Design Document

for

Student Attention Detector and Notifier

Project 10

March 15, 2018

Prepared by Group 6 –

Archit Jugran (160101087)

Shubham Goel (160101083)

Yagyansh Bhatia (160101079)

Supervised by

Prof. Samit Bhattacharya

Index

Page #

1. Introduction

This Software Design Document is a document which will be used for implementation in a programming language. Within this Software Design Document are narrative and graphical documentation of the software design for the project including Data Flow Diagram(DFD) model, Use case and Entity Relationship(ER diagram).

1.1 Purpose

Software design document is a written description of a software product, that a software designer writes in order to give a software development team overall guidance to the architecture of the software project. The purpose of the project is to allow the teacher to know the names of students who may not be attentive in class, for a better class environment and for the benefit of the students themselves.

1.2 Intended Audience

The intended audience of this document is our overseeing project professor, Dr. Samit Bhattacharya and our teaching assistants , Sir Ujjwal Biswas, Sir Md Shakeel Iqbal Saikia and Sir Subrata Tikadar.

1.3 Scope of Product

The product aims at making the classroom environment better by sending warnings in the form of notifications to un-attentive students as well as sending notifications to the teacher about the students, hence motivating the students to stay attentive in class for their own benefit. The system is based on a database on a server which accepts, processes and pushes data from/to Android devices. A comfortable user-experience will also be built.

1.4 References

1.Tools for creating DFD diagrams and USE case model and ER Diagram:

a)SmartDraw version 2013

b)Microsoft Power-Point Version 2016

2.Other Source for reference :

a).http://nptel.ac.in/downloads/106105087/

b).Software Engineering By- Roger S. Pressman

3. Android Studio: https://developer.android.com/training/index.html

1.5 Document Conventions

Term	Definition
Teacher / Professor	Person who shall be using the software for monitoring
Student	Person who shall be monitored by the instructor.
User	Collectively refers to the students and teachers.
Google Firebase	Firebase is a mobile and web application development platform which can be used as a backend for a mobile application to store data and send notifications.
Android	A mobile operating system developed by Google.
Node.JS	A web application framework.
Javascript	A high-level, interpreted programming language for the front-end of a system

2. Design Overview

This section gives a brief overview of design for the reader of the document to understand the design better and see a summary before proceeding to view this document details.

2.1 Background

This application will be an extension of the already existing system which can calculate the attention levels of all students seated in a classroom during a lecture . The responsibility of my application will be to send notifications to the students and teacher (without disturbing the teacher during the lecture , for example, by saving teacher's notifications silently in a log which can be viewed later) to warn them about not being attentive in class or motivate them for being more attentive in class.

2.2 System Architecture

The application use case environment consists of a Professor and students attending the lecture. We are building the application under the assumption that multiple lectures are being held in different rooms and the application can function in that situation . A server is created by the professor and the students connect to server using their Android devices by logging in and entering the room number . The server uses the cloud database for relaying data across the professor and student devices. Whenever the state of a student i.e. his attention level changes, the data (state) is updated in the database and a "Event Listener" which is attached to the database gets triggered. In case of a change in attention, the student is notified immediately using a silent notification but for the teacher, all notifications get saved in a log which can be accessed by the teacher at any moment of time . However , if more than 50 % of the students' attentions fall down, the teacher is notified using a vibrating notification immediately.

2.3 System Interfaces and Implementing Technologies

2.3.1 User Interfaces :

Using the user interface, the student and teacher can login/register and connect to the server . Afterwards, the student will receive notifications and the teacher can view all notifications sent to different students.

2.3.2 Software and Communication Interfaces :

The state change data collected will be stored in a realtime Firebase database. Node.js will be used to write a Google Cloud function which will get triggered whenever there is a change in the database

and will send notifications to the student device and save them in the log of teacher device. Android Studio will be used to write the basic code for the application which will be connected to Firebase and the Google Cloud Function.

2.4 Assumptions and Dependencies

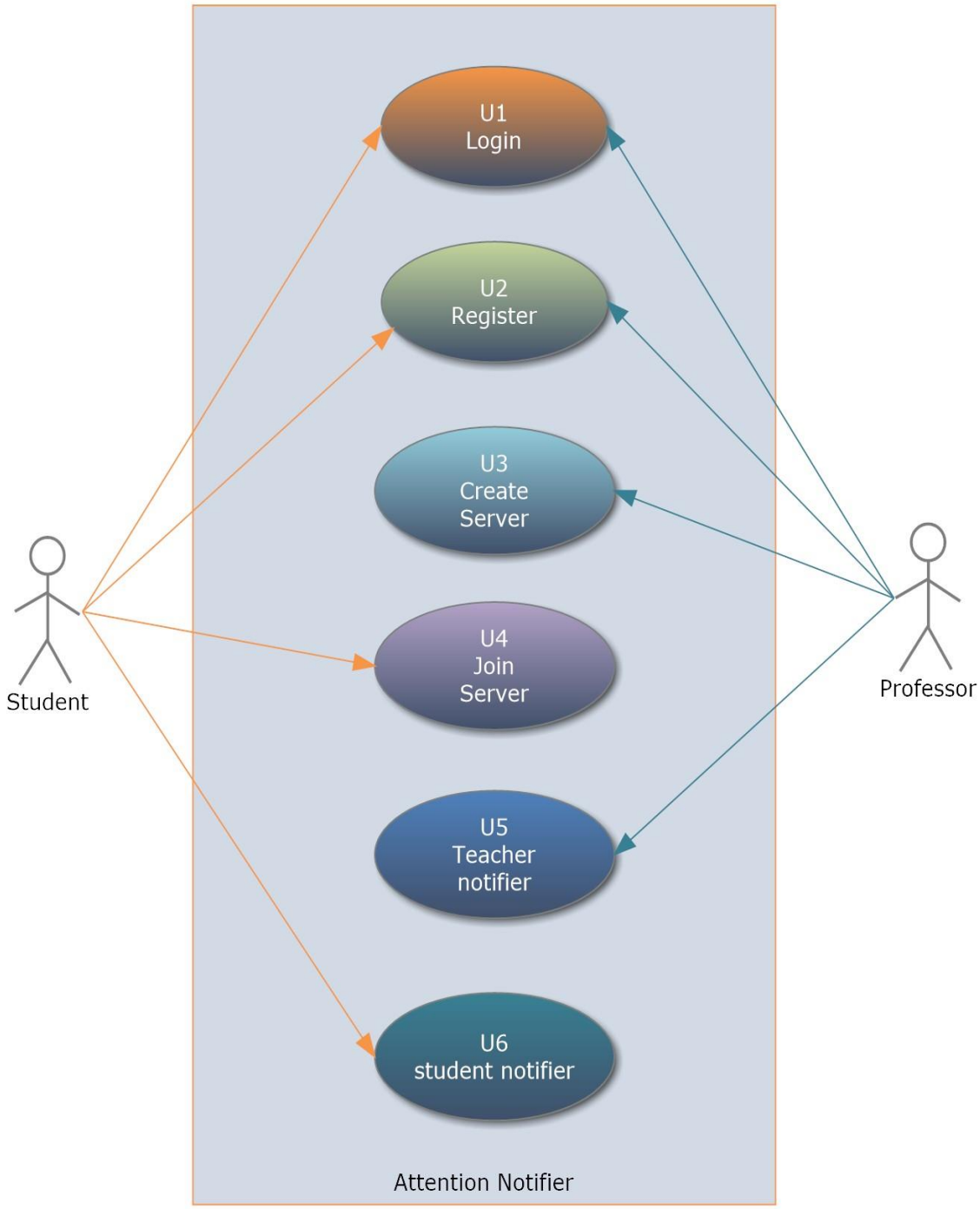
We are assuming that the application is being used on an Android device meeting the requirements :

