CS3354 Software Engineering
Final Project Deliverable 2

Group 10
The Fourth Dimension

Mohamed Elkhaled
Muhammad Hamdan
Jose Francisco De La Cruz
Shahneel Lalani
Nicholas Vitale
Akhil Mathew
Elam Bruce

1. Well described delegation of tasks, i.e. who did what in the project. Now that your project is complete, you are required to submit the delegation of tasks from beginning of the project until the end. Please make sure to fairly distribute tasks in the team and remember that in the end of the semester, each member of a team will receive the same grade. See grading policy below for more detail. If no/poor contribution by a member, please specify clearly so that we can grade each student fairly.

Mohamed Elkhaled
- Function and non-functional requirements
- Compiled content from project deliverable 1 and deliverable 2 into a powerpoint

Muhammad Hamdan
- Explain which Architectural Design we used and why
- Compared and contrasted different designs with our proposed designs
- Provided in-depth research to validate the uniqueness of our design

Jose Francisco De La Cruz
- Explained which software process model is employed in the project and why
- Described the delegation of tasks
- Described the conclusions drawn from the extensive research and effort needed to produce a roadmap for a new technology

Shahneel Lalani
- Set up the GitHub repository and add all team members and TA
- Researched/calculated the estimated costs, time and complexity of producing our design

Nicholas Vitale
- Created sequence diagrams of the features our design will implement
- Assisted with the research for the cost, time and complexity of producing our design

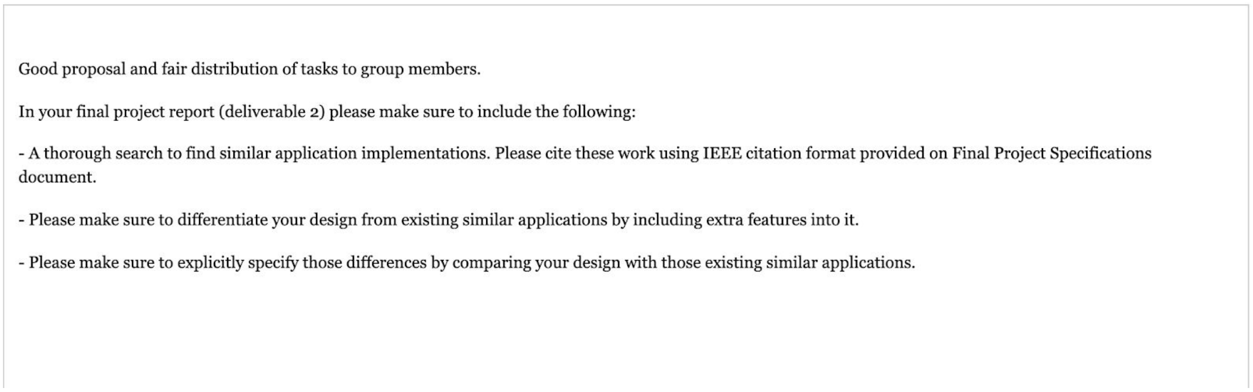
Akhil Mathew
- Created the Use Case Diagram
- Created and edited the video presentation
- Compiled everyone’s work, proofread, and submitted for each deadline

Elam Bruce
- Created the class diagrams for our design
- Created the JUnit test cases for our design

2. Project Deliverable 1 Content:

Please attach here the Final Project draft description (that contains the instructor feedback). Address the feedback provided for your proposal by listing what you did / plan to do to comply with those proposed changes and or requests for additions to your project.

INSTRUCTORS FEEDBACK



Good proposal and fair distribution of tasks to group members.

In your final project report (deliverable 2) please make sure to include the following:

- A thorough search to find similar application implementations. Please cite these work using IEEE citation format provided on Final Project Specifications document.

- Please make sure to differentiate your design from existing similar applications by including extra features into it.

- Please make sure to explicitly specify those differences by comparing your design with those existing similar applications.

Google Calendar:

Google Calendar: Free Calendar App for Personal Use. [Online]. Available: https://www.google.com/calendar/about/. [Accessed: 08-Oct-2020].

Apple Calendar:

“Calendar User Guide for Mac,” *Apple Support*. [Online]. Available: https://support.apple.com/guide/calendar/welcome/mac. [Accessed: 08-Oct-2020].

Microsoft Outlook Calendar:

“Introduction to the Outlook Calendar,” *Outlook*. [Online]. Available: https://support.microsoft.com/en-us/office/introduction-to-the-outlook-calendar-d94c5203-77c7-48ec-90a5-2e2bc10bd6f8. [Accessed: 08-Oct-2020].

