Applied Information Technology And Computer Science Vol. 1 No. 1 (2020) 150-159

AITCS

Homepage: http://publisher.uthm.edu.my/periodicals/index.php/aitcs
e-ISSN : 2773-5141

University Student's Academic Scheduler

Chai Long Hui[1], Hanayanti Hafit[1]*

[1]Faculty Computer Science & Information Technology,
Universiti Tun Hussein Onn Malaysia, Parit Raja, 86400, MALAYSIA

DOI: https://doi.org/10.30880/aitcs.2020.01.01.015
Received 2 November 2020; Accepted 25 November 2020; Available online 30 December 2020

Abstract: The current existing application does not specialized for university students' need in schedule tasks and events within a single place. This project is to design and develop then test a new android mobile application that specialized for university students which help them to schedule task. The application is developed using the Object-Oriented (OO) concept and implemented using the Java programming language. Academic scheduler for university students may help university students scheduling their tasks. Hence, it may be a way of resolving the task scheduling problem faced by university students.

Keywords: Task Scheduler, University Student

1. Introduction

This section is dedicated to project introduction that review the project background of this project.

1.1 Project Domain

University students have compact and busy scheduler ongoing. Lack of inefficient task arrangement will cause the unknowing task to due first. According to the study from the University of Mary Washington, only recorded 46.9% of students have time management rates high or effective [1]. Another study shows 6.9% in Grade Point Average (GPA) variance different because of student time management skills. This concludes that academic GPA is effected by time management [2]. 85.4% of the students use different patterns of academic calendar applications to remind event but research found only 48.8% of students use when events are important [1]. Task scheduling applications today are based on patent from 1997 named "Method and system for schedule and task management" [3]. This kind of task schedule is too general in functionality. Hence, some of the current available task schedulers are able to communicate with system calendar application that enables synchronize features but not specify for university student's need. This leads to another problem which is task scheduler is lack of some feature that not specialized for university students. Hence, this leads some of the students to choose a calendar and reminder to record down lecture events and assignment/project tasks respectively. This is far more complicated for students because they need to switch back and forward to record down lecture event and task.

*Corresponding author: hana@uthm.edu.my

1.2 Objective

This project has three main objectives which to design, develop and test university students' academic scheduler. The application involves one user that is a university student who controls the entire operation of the application. The application has six main modules. There are event/task management module, routine module, user account and login register module and search module.

1.3 Project significant and expected output

By completing this project, this may help university students to arrange their task and recording events in one place. Application may help university students to simplify the process recording task and event by unified calendar and reminder. So, the tasks' schedule may resolve through using the proposed application. This application also practices critical tasks to be completed first. This may help university students know which task will be due first. Last, proposed application unifies both version of calendar and reminder without using a different application for different purposes.

2. Related Works

This section dedicated to the studies of previous works that used to develop the application.

2.1 Task Scheduler

The task scheduler is a task management system that schedules the task of the user. Therefore, the management system that involves an arrangement task will adapt scheduling techniques into the system. Thus, the task scheduler is able to schedule and monitor the task. Inside the task scheduler system, the task requires allocates itself, which requires scheduling algorithms.

In this task scheduler, the allocation of the task is limited by the constraint of the time, date and deadline. In the nutshell, the student's task scheduler involves lectures, assignments, project and final examination that requires separate arrangements of each event scheduling as some of the tasks are not event that did not have a fixed time. This is because the event has a hard constraint. Hard constraints are the constraint that cannot be violated no matter how worst the schedule. If the constraints break, the schedule becomes meaningless. While soft constraints are constraint set to be scheduled. This meant after the task schedule need to avoid hard constraint events [4].

2.2 Shortest Time Remaining Algorithm

The shortest time remaining algorithm is used to determined which task need to complete first based on the due date entered by the user. The algorithm trigger once the task is exist and allocate the task into highest priority with shortest left time left. This is used to determine the order of the task to complete on the list. This algorithm ensures the task module in the application can work seamlessly and help university students decide which task due first and need to do first.

2.3 Java Programming Language

The application is implemented on the Android platform. It is implemented using Java programming language. Java is an Object-Oriented programming (OOP) language that develops by James Gosling in Sun Microsystem company [5]. Java is adapting a lot of friendly features like Object-Oriented (OO) which includes characteristic of Encapsulation, Inheritance, Abstraction and Polymorphism, Robust and platform independent. This advance characteristic of the Java programming is useful for the development of the application as the amount of source code can be reduce.