1. Operating system used : Android v4.0 / API Level 15 or higher
2. Wi-Fi with internet connection
3. Google Firebase server properly set up.

If any of the requirements or hardware resources are not met, then the application may not work properly as intended or even may not work at all. Hence the mobile phones must have sufficient resources not occupied i.e. not allocated to other processes.

3. Use Cases

Use Case Diagram: Attention Notifier



3.1 U1 : Login

Actors : Student, Professor (both are collectively referred to as "User")
a) **Scenario 1** : Mainline sequence
1. *User* : Select Login option
2. *System* : Prompt to enter login details
3. *User* : Enter login details i.e. username and password
4. *System* : Display login acknowledgement

b) **Scenario** 2: At step 4 of mainline sequence
4. *System* : Display error in case of wrong login details

3.2 U2 : Register

Actors : Student, Professor (both are collectively referred to as "User")
a) **Scenario 1** : Mainline sequence
1. *User* : Select Register option
2. *System* : Prompt to enter username and password
3. *User* : Enter username and password and click on "Next" button
4. *System* : Prompt to enter full name and position
5. *User* : Enter full name and position and click on "Submit" button
4. *System* : Displays "Successfully registered"

b) **Scenario 2** : At step 4 of mainline sequence
4. *System* : Prompts user to choose different username if the username is already registered by a different user.

3.3 U3 : Create Server

Actors : Teacher
Precondition : Teacher has logged in
a) **Scenario 1** : Mainline Sequence
1. *System* : Display prompt to enter room number
2. *Teacher* : Enter room number
3. *System* : Displays message that room number successfully registered

3.4 U4 : Join server

Actors : Student
Precondition : Student has logged in
a) **Scenario 1** : Mainline sequence
1. *System* : Display prompt to enter room number to join
2. *Student* : Enter room number
3. *System* : Displays message that room number successfully joined

b)**Scenario 2** : At step 3 of mainline sequence
3. *System* : Displays error if room number not created by some teacher before.

3.5 U5 : Teacher Notifier

Actors : Teacher
Precondition : Teacher has logged in and created server and some students have joined the server
a)**Scenario 1** : Mainline sequence
1. *Teacher* : Select "Display log" option
2. *System* : Displays log of messages silently

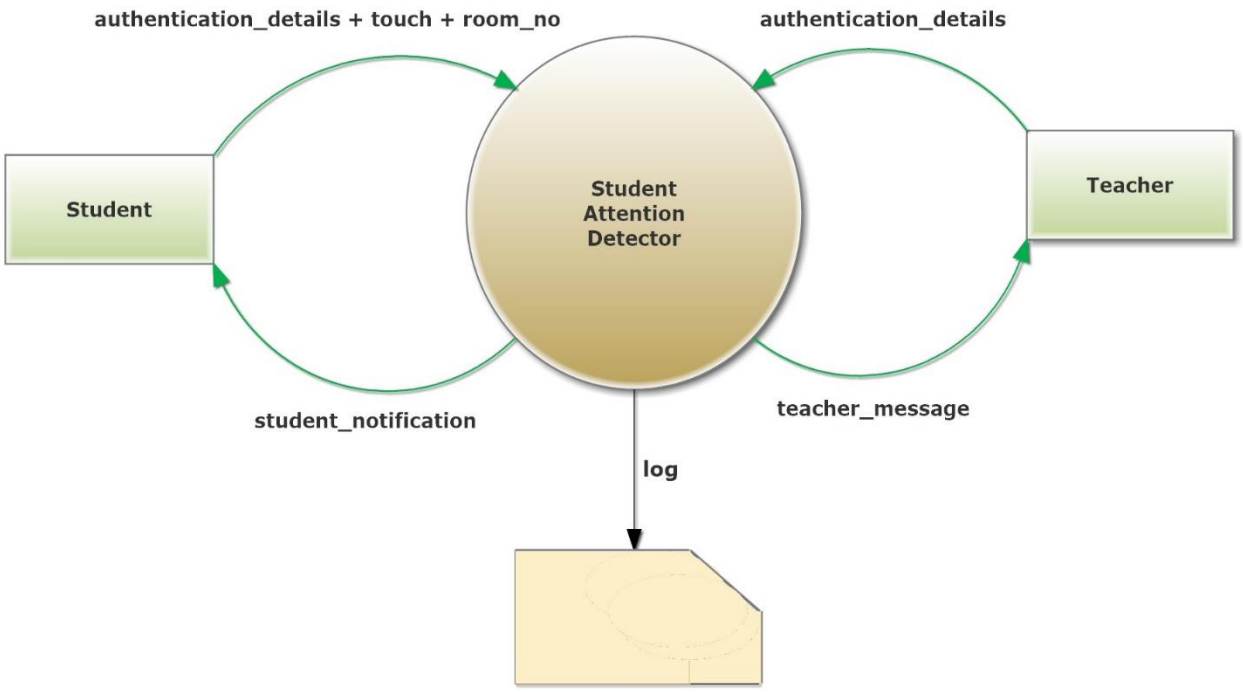
b) **Scenario 2** : Alternate sequence on step 1
1. *System* : Vibrate and display prompt that more than half of class attention decreased