The 4th Dimension:

Although other calendar applications are very useful, our software engineering project provides users of all makes and backgrounds a comprehensive dose of organization. The 4th Dimension (4D) is an intuitive calendar application that allows users to experience the same task management skills as the world's best chief administrative officers. 4D divides the user's schedule into three main categories: daily, weekly, and monthly view, including an optional view called agenda. Each view is composed of events that are compared to each other to avoid time conflicts. Furthermore, users have the option to add driving routes to inform them of the best time to leave for their event. Although 4D appears to be another run of the mill calendar application, the tools and features will quickly dissuade any user from viewing it as such.

Project Repository URL: https://github.com/S-Lalani/3354-TheFourthDimension.git

Delegation of tasks:

As far as the responsibility goes, we all were very responsive to each other and were able to quickly manage and delegate all of the tasks that needed to be done for Project Deliverable 1. Here is a list of each team members individuals tasks and contributions:

Shahneel Lalani - set up the GitHub repository and add all team members and TA
Muhammad Hamdan - explain which Architectural Design we used and why
Jose De La Cruz - explain which software process model is employed in the project and why
Mohamed El-Khaled - work on functional and nonfunctional requirements
Akhil Mathew - work on the use case diagram
Nick Vitale - work on the sequence diagrams
Elam Bruce - work on the class diagram

Which software process model is employed in the project and why.

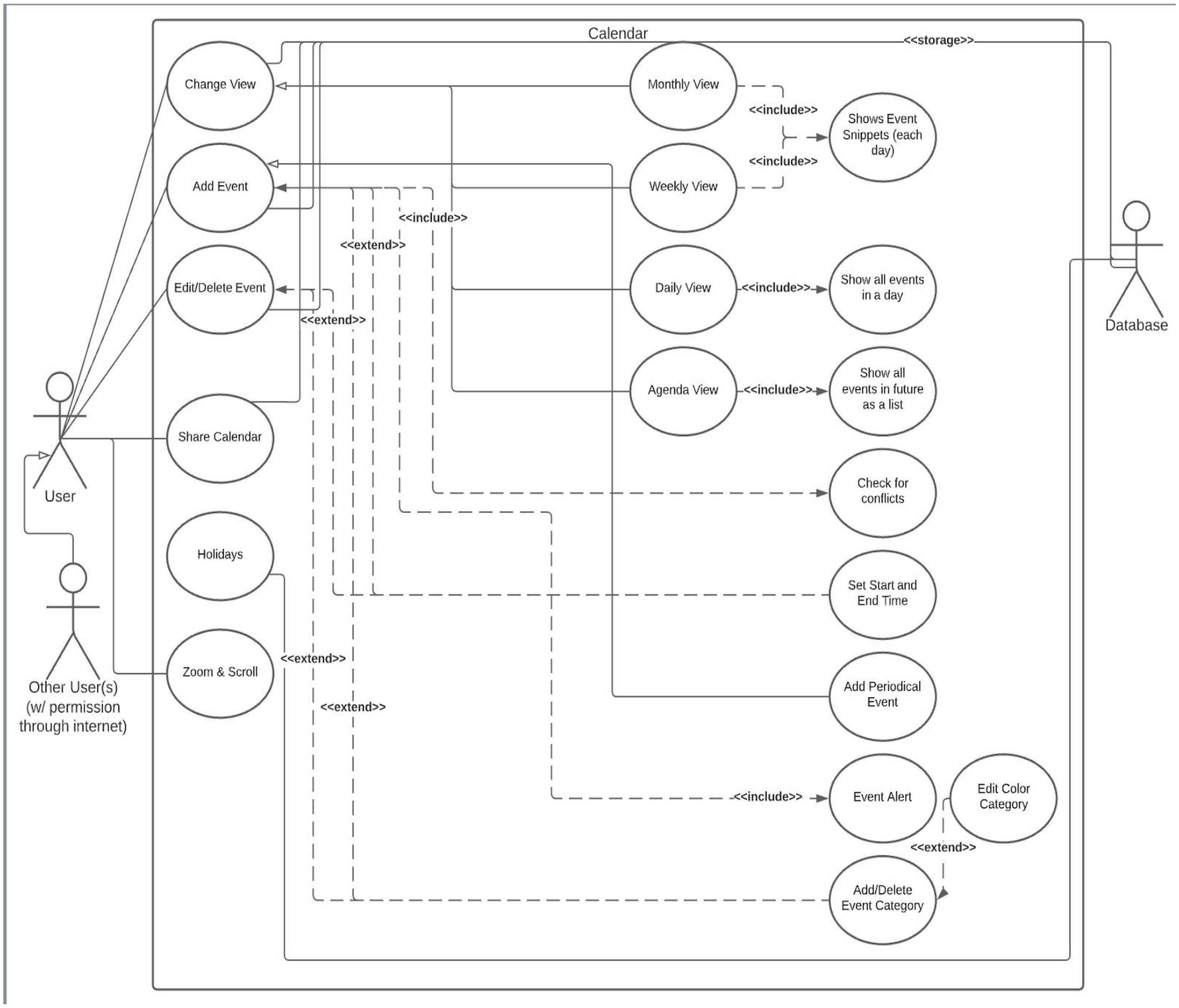
For the software process model, our group employed the Waterfall Model since it defines all the basic activities for a software process, so it is easy to track progress on the project. Also a detailed plan is created before work is started, and the plan remains the same throughout the duration of the project, thus it fits the criteria for our software implementation.

