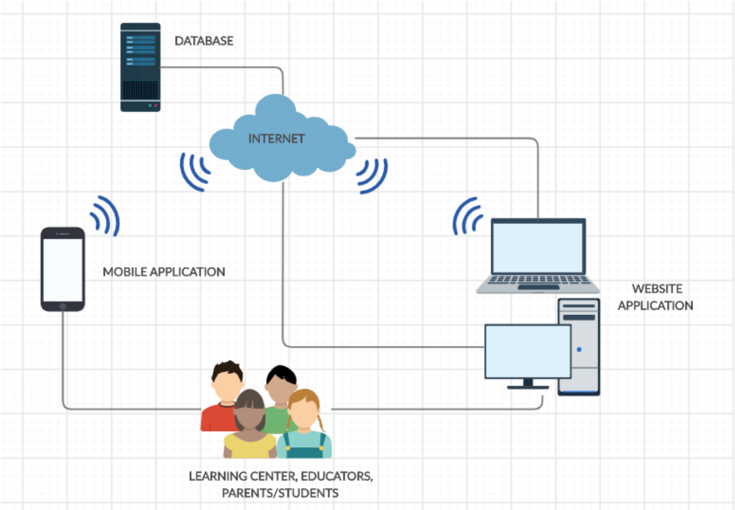


Figure 30: **Network Topology**

Figure 30 shows the network topology of the system. As shown the user can use both web and mobile app with the help of the internet. They can manage classes, check schedules, post and search jobs, etc.

### Development/Construction/Build Phase

The Development Phase marks the end of the initial process segment and marks the beginning of development. This phase is intended to turn the prototyped system design in the Design Phase into a working system that meets all defined system requirements. Two elements are required to complete this phase successfully: 1) a complete set of design specifications and 2) proper processes, standards and tools.

### Technology Stack Diagram

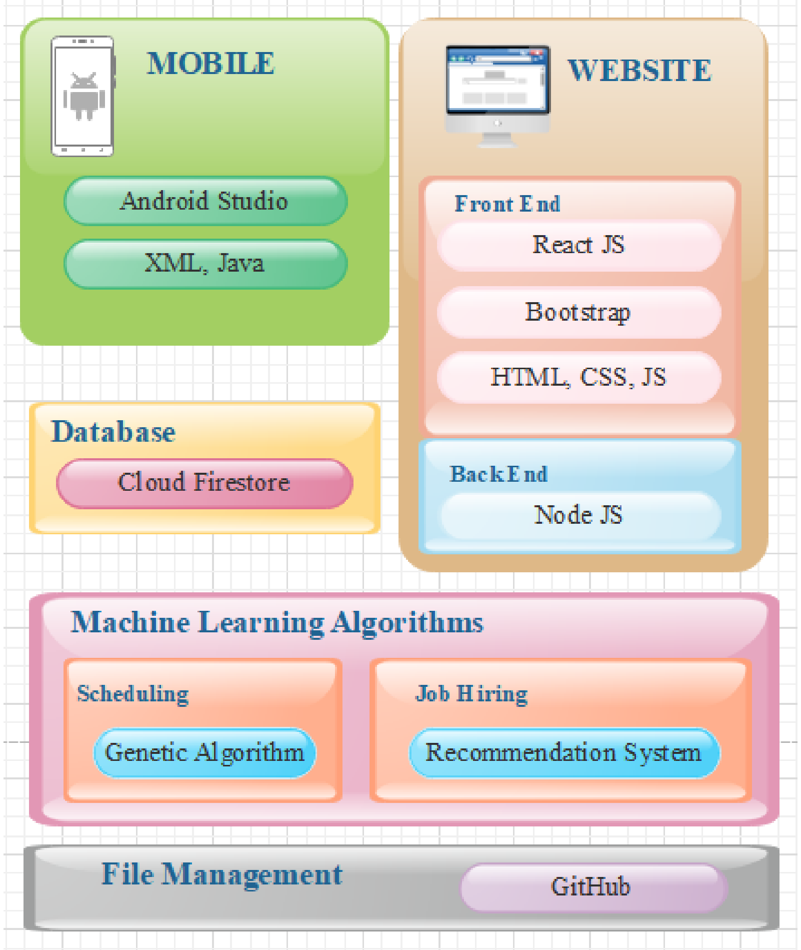


Figure 31: **Technology Stack Diagram**

Figure 31 shows the technology stack diagram representing the different technologies the project uses and the purpose for each specific language.

**Android Studio** is an integrated development environment for the Android operating system. It was built on JetBrains' IntelliJ IDEA software and designed for android development. It comprises both frontend and backend development by using XML and java.

**XML**, meaning eXtensible Markup Language, is a markup language built as a standard way to encode data in internet-based applications. Android uses it in creating layouts and components as Front End for typical applications.

**Java** is one of the languages used in android development. Java's mobile version is called Java ME. Many smartphones and tablets support it. The Java Platform Micro Edition (Java ME) provides a flexible, secure environment for building and running applications that target embedded and mobile devices. Java ME addresses the challenge of running applications on devices that are low on memory, display, and power available.

**ReactJs** is a JavaScript library that is declarative, powerful and scalable to create user interfaces. This allows complex UIs to be constructed from small and isolated pieces of code called "components."

**Bootstrap** is a free and open-source front end development platform for website and web app construction. The architecture for Bootstrap is based on HTML, CSS, and JavaScript (JS) to promote the development of responsive, first mobile sites and apps.

**HTML**, or HyperText Markup Language, is the standard markup language for creating Web pages. It describes the structure of a Web page. Consisting of a series of elements or tags, it tells the browser how to display content.

**CSS**, short for Cascading Style Sheets, a new feature introduced to HTML that provides more control over how pages present to both website developers and users.