3.6 U6 : Student Notifier

Actors : Student
Precondition : Student has logged in and joined a server created by some teacher
a)**Scenario 1** : Mainline sequence
1. *System* : Display notification to student device about attention level change
2. *Student* : Touch the notification
3. *System* : Prompt that notification confirmed.

4. Data Flow Diagram

4.1 Level 0 DFD or Context Diagram



BLUE ARROW (→) : For data flow within the same level of DFD

GREEN ARROW (→): For data flow to other level of DFD

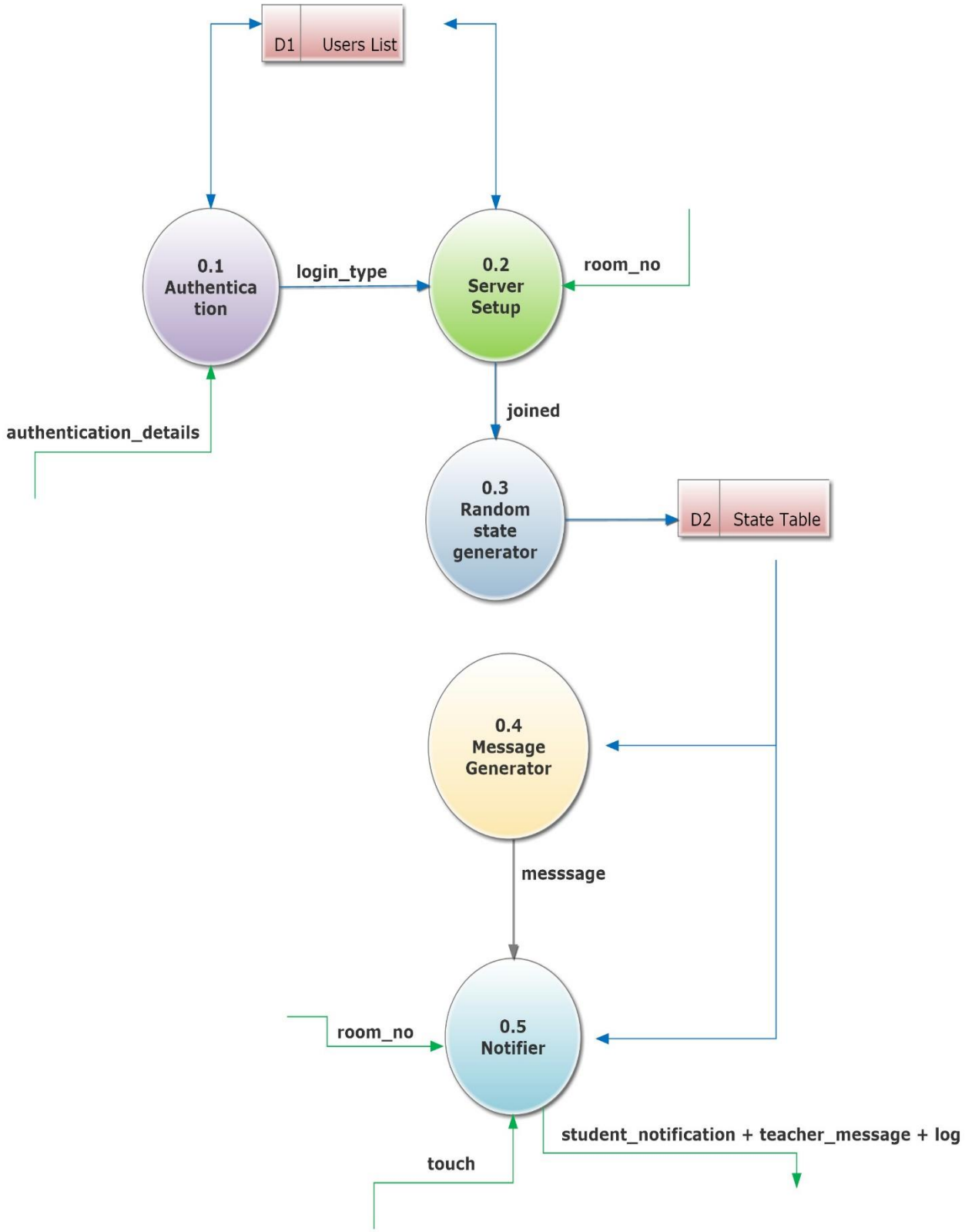
NOTE : Use the DATA DICTIONARY whenever have any confusion about flow of any data

4.1.1 Data Dictionary for Level O

authentication_details	[login_details , register_details] /* Enter either login details or registration details */
login_details	username + password /* Login credentials for already registered user */
register_details	username + password + name + position /* Enter first time registration data */
room_no	Integer