<u>Software Requirements</u>

- **Functional requirements:**
  - A User will be able to add events on a certain date
  - A User will be able to check general holidays or events on any given date
  - The Calendar will save any and all events that the user adds
  - The user will be able to plan his/her day hour by hour for each date
  - Users will be able to share their day plan through text or email
  - This software will be available for download on any smart phone or PC

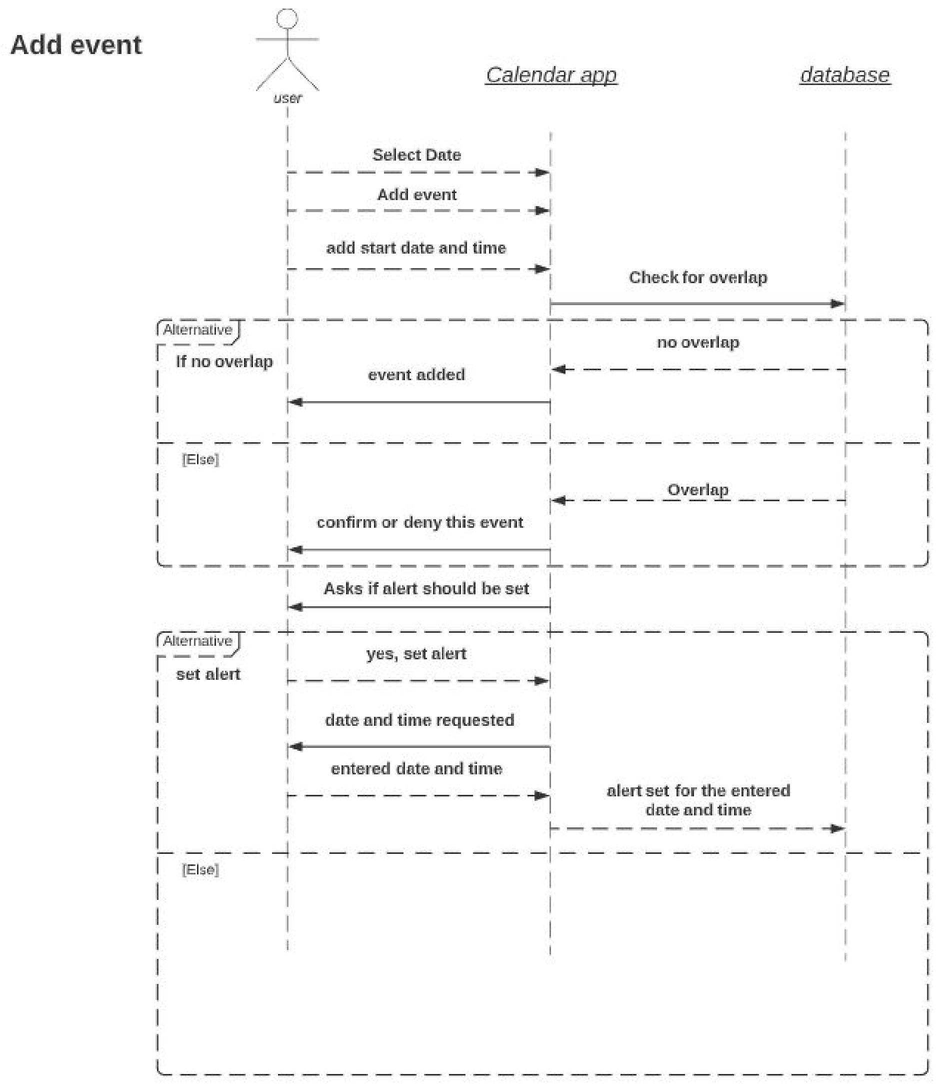
- **Non-functional requirements:**
  - Usability Req: software will be able to move from PC to phone with no problem
  - Dependability Req: Software will auto update when one is available
  - Security Req: software will require a password before opening
  - Regulatory Req: <u>ASSUMING IT'S FINE BY STATE LAW</u>, The software can save data users put in their daily use for law enforcement benefits only
  - Ethical Req: software will not share any user data for a profit
  - Environmental Req: Must have a smart device or PC to download
  - Operational Req: when adding must be connected to wifi
  - Development Req: when developing must have access to a MacBook for app store use
  - Performance Req: Software will load up completely in 3 seconds
  - Space Req: Software will be relatively small to be able to be downloaded on any smart device
  - Accounting Req: <u>ASSUMING NO LAWS STOP USERS</u>: there will be no restrictions on who can download the software
  - Safety Req: Per customer Safety the data collected will not be shared with anyone for any reason unless for legal issues