2.4 Firebase Realtime Database

As the application records and manages the university student task and event, hence, it needs a database to store of the user content. The application selects Firebase Realtime database as database and authentication ser-vice by Google. Google aims to provide database infrastructure to develop mobile applications. The Firebase is a type of application programming interface (known as API) and opens available for mobile and web applications [6]. The Firebase offer in two types of database which is Realtime database and not Realtime database known as Cloud Firestore [7].

2.5 Review Current Similar Application

To produce application that meet public standard, it is important to study about the existing application that similar to the developed application for better understanding. Total of three applications named "Class Timetable", "Weekly Timetable" and "The Homework App" is selected. Comparison of the application functionality on different feature is described in Table 1.

Table 1: Comparison current similar system and proposed system

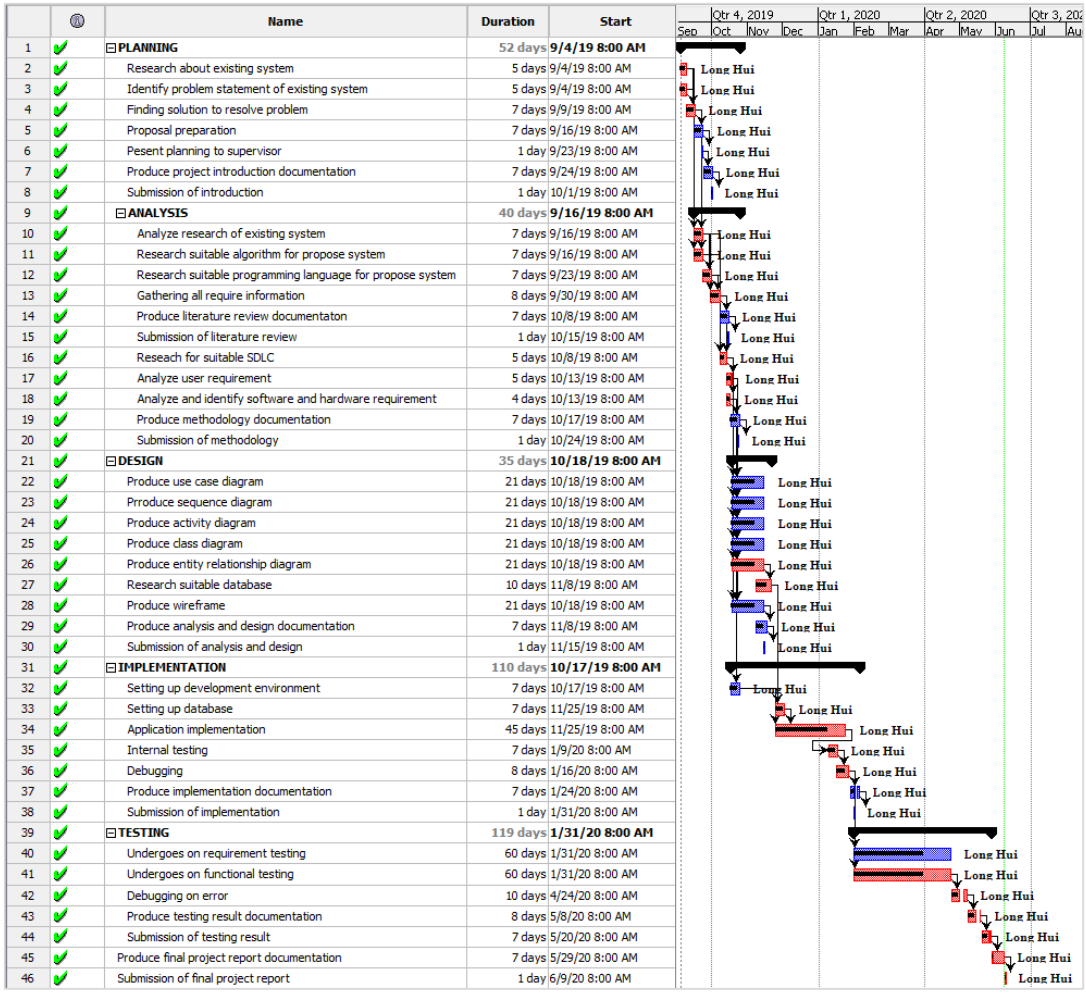
Feature	Class Timetable	Weekly Timetable	The Homework App	Proposed
Timetable Schedule	✓	✓	✓	✓
Task Schedule	✓	✓	✓	✓
Notification on Task	X	✓	✓	X
Course Scope	User Define	User Define	User Define	Built in
View Style	List View	List View	Timeline	Card View
Account Mode	X	X	X	✓
Highlight Important Upcoming Task	X	X	✓	✓
Priority Arrangement on Task Schedule	X	X	✓	✓
Background Algorithm	X	X	Priority Scheduling Algorithm	Shortest Time Remaining
Output	Timetable and Task Schedule	Timetable and Task Schedule	Timetable and Task Schedule	Event Schedule, Task Schedule