	/* Room number in which teacher is teaching or student is studying */
username	String /* Preffered Username of user*/
password	String /* Password for user account */
position	[“Teacher”,”Student”] /* Is the user a teacher or a student */
name	String /* Full name of the user */
touch	Control Command /*command to confirm notification */
student_notification	room_no+username+[“Keep up the good level of attention!”, “Please pay more attention in class “] /* The notification that is to be sent to a student whose state is changing from one range to another */
confirm	Boolean /* True if student has confirmed notification */
log	{ student_notification + confirm }* /* Zero or more notifications that were sent to student are contained in the log saved on teacher’s phone*/
teacher_message	(notification) /* the message sent to teacher consists of a notification if 50% of class attention falls as well as an indication of whether the student has confirmed notification or not */
notification	alert + vibrate /* notification will come with vibration */
vibrate	Control command /* Command to vibrate the phone */
alert	“More than 50 % of the students attention is going down”

4.2 Level 1 DFD

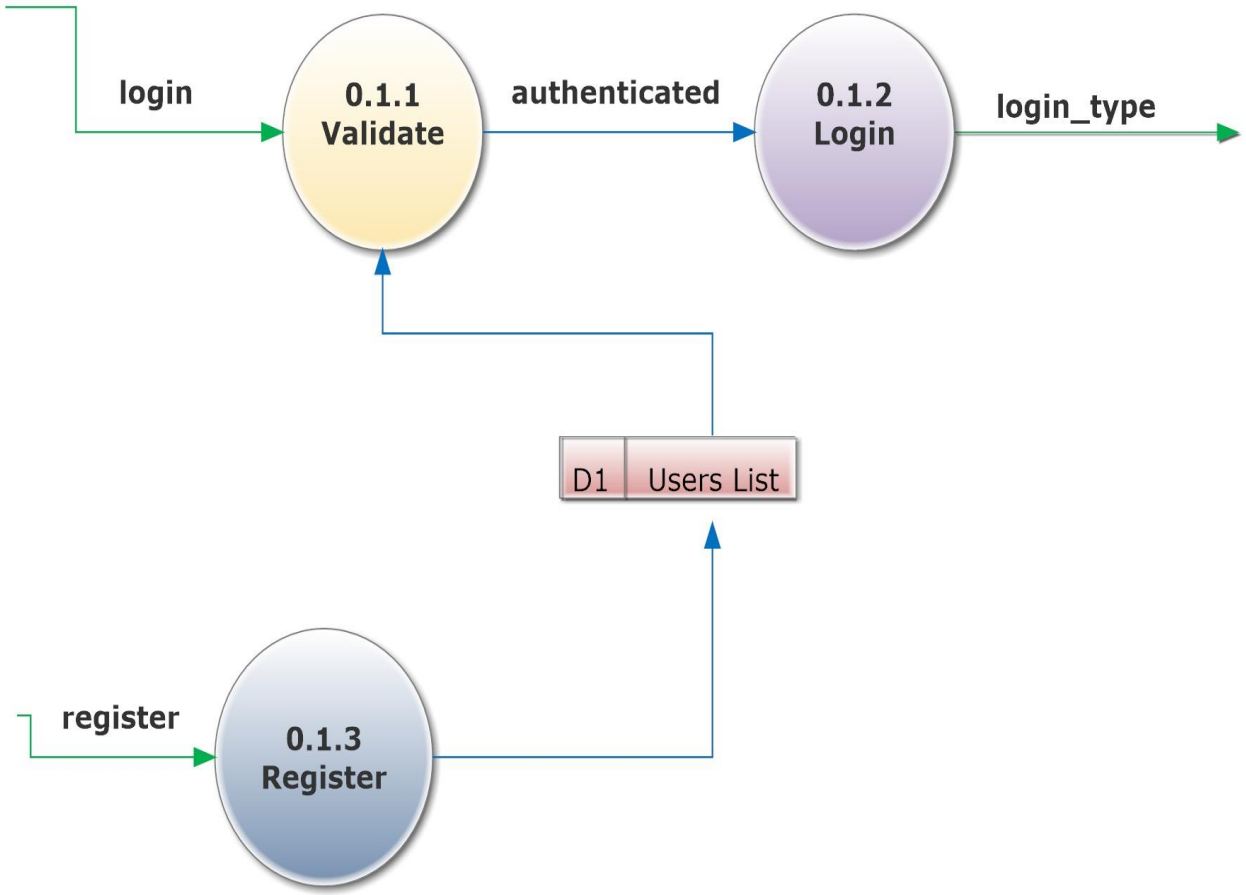


4.2.1 Additional Data Dictionary for Level 1

login_type	[“Teacher”,”Student”] /* Tells the type of login */
joined	[teacher_join , student_join] /* Indicates whether current user has joined server or not*/
teacher_join	Boolean /* True if teacher has joined server */
student_join	Boolean /* True if student has joined server */
message	room_no+username+[“Keep up the good level of attention!”, “Please pay more attention in class “] /* The notification that is to be sent to a student whose state is changing from one range to another */

4.3 Level 2 DFD s

4.3.1 Authentication



Func 0.1.1 : Validate – Validates the login credentials entered by user from the users' list database D1 and returns true if user validated, else false.