Use case diagram – Provide a use case diagram for your project.

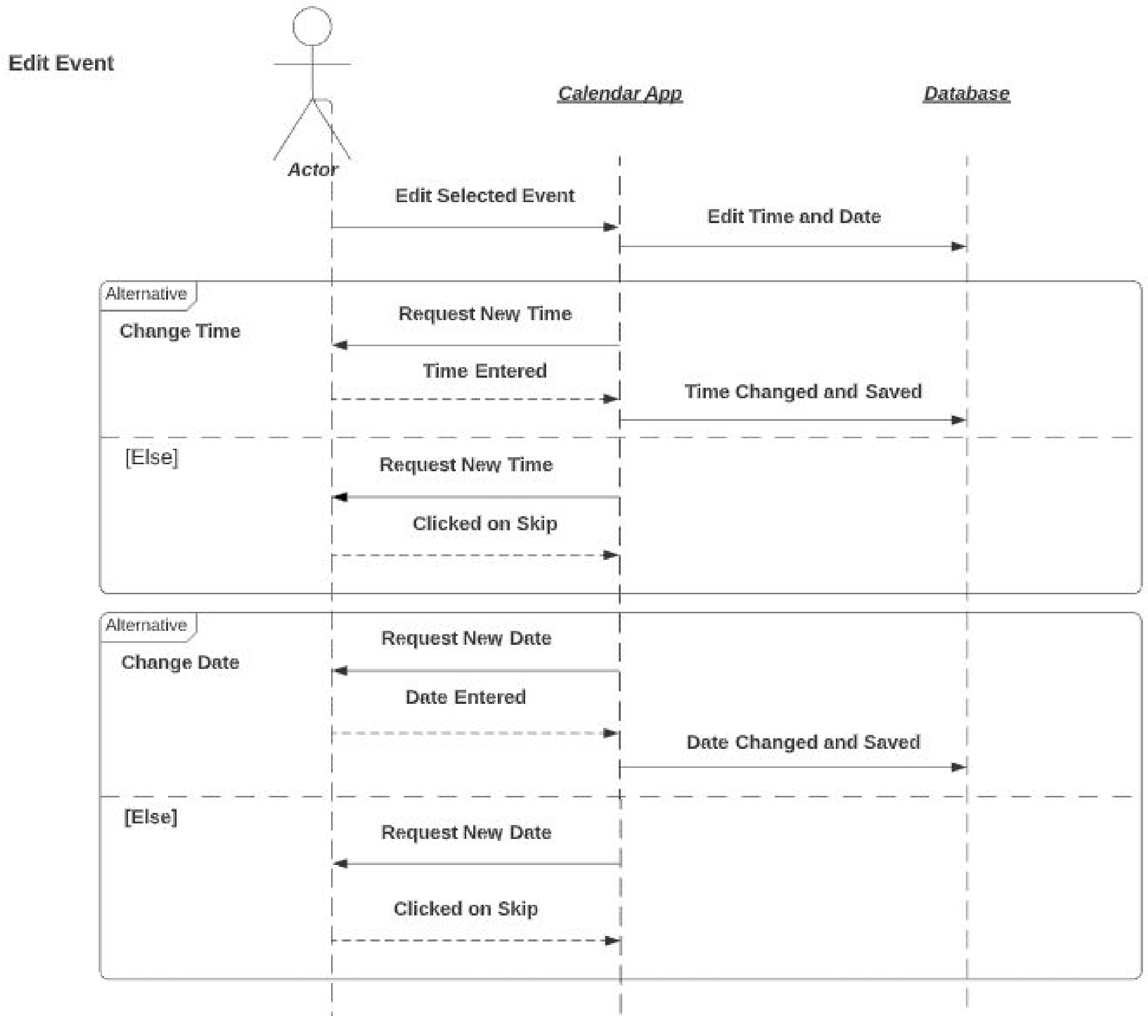


Sequence diagram – Provide sequence diagrams for each use case of your project. Please note that there should be an individual sequence diagram for each use case of your project.