3. System Methodology

The methodology in developing this application is object-oriented development model. The existing problem identifies and proposes a new solution regarding the problem that currently exists on the existing system. In the planning phase, the interview session with UTHM students are undergone to understand UTHM students plan their task. In the meanwhile, the information planning stage is documented as guidance for the upcoming phase. Hence, the Gantt chart is produced as planning documentation as a project guide. Figure 1 shows the project Gantt chart. While for the analysis phase, the user requirement for the proposed application is identified to develop an application that suit for university students. This phase identifies the application functionality, application specification, application usability and application suitability. Design phase follow after analysis phase, the abstraction view of all application components like user interface design and database design is

produced to act as blueprint to the implementation phase. While, entity-relationship diagram (ERD) diagram in the analysis phase help in developing a database conceptual view.

Then, the implementation phase is where the coding part comes into. All of the information gathering in the analysis phase, wireframe design of user interface and conceptual model of the database in the design phase translates into application in this phase. The application is developed using Java programming language in Android Studio IDE. The user interfaces are also developed using Android Studio IDE and extensive markup language (XML). Moreover, the database is using Firebase Realtime database and it is located at the Google data center. In the testing phase, the complete proposed application is tested according to the test plan. To make sure application run without error or bug, all modules of the application are tested. Debugging is made if application counters error or bug during the testing phase. This ensures application standard and requirements. Lastly, the methodology is end up with maintenance phase.



	Name	Duration	Start
1	**PLANNING**	52 days	**9/4/19 8:00 AM**
2	Research about existing system	5 days	9/4/19 8:00 AM
3	Identify problem statement of existing system	5 days	9/4/19 8:00 AM
4	Finding solution to resolve problem	7 days	9/9/19 8:00 AM
5	Proposal preparation	7 days	9/16/19 8:00 AM
6	Pesent planning to supervisor	1 day	9/23/19 8:00 AM
7	Produce project introduction documentation	7 days	9/24/19 8:00 AM
8	Submission of introduction	1 day	10/1/19 8:00 AM
9	**ANALYSIS**	40 days	**9/16/19 8:00 AM**
10	Analyze research of existing system	7 days	9/16/19 8:00 AM
11	Research suitable algorithm for propose system	7 days	9/16/19 8:00 AM
12	Research suitable programming language for propose system	7 days	9/23/19 8:00 AM
13	Gathering all require information	8 days	9/30/19 8:00 AM
14	Produce literature review documentaton	7 days	10/8/19 8:00 AM
15	Submission of literature review	1 day	10/15/19 8:00 AM
16	Reseach for suitable SDLC	5 days	10/8/19 8:00 AM
17	Analyze user requirement	5 days	10/13/19 8:00 AM
18	Analyze and identify software and hardware requirement	4 days	10/13/19 8:00 AM
19	Produce methodology documentation	7 days	10/17/19 8:00 AM
20	Submission of methodology	1 day	10/24/19 8:00 AM
21	**DESIGN**	35 days	**10/18/19 8:00 AM**
22	Produce use case diagram	21 days	10/18/19 8:00 AM
23	Prroduce sequence diagram	21 days	10/18/19 8:00 AM
24	Produce activity diagram	21 days	10/18/19 8:00 AM
25	Produce class diagram	21 days	10/18/19 8:00 AM
26	Produce entity relationship diagram	21 days	10/18/19 8:00 AM
27	Research suitable database	10 days	11/8/19 8:00 AM
28	Produce wireframe	21 days	10/18/19 8:00 AM
29	Produce analysis and design documentation	7 days	11/8/19 8:00 AM
30	Submission of analysis and design	1 day	11/15/19 8:00 AM
31	**IMPLEMENTATION**	110 days	**10/17/19 8:00 AM**
32	Setting up development environment	7 days	10/17/19 8:00 AM
33	Setting up database	7 days	11/25/19 8:00 AM
34	Application implementation	45 days	11/25/19 8:00 AM
35	Internal testing	7 days	1/9/20 8:00 AM
36	Debugging	8 days	1/16/20 8:00 AM
37	Produce implementation documentation	7 days	1/24/20 8:00 AM
38	Submission of implementation	1 day	1/31/20 8:00 AM
39	**TESTING**	119 days	**1/31/20 8:00 AM**
40	Undergoes on requirement testing	60 days	1/31/20 8:00 AM
41	Undergoes on functional testing	60 days	1/31/20 8:00 AM
42	Debugging on error	10 days	4/24/20 8:00 AM
43	Produce testing result documentation	8 days	5/8/20 8:00 AM
44	Submission of testing result	7 days	5/20/20 8:00 AM
45	Produce final project report documentation	7 days	5/29/20 8:00 AM
46	Submission of final project report	1 day	6/9/20 8:00 AM