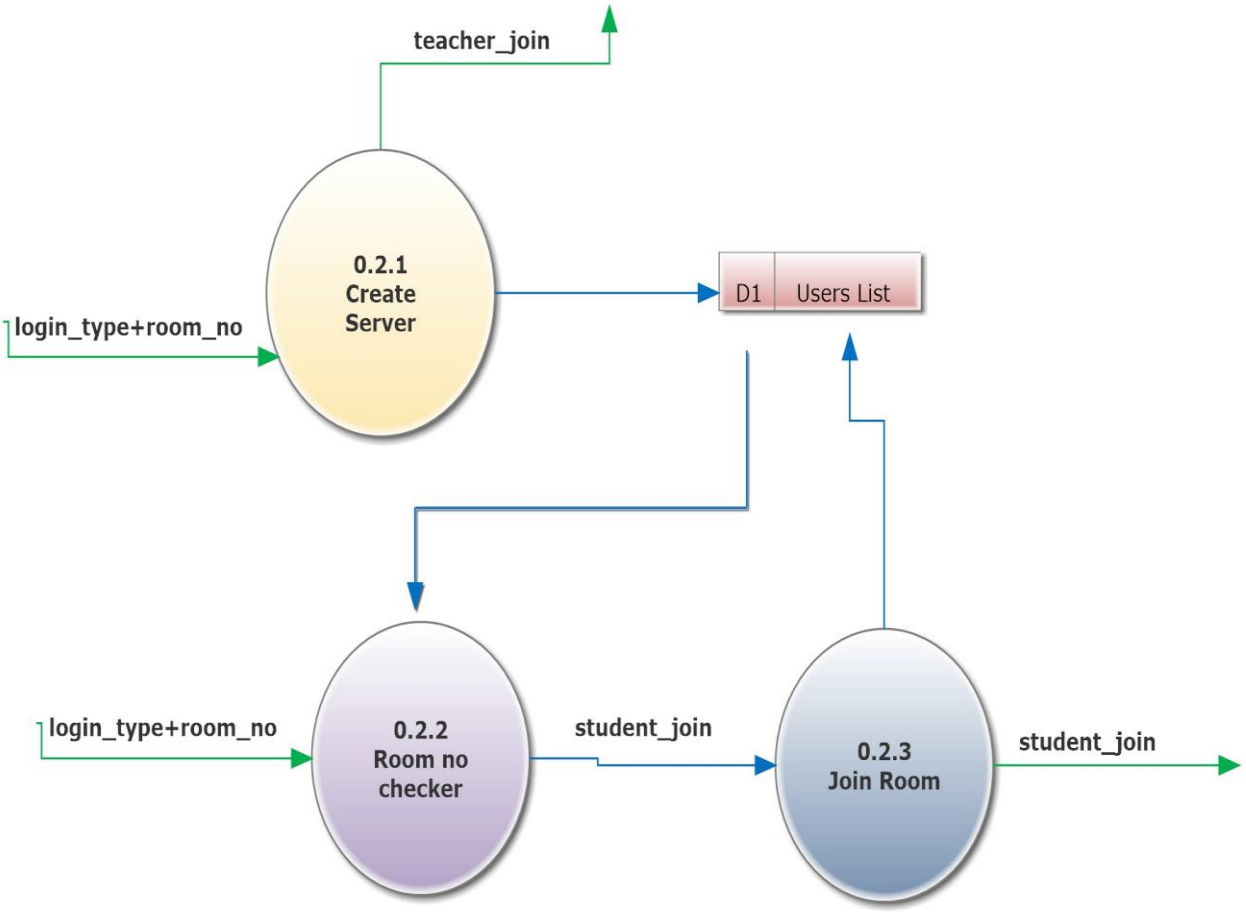
Func 0.1.2 : Login – Log in the user if the variable authenticated is true

Func 0.1.3 : Register – Registers the user and adds the entered details of user into the users' list database D1.

Additional data dictionary:

authenticated	Boolean /* True if user has been validated otherwise false */

4.3.2 Server setup

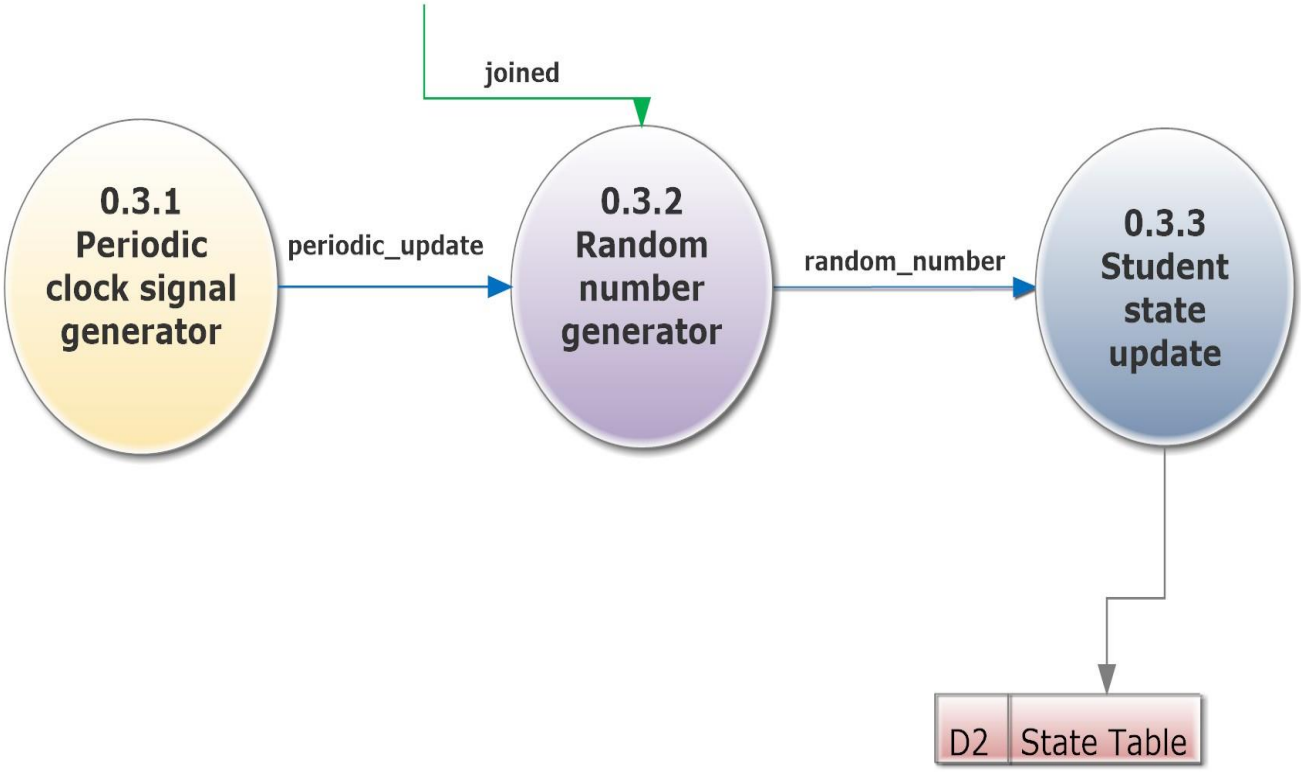


Func 0.2.1 : Create server -- Checks if login type is "teacher", then creates a server by registering the room number in the users' list database D1 . If login type is "student" , the process does nothing.