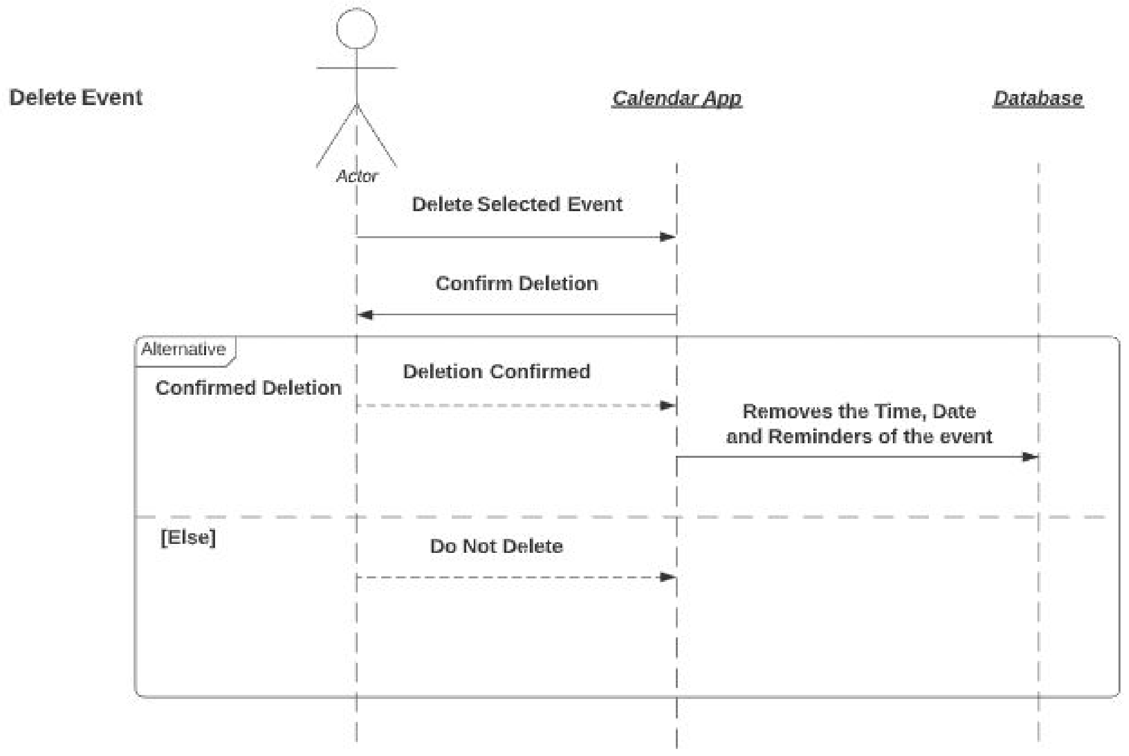
Add Event Sequence Diagram-



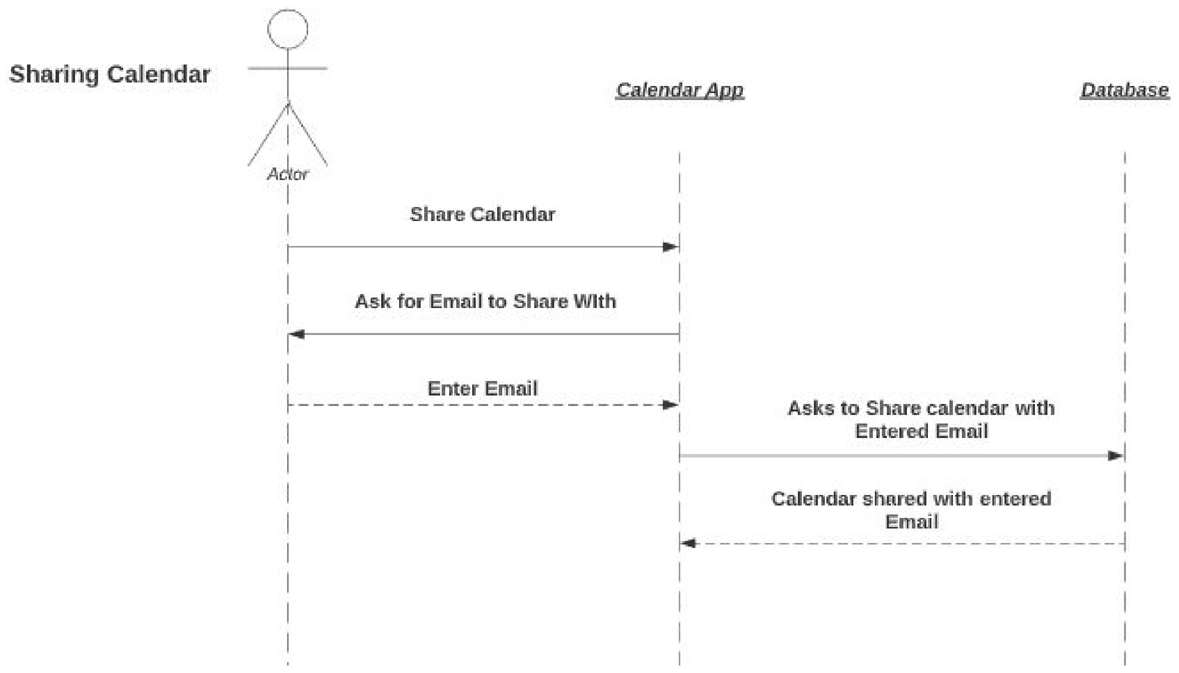
Edit Event Sequence Diagram-



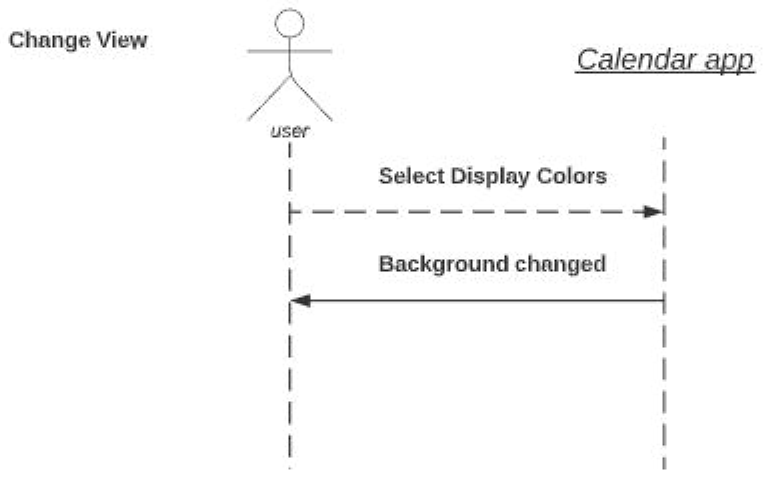
Delete Event Sequence Diagram-