Figure 1: Project Gantt chart

4. System Analysis and Design

This section dedicated to application analysis and its design.

4.1 Use case

The application started with system analysis that described use case and activity. The application has fifteen type of the use case. They are login, register under register and login module, password reset, account update, account display, account deactivate under account module, routine view under routine

module, category search under search module, task create, display, delete and update under event management module and event create, display, delete and update under event management module. Figure 2 shows the use case of the application.

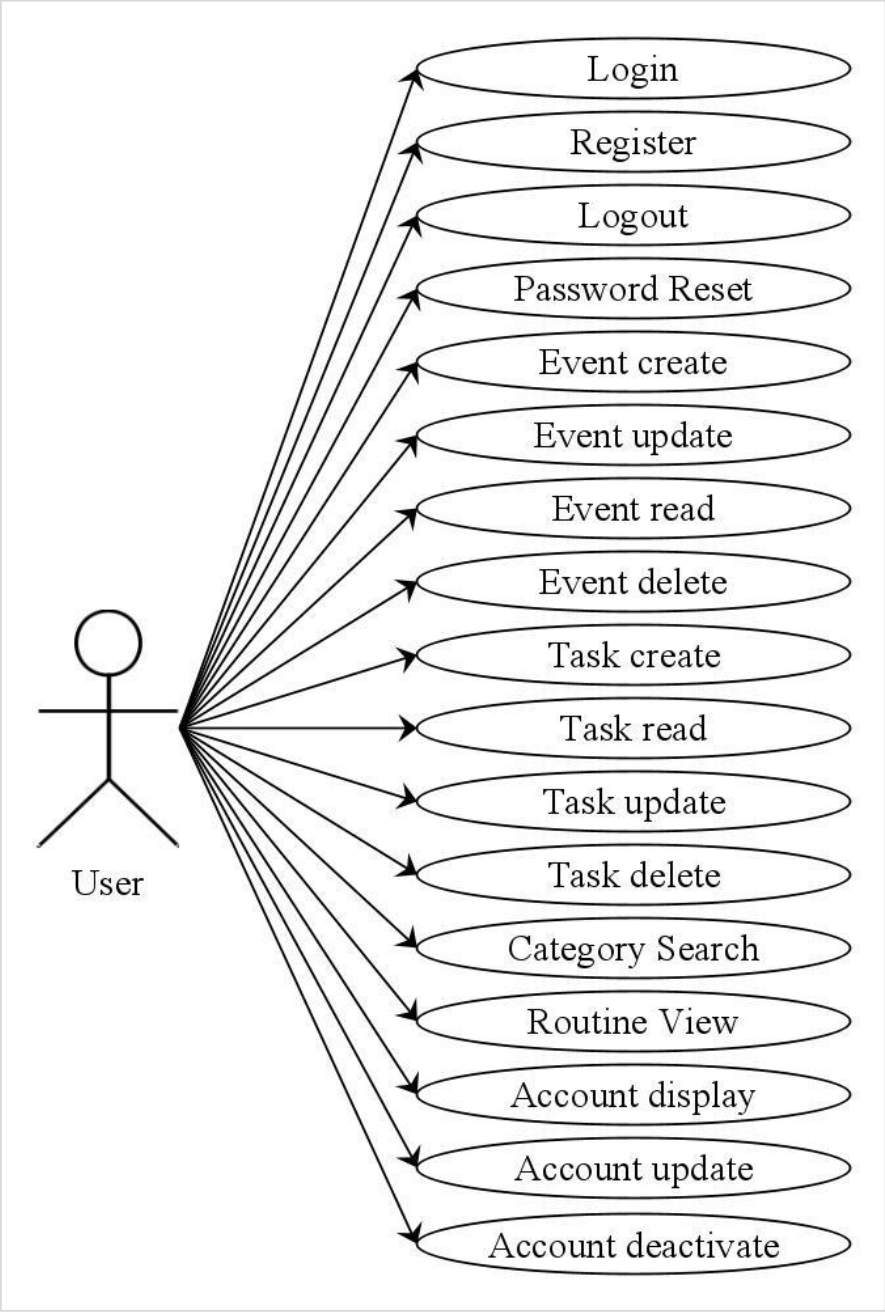


Figure 2: Classes diagram of university student academic scheduler application

4.2 User Activity

The activity of the system analysis describes the flow of the application. The university student academic scheduler application starts with the login page. The