Func 0.2.2 : Room no. Checker – This process executes only if login type is "student" and checks if the room number entered by student has been registered by a teacher or not by accessing data from the Users' List database D1. If not available, the student will reenter the room number.

Func 0.2.3 : Join room – Makes the student join the server corresponding to the entered room number.

4.3.3 Random state generator

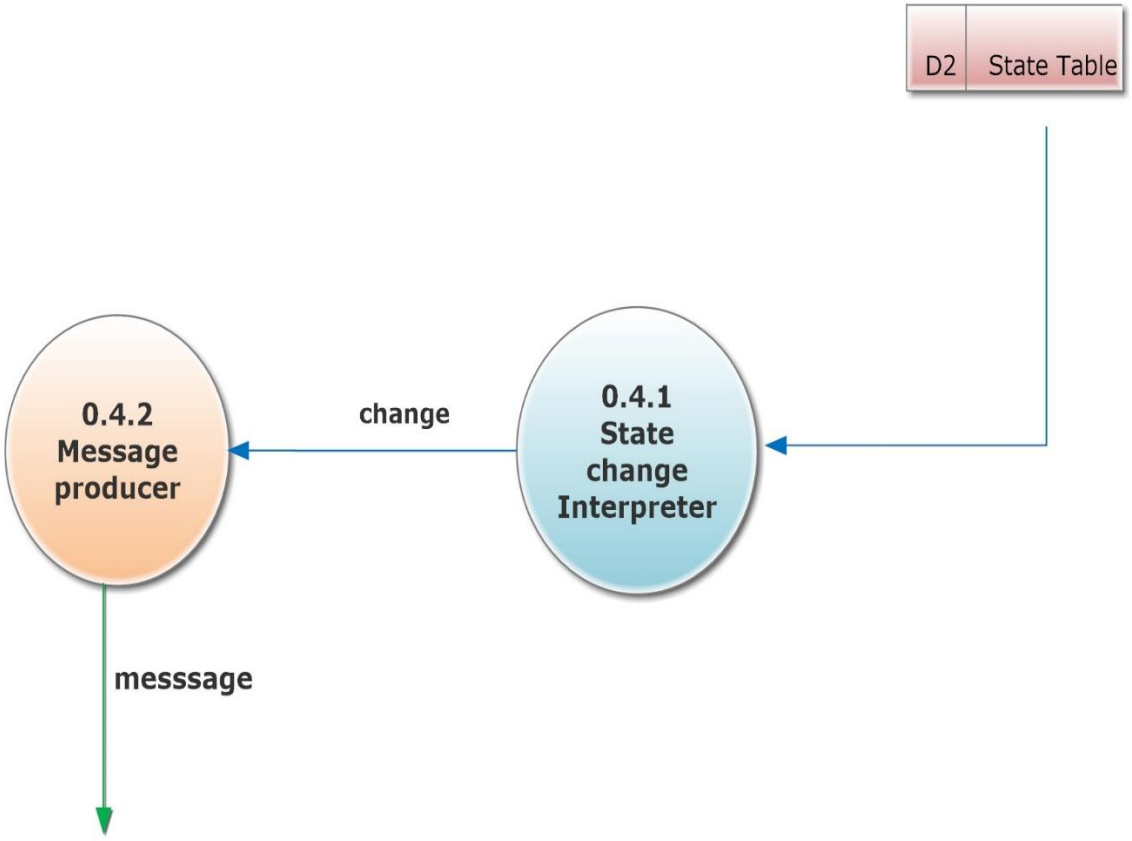


Func 0.3.1 : **Periodic clock signal generator** -- Generates a clock signal which becomes true at an interval of 5 minutes . Thus after every 5 minutes, the output of the process becomes true.
Func 0.3.2 : Random number generator -- Random numbers from 1 to 10 are generated for all students whenever the input to the process is true
Func 0.3.3 : Student state update -- The randomly generated number is sent for updation to the state table database D2.

Additional data dictionary:

periodic_update	Boolean /* True after every 5 minutes */
random_number	Integer /* Integer from 1 to 10 */

4.3.4 Message Generator



Func 0.4.1 : State change interpreter – Analyzes current state and previous state attributes of student from database D2 (state table) to determine if there has been a change in interpretation of state of a student.
Change in interpretation refers to a change in attention state of a student from one range to another for the following three ranges :