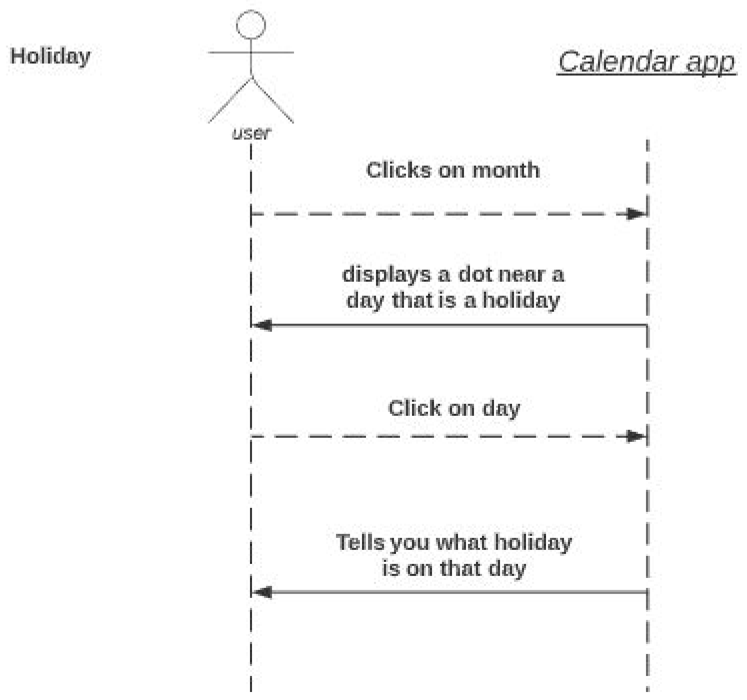
Sharing Calendar Sequence Diagram -


Sharing Calendar
Actor
Calendar App
Database
Share Calendar
Ask for Email to Share WIth
Enter Email
Asks to Share calendar with Entered Email
Calendar shared with entered Email