user is required to login to access the application. For the new user, the user can log in by register a new account. Then, the user can proceed with the registered account to log into the system. After the user successful log into the application, the user can perform various kind of function such as create a new event, update, delete or view the existing event or update current user profile or view event in category way. The homepage shows the event and task. Users can add, view, update and delete the event through this interface. Then, the user also can

navigate to other pages like category or user account page by using the navigator located at the bottom of the page. Lastly, the user can log out from the application by pressing the log out button at the user account page. Figure 3 shows the activity diagram of the application.

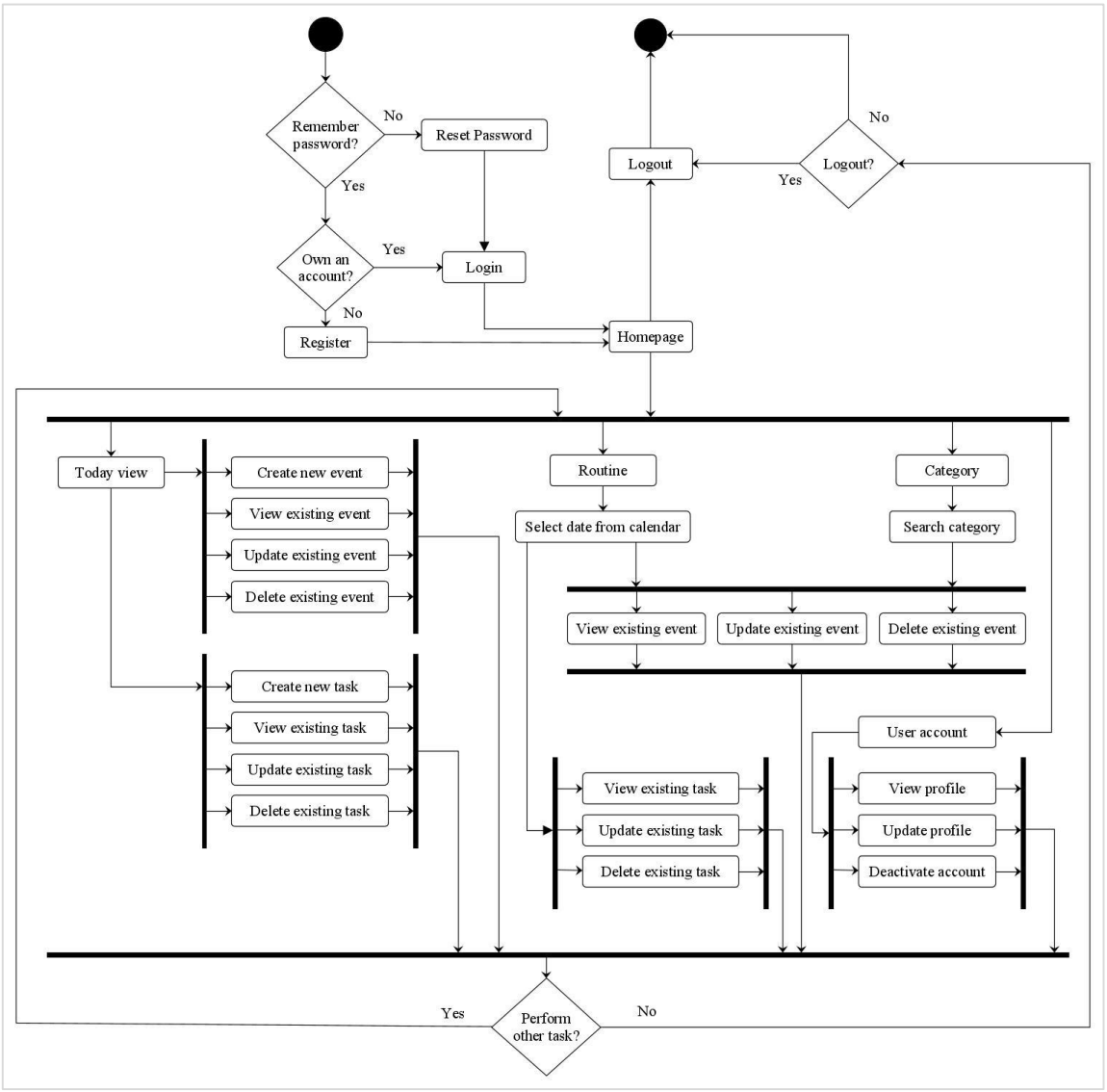


Figure 3: Entity Relationship Diagram (ERD) of university student academic scheduler application