Interpretation	State
Low	1 to 4
Medium	5 to 7
High	8 to 10

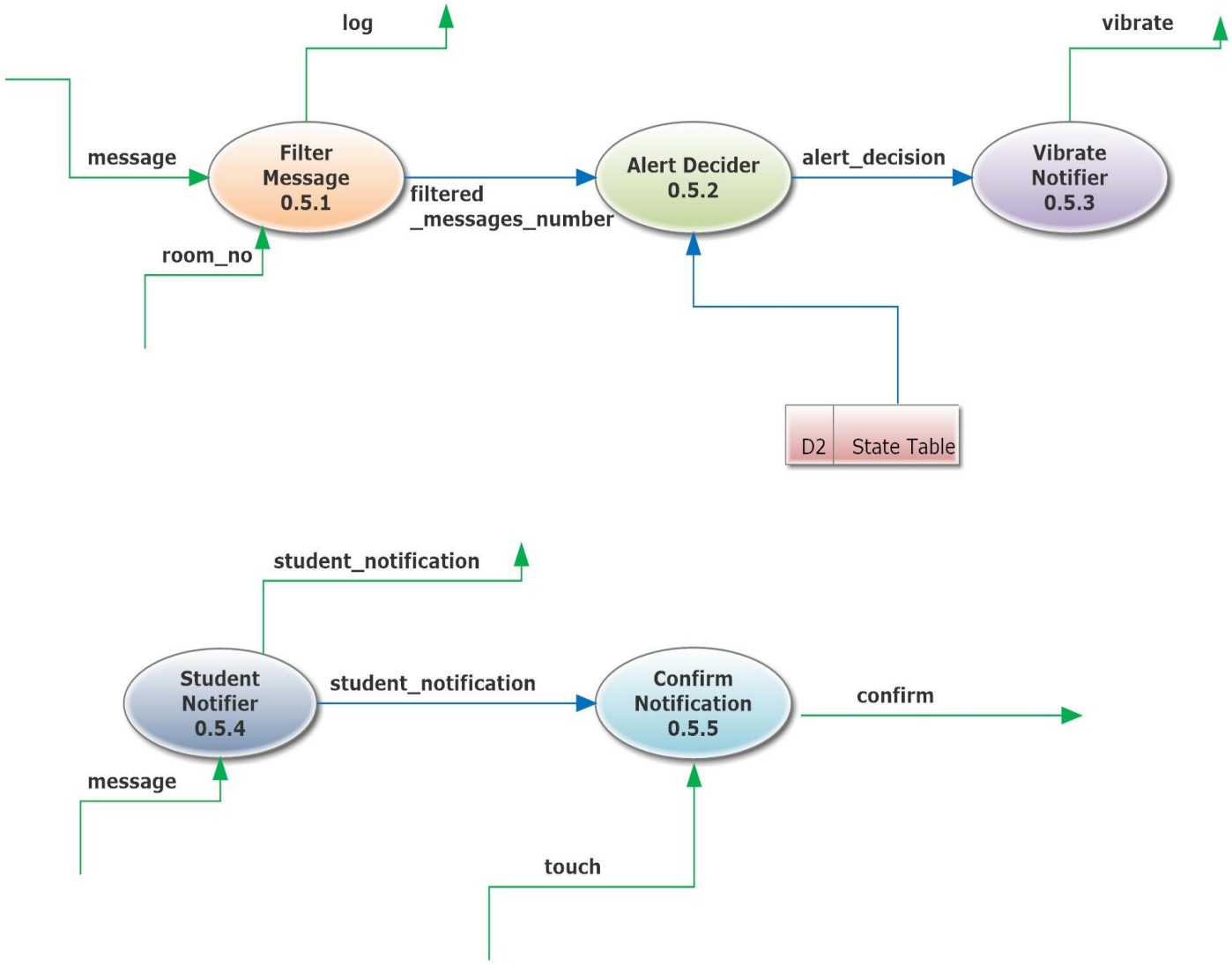
Whenever there is a change in the interpretation of the state of a student, this function will return 0 or 1 corresponding to low to high and high to low respectively,
otherwise 2.

Func 0.4.2 : **Message producer** – Generates message based on result of state change interpretor . Message produced contains room number, student username and state change interpretation.

Additional data dictionary:

change	room_no + username + [0 , 1 , 2] /* Whenever there is a change in the interpretation of the state of a student, this function will return 0 or 1 corresponding to low to high and high to low respectively, otherwise 2*/

4.3.5 Notifier



Func 0.5.1 : Filter message – Messages are filtered based on room number so that only the messages corresponding to the room number in which teacher is teaching are sent to log of the device belonging to the teacher. It also calculates the number of filtered messages with students' attention going down as well as up.

Func 0.5.2 : Alert Decider – It stores a counter which keeps track of the total number of students' whose attention is low by the formula:

c = c + filtered_messages_going_down – filtered_messages_going_up ;

Assuming that in the beginning of the class, all students' attention is high since they have just entered the class. Therefore the counter c is initialized with c=0 .

Then this function calculates the total strength of class corresponding to room number from the state table database D2. If more than 50 % of class attention is down, then this function returns True, else False.

Func 0.5.3 : Vibrate Notifier – If the input to the function i.e. alert_decision is true , then it sends a command to vibrate the phone otherwise it does nothing.