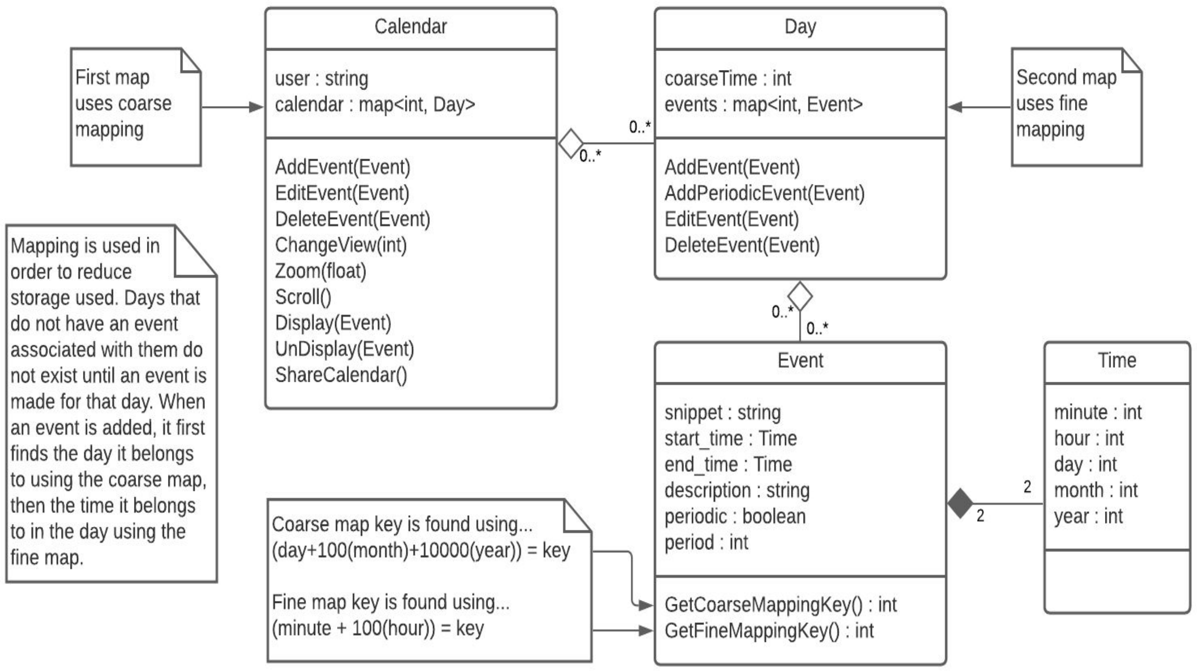
Change View Sequence Diagram -



Holidays Sequence Diagram -


Holiday
user
Calendar app
Clicks on month
displays a dot near a day that is a holiday
Click on day
Tells you what holiday is on that day

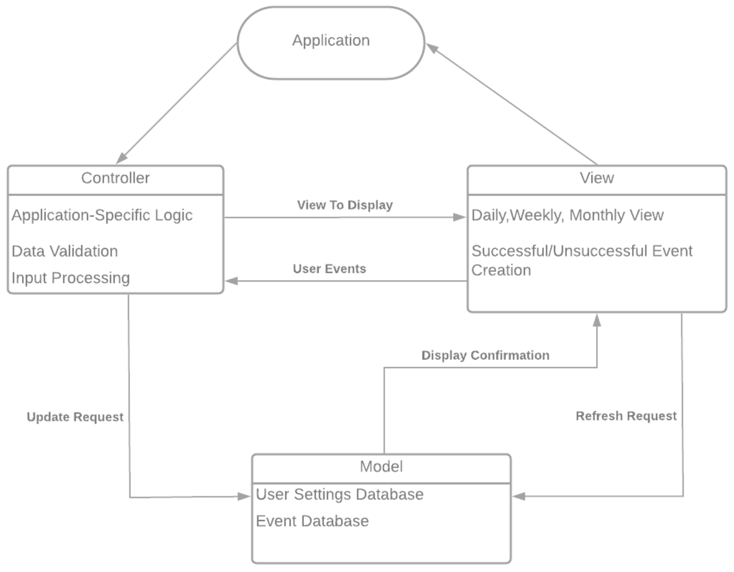
Class diagram – Provide a class diagram of your project. The class diagram should be unique (only one) and should include all classes of your project. Please make sure to include cardinalities, and relationship types (such as generalization and aggregation) between classes in your class diagram. Also make sure that each class has class name, attributes, and methods named.