4.3 Application Classes

The application design described about application classes and its database design. The application has six main modules that summarize up total twenty-four classes found in the application. Figure 4 illustrated the classes built in the application.

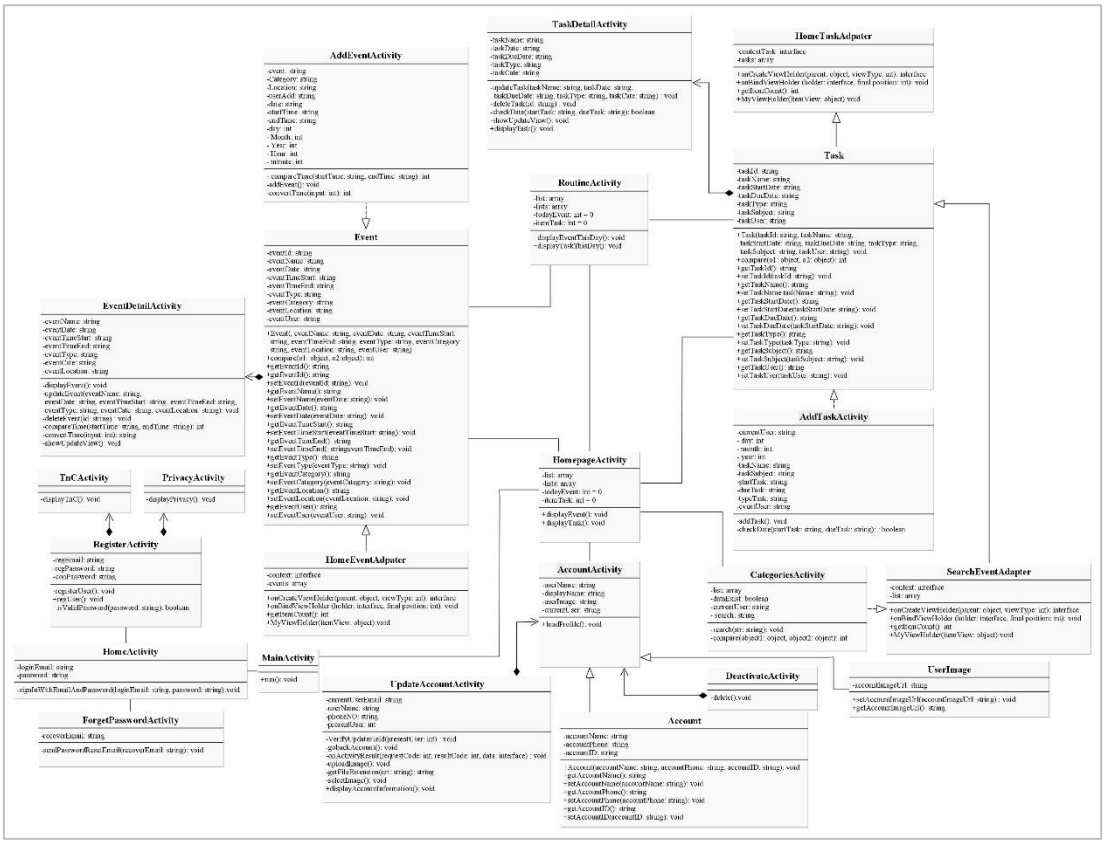


Figure 4: Classes diagram of university student academic scheduler application

4.4 Database

The application used Google Firebase Realtime database as the database. The application database consists 3 table which name “task”, “event” and “users”.

5. Implementation and Testing

This section dedicated to the implementation and testing of this application.

5.1 Implementation

University student academic scheduler is developed using android studio IDE. The database of this application is the Firebase authentication for the login module and register module while Firebase Realtime database for storing and retrieving user data and Firebase storage for saving user profiles. This application is implemented using Java programming language and Extensible Markup Language (XML). The Java class handling logic and algorithm and XML rendering the application layout. This application has six main modules. They are user account, login and register module, task management module, event management module, routine module, and categories search module. The user account module handles user profile update, view, and deactivate the account. Then, the login module ensures the user authenticate themselves before access to the application. This helps protect their personal information. While the register module allows the new user to obtain a new account to use this application.