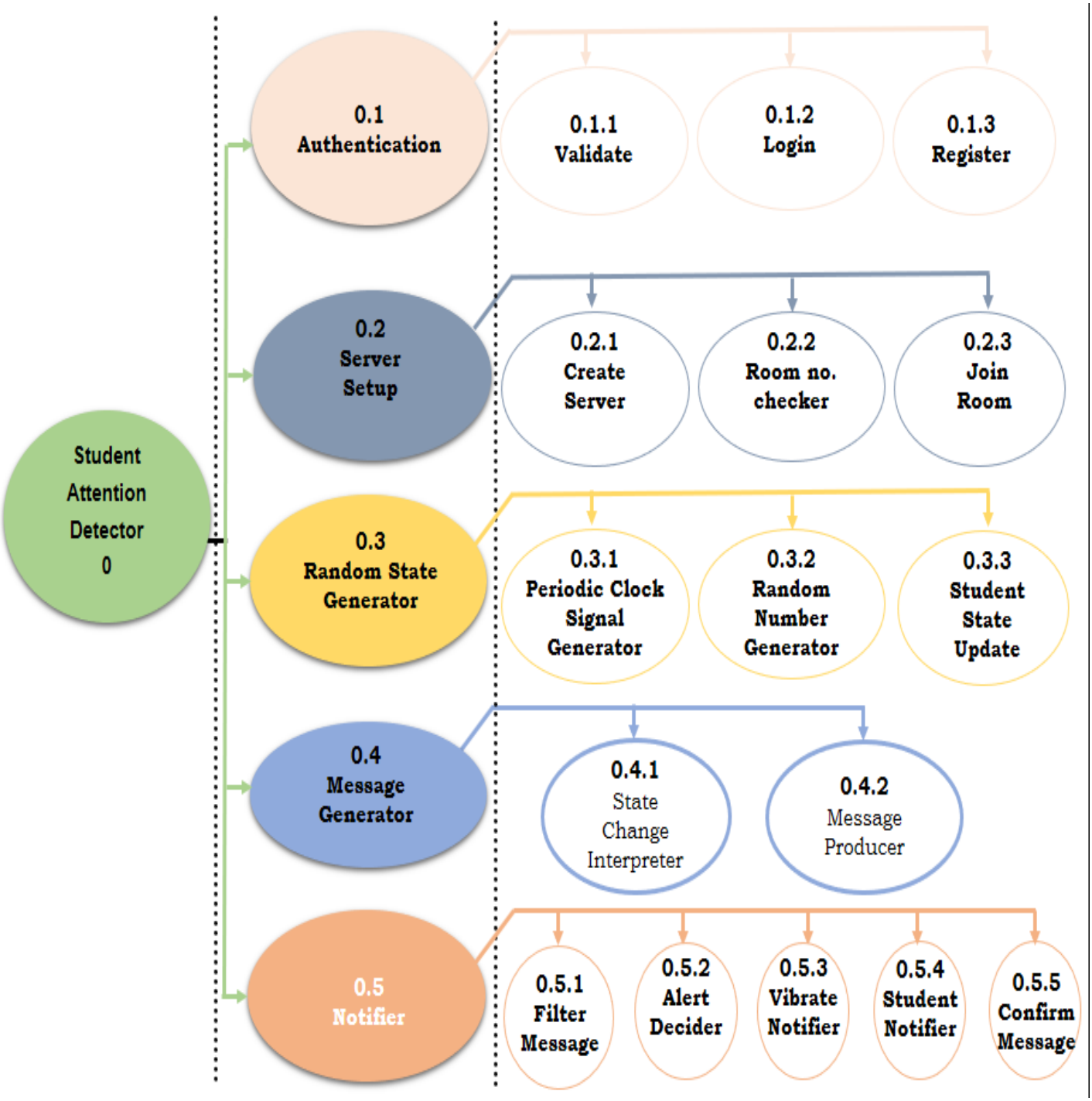
Func 0.5.4 : Student Notifier -- The message is pushed to corresponding student's device in the form of a notification.

Func 0.5.5 : Confirm Notification – The notification , if touched by user gets confirmed and the function returns true.

Additional data dictionary:

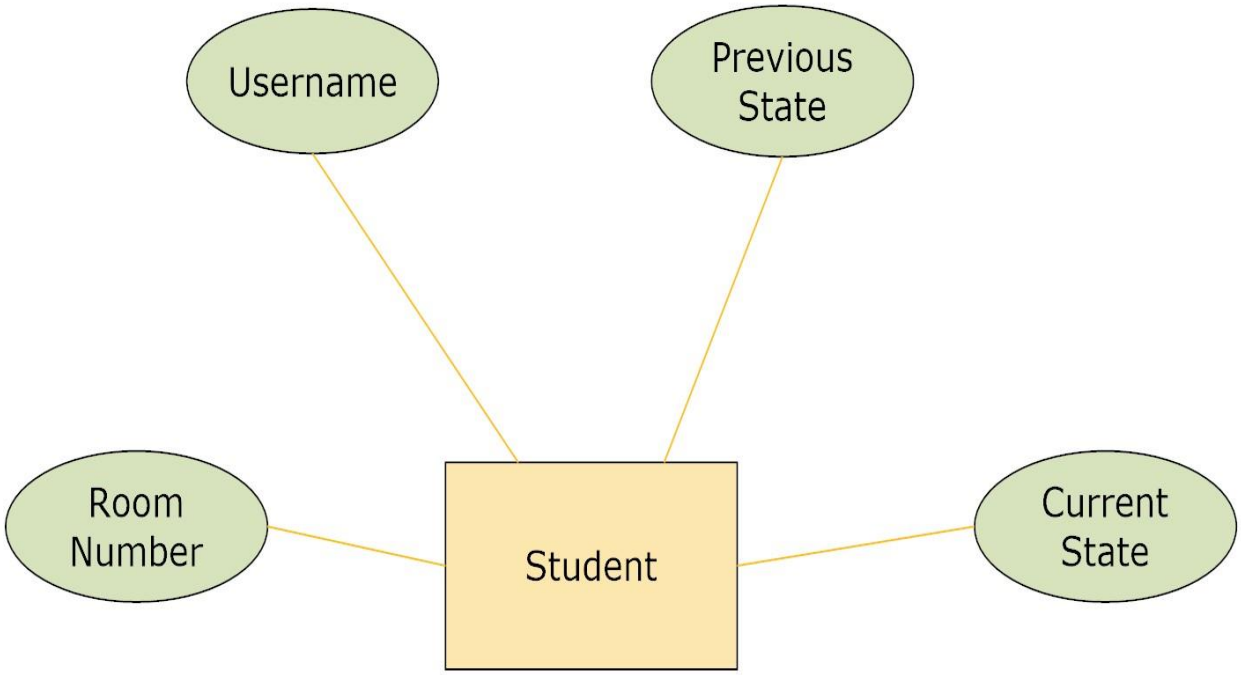
filtered_messages_number	filtered_messages_going_down + filtered_messages_going_up /* The total number of messages that were filtered at that point of time according to room number */
filtered_messages_going_down	Integer /* The total number of filtered messages which mean that students' attention is falling */
filtered_messages_going_up	Integer /* The total number of filtered messages which mean that students' attention is rising */
alert_decision	Boolean /* True if more than 50 % of students' attention is down otherwise False */

5. Process Decomposition



6. Entity Relationship Diagram

6.1 For State Table Database D2



6.2 For Users' List Database D1