**JavaScript** is a scripting language on the client-side. It means that the web browser of the client interprets the source code instead of the webserver. JavaScript functions can run without interacting with the server after a web page loads.

**Node.js** (Node) is an open-source framework for the server-side execution of JavaScript code. Node is useful for developing applications requiring a persistent connection between the browser and the server and is often used for real-time applications such as chat, news feeds, and web push notifications.

**Cloud Firestore** is a repository of NoSQL documents designed for automatic scaling, high performance, and ease of application development.

**Genetic Algorithm** is a search heuristic based on Charles Darwin's theory of natural evolution. The algorithm reflects the natural selection process in which the most suitable individuals are selected for reproduction to produce the next-generation offspring. It consists of five phases–initial population, fitness function, selection, crossover, and mutation.

**Recommendation system** is a group of machine learning algorithms that strives to predict user preferences and make suggestions that clients would be interested in. It has two approaches to making recommendations–collaborative filtering and content filtering. Collaborative filtering involves comparing the behavior of similar groups to predict what a user, with likely behaviors, would want. Meanwhile, content filtering is based on a description of the item and a profile of the user's preferences.

**GitHub** is a system used to store a project's source code and record any modifications to that code in its entire history. It allows developers to work more efficiently on a project by providing resources from different developers to manage potentially conflicting changes.

## Software Specification

The software specification describes the functional requirements of the study. It includes the programming language, platform for development, management of the database, and machine learning algorithms.

The mobile development uses Android Studio IDE with Java being the back end programming language, and XML for front end builds. The mobile application is for Android devices. The development uses minimum API Level 21 to run with devices Android 5.0 Lollipop and higher. The researchers decided with the minimum API based on the worldwide Android version distribution, according to Holst (2019) and Protalinski (2019), where roughly 90% of devices running in Android have versions 5.0 and higher.

The website or web app development involves the use of Node Js for backend support. For the frontend, React js is the primary framework to create the components with Bootstrap providing design beatification. HTML, CSS, and Javascript are additional languages to use when necessitated. Sublime IDE is the preferred environment used to code the web app. However, collaborators may use other applications suitable for the purpose.

Genetic algorithm is the preferred machine learning algorithm to use for scheduling classes. Making of class schedules are NP-hard problems and does not have a definite correct answer, only an optimal one. The heuristic approach is usually enough for simple cases but with the complexity of the system. It is decided to go with a Genetic Algorithm for a better solution.

The hiring module makes use of Recommendation systems to efficiently suggest a list of qualified job seekers to a learning center with job vacancies and a list of job vacancies to a job seeker. Content-filtering is the initial approach to the small dataset until such time when collaborative filtering can add to the efficiency of the recommendations.

Cloud Firestore is the database of choice to support the project. Both mobile and web application connects to Firestore for all data. GitHub supports the collaboration of the members and allows them to code concurrently for more efficient and time-conscious development.

## Program Specifications

Program specifications contain the list of algorithms needed for the system.

Table 20

SOFTWARE LIST OF MODULES

| **Programmer/s** | **Modules** | **Learning Center** | **Educator** | **Parent or Student** |
|---|---|---|---|---|
| Jephunneh<br>Rhea Shane<br>Cristian | **Account Management** | | | |
| | 1. Registration | * | * | * |
| | 2. Authentication | * | * | * |
| | 3. Login | * | * | * |
| | 4. Profiling | * | * | |
| | No. of Points *(1 point per module per user)* | 1 | 1 | 1 |
| Jephunneh<br>Rhea Shane<br>Cristian | **Hiring Module** | | | |
| | 1. Hiring Profile/Resume | | * | |
| | 2. Job Searching | | * | |
| | 3. Job Posting | * | | |
| | 4. Job Suggestion | | * | |
| | 5. Hire Suggestion | * | | |
| | 6. Hiring | * | | |
| | No. of Points *(1 point per module per user)* | 1 | 1 | 0 |
| Jephunneh | **Enrollment Module** | | | |

| | | | | |
|---|---|---|---|---|
| Rhea Shane<br>Cristian | 1. Input/Add Course Details | * | | |
| | 2. Search/Display Course List | * | * | * |
| | 3. Course Selection | | | * |
| | 4. Fee Calculation | | | * |
| | 5. Enrolment Details and Processes | | | * |
| | 6. Payment Scheme Selection | | | * |
| | 7. Payment | | | * |
| | 8. Record Payment | * | | * |
| No. of Points *(1 point per module per user)* | | 1 | 1 | 1 |
| Jephunneh<br>Rhea Shane<br>Cristian | **Scheduling Module** | | | |
| | 1. Input Class Details | * | | |
| | 2. Update Class Details | * | | |
| | 3. Input Schedules | * | | |
| | 4. Schedule Request | | * | * |
| | 5. Update Schedules | * | | |
| | 6. Generate Calendar of Activities | * | * | * |
| | 7. Notification of Changes | * | * | * |
| No. of Points *(1 point per module per user)* | | 1 | 1 | 1 |
| Jephunneh<br>Rhea Shane<br>Cristian | **Teaching Assistance Module** | | | |
| | 1. Retrieve Class Details | | * | |
| | 2. Suggest Daily Lesson Plan | | * | |
| | 3. Keep Student Records | | * | |
| | 4. Track Student Progress | | * | |
| No. of Points *(1 point per module per user)* | | 0 | 1 | 0 |
| Number of Modules per User *(equals total no. of points per user)* | | 4 | 5 | 3 |
| Total Number of Modules | | 12 | | |

Table 20 shows the comparison of the access level of each type of account. The table shows that multiple types of accounts or a specific type of account can access a module. It also shows the programmer/s assigned to develop per module.