For the event management module and task management module is used to handle user event and task data operation. The module allows the user to create new events or tasks, view existing events and tasks, update current existing events and tasks, and delete existing events or tasks. This module handling the scheduling view ensures event sorting into timeline view and schedule task into a critical task to complete first list. Figure 6 shows the code segment of today view that include today event and task to

complete first. Figure 7 shows the code segment of the short time remaining algorithm implement into application to determine task need to do first.

```
databaseReference = FirebaseDatabase.getInstance().getReference().child("event");
databaseReference.orderByChild("eventUser").equalTo(currentUser).addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
        for(DataSnapshot dataSnapshot1: dataSnapshot.getChildren()){
            Event e = dataSnapshot1.getValue(Event.class);
            if (e.getEventDate() != null && e.getEventDate().equals(displayToday)) {...}
            Collections.sort(list,Event.BY_TIME_ASCENDING);
        }
        if (todayEvent != 0) {
            homeEventAdapter = new HomeEventAdapter( context: HomepageActivity.this,list);
            recyclerView.setAdapter(homeEventAdapter);
        }else{...}
    }

    @Override
    public void onCancelled(@NonNull DatabaseError databaseError) {...}
});

//today on Task
recyclerView2 = (RecyclerView) findViewById(R.id.recyclerView);
recyclerView2.setLayoutManager(new LinearLayoutManager( context: this));
lists = new ArrayList<Task>();
databaseReference2 = FirebaseDatabase.getInstance().getReference().child("task");
databaseReference2.orderByChild("taskUser").equalTo(currentUser).addValueEventListener(new ValueEventListener() {

    @Override
    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
        for(DataSnapshot dataSnapshot1: dataSnapshot.getChildren()){...}
        if (itemTask != 0) {
            Collections.sort(lists,Task.BY_TIME_LEFT_SORT);
            homeTaskAdapter = new HomeTaskAdapter( ct: HomepageActivity.this,lists);
            homeTaskAdapter.notifyDataSetChanged();
            //...
            recyclerView2.setAdapter(homeTaskAdapter);
        }else{...}
    }
```

Figure 6: Code segment to event and task reader

```
    public int compare(Task t1, Task t2) {

        DateTimeFormatter dtf = DateTimeFormatter.ofPattern("[d-M-yyyy][dd-M-yyyy][d-MM-yyyy][dd-MM-yyyy]");

        String inputString1 = t1.getTaskDueDate();
        String inputString3 = t2.getTaskDueDate();

        //today
        DateFormat dateFormat = new SimpleDateFormat( pattern: "dd-MM-yyyy");
        Date date = new Date();
        String now = dateFormat.format(date);
        LocalDate today = LocalDate.parse(now, dtf);

        LocalDate date1 = LocalDate.parse(inputString1, dtf);
        LocalDate date3 = LocalDate.parse(inputString3, dtf);

        Long daysBetweenT1 = ChronoUnit.DAYS.between(today, date1);
        Long daysBetweenT2 = ChronoUnit.DAYS.between(today, date3);

        return daysBetweenT1.compareTo(daysBetweenT2);
```

Figure 7: Code segment to determine the order of task to complete first

The routine module displays the due task and event on a selected date. This allows users to find due tasks and events on a particular date. This module extends the event and task management module which allows the user to directly manipulate data after view. Last, the category search module enables users to search the event belong to its categories. This helps user to find all specific category by querying the event category. The user interface of the application is developed along with the logic by using Extensible Markup Language in Android Studio IDE. There are four main screens and a total of 12 user interfaces. Figure 8 shows the application user interfaces.

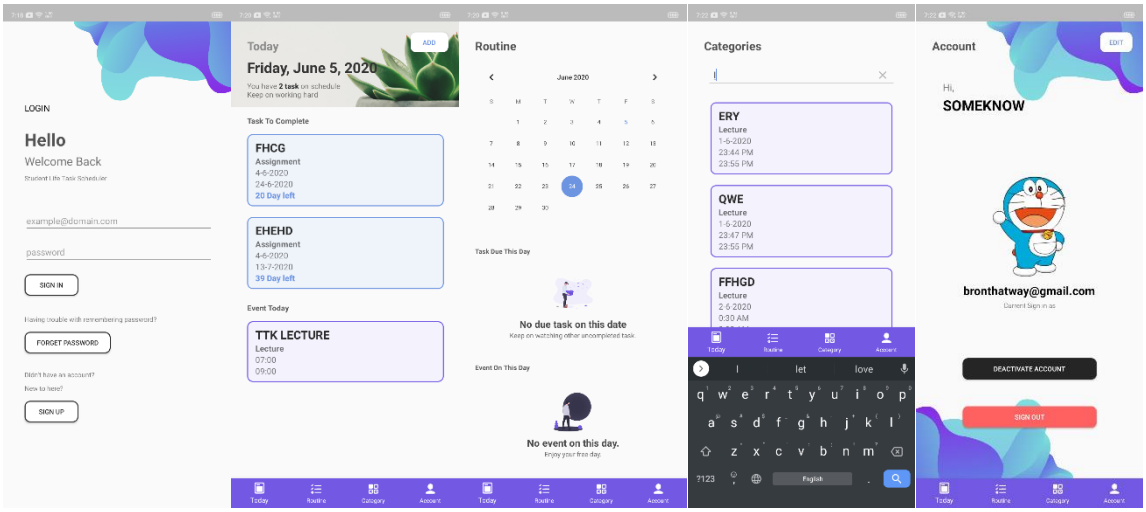


Figure 8: User interfaces of the application

3.2 Testing

To ensure each module in this application meet the requirement and functioning properly, hence, the testing phase is implemented to ensure each module working according to expected output under the given condition. After performing testing and observer each module output result in the application, all six modules in the application passed the functional testing. This ensures the application operates without error and bug. Table 2 shows the result of this application test plan.

Table 2: Functional testing of the application

Modules	Test	Expected result	Output result
Register & Login module	User enter incorrect or invalid format email and password	Application prompt user to enter valid email or password	Pass
	User enter correct format email and strong password	Application register user as all field is not empty and match requirement	Pass
	User enter wrong email and password combination	Application prompt username and password combination is wrong	Pass
	User enter correct email and password combination	Application log user into homepage of the application	Pass
Event management module	User create new or update current existing event with valid input and format	Application create or update event detail entered information and show on application	Pass
	User tap on exiting event in the application	Application show event detail on screen	Pass
	User tap on event delete button	Application remove event from the application and database	Pass
Routine module	User select date from calendar	Application show the event and due task for selected date	Pass

Table 2 (cont.)

Modules	Test	Expected result	Output result
Categories module	User enter event categories that exist in user database	Application return all event that belong to category enter by user else return no result found	Pass
User account module	User view account information that user account	Application display user account information	Pass
	User update information with valid information and format	Application update user account information	Pass
	User deactivate account	Application remove user account and data	Pass

6. Conclusion

University student's academic scheduler has been developed successfully and meets all objectives. This application unifies the calendar and task reminder to help students focus on one application and centralize the academic schedule information. Thus, the limitation can transform into future work that ensures to resolve its limitation and improvement this application functionality and usability.

Acknowledgement

The authors would like to thank the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia for its support and encouragement throughout the process of conducting this study.

References

[1] J. Mei, "Learning Management System Calendar Reminders and Effects on Time Management and Academic Performance", International Research and Review: Journal of Phi Beta Delta Honor Society for International Scholars, vol. 6 no. 1, pp.29-45, Fall 2016.

[2] A. Pehlivan, "The Effect of the Time Management Skills of Students Taking a Financial Accounting Course on their Course Grades and Grade Point Averages", International Journal of Business and Social Science vol. 4 no. 5, pp. 196-203, May 2013.

[3] S. F. Mansour, "Method and system for schedule and task management," U.S. Patent 6 035 278, Mar. 7, 2000.

[4] N. Anirudha et al., "An Algorithm to Automatically Generate Schedule for School Lectures Using a Heuristic Approach," International Journal of Machine Learning and Computing vol. 2 no. 4, pp. 492-495, August 2012.

[5] S. W. Sangle, "Study on Java Language," in International Conference on Recent Trends in Engineering Science and Technology, ICRTEST 2017, Selukate, Maharashtra, India, January 21-22, 2017, Suresh Deshmukh College of Engineering Selukate Wardha: IJRITCC, 2017. pp. 281-284.

[6] L. Moroney, The Definitive Guide to Firebase: Build Android Apps on Google's Mobile Platform. Seattle, WA: Apress, 2017.

[7] N. Smyth, Firebase Essentials - Android Edition. Scotts Valley, CA: CreateSpace Independent Publishing Platform, 2017.