Calendar
user : string
calendar : map<int, Day>
AddEvent(Event)
EditEvent(Event)
DeleteEvent(Event)
ChangeView(int)
Zoom(float)
Scroll()
Display(Event)
UnDisplay(Event)
ShareCalendar()
Day
coarseTime : int
events : map<int, Event>
AddEvent(Event)
AddPeriodicEvent(Event)
EditEvent(Event)
DeleteEvent(Event)
Event
snippet : string
start_time : Time
end_time : Time
description : string
periodic : boolean
period : int
GetCoarseMappingKey() : int
GetFineMappingKey() : int
Time
minute : int
hour : int
day : int
month : int
year : int
0..*
0..*
0..*
0..*
2
2
First map uses coarse mapping
Second map uses fine mapping
Mapping is used in order to reduce storage used. Days that do not have an event associated with them do not exist until an event is made for that day. When an event is added, it first finds the day it belongs to using the coarse map, then the time it belongs to in the day using the fine map.
Coarse map key is found using... (day+100(month)+10000(year)) = key
Fine map key is found using... (minute + 100(hour)) = key

Architectural design – Provide an architectural design of your project. Based on the characteristics of your project, choose and apply only one appropriate architectural pattern.

Model-View-Controller (MVC) pattern

As we know, the Model-View-Controller (MVC) architectural pattern separates interaction and presentation with the system data. The 3 components that make up the MVC is the Model component which manages and associate's operations to a system data, View component which defines and manages how data will be presented to the user while updating its presentation if needed, and Controller component which manages user interaction. For example, Key presses, mouse clicks etc. Moreover, it passes these interactions to the view and the model. An example of the MVC pattern can be shown through a java application by implementing the model, view, and controller into separated classes which can be used to show a graphical web-based system. The strength of this pattern is that it allows data to change independently from its representation and supports the demonstration of the same data in different ways while making changes to one representation shown in all. The consensus that our group chose to do is the MVC architecture pattern because it is superior to other models for smaller, more focused applications. More so, we agreed that the view component of this architecture is very important for monthly, weakly, and daily views. Since the code we are not going to be implementing the code, the disadvantage of this pattern of being too complex will not be a concern. Another reason is because it has high cohesion and low coupling and allows use to work on code simultaneously yet separately.



3. **Choose only one of the following two options and specify clearly which is your choice. This will help us post a live presentation schedule.**

- Option 1: Present live during scheduled class time via BBC (Blackboard collaborate). Each member of the group has to be present and participate in this option. No need to pre record your presentations if you choose option 1 as your live presentations will be recorded. Still, you should include your (non-recorded) presentation slides into your project deliverable2 submission bundle.

- Option 2: Prerecord your captioned presentations. Save your captioned recording at a URL. Then provide this URL (where you host your pre recorded presentation) exactly here, inside your Final Project Deliverable2 report so that we post it for students to access. Remember: Captioned recordings are UTD requirements. Make sure each group member talks in that recorded presentation. DO NOT SEND TO US your presentation recordings, as your captioned pre-recording file will be too large. Only provide its URL here. Still, you should include your (non recorded) presentation slides into your project deliverable2 submission bundle.

We decided as a group to go with Option 2 and pre-record our captioned presentation.
https://youtu.be/bHf-JJh_hT8

4. Project Scheduling, Cost, Effort and Pricing Estimation, Project duration and staffing: Include a detailed study of project scheduling, cost and pricing estimation for your project. Please include the following for scheduling and estimation studies:

4.1 Project Scheduling. Make an estimation on the schedule of your project. Please provide start date, end date by giving justifications about your estimation. Also provide the details for:

- **Whether weekends will be counted in your schedule or not**
- **What is the number of working hours per day for the project**

This project would take around 1 month to fully complete. If we use the FP method and then calculate the time it comes out to 1 week. The reason for extending this to a month is because we also have to test code and account for troubles we may run into along the way. Another thing we have to consider is the documentation for the code. We would be working Monday - Friday 9 pm to 5 pm, with an hour lunch break in the middle. This means if we start on december 1st we should have it done by january 5th. The reason we are going a little over a month is because we would give off from december 23rd-december 27th for christmas,and also give new years day off.

4.2. Cost, Effort and Pricing Estimation. Describe in detail which method you use to calculate the estimated cost and in turn the price for your project. Please choose one of the two alternative cost modeling techniques and apply that only:

- **Function Point (FP)**
- **Application composition**

For the final group project, we decided to use the Function Point (FP) Method to calculate the estimated cost for our software project because we felt it was the best fit for our application.

	Function Category	Count	Complexity			Count x Complexity
			Simple	Average	Complex	
1	Number of user input	10	3	4	6	30
2	Number of user output	4	4	5	7	16
3	Number of user queries	3	3	4	6	12
4	Number of data files & relational tables	7	7	10	15	105
5	Number of external interfaces	1	5	7	10	7
						GFP = 170

We discussed the various parameters and the complexity for each of the function categories, and made changes/added to the corresponding function categories that we all agreed upon and decided fits our project best. The following are the parameters for each of the specific function categories we decided and agreed upon on as a team:

User inputs: add event, change time, change date, alert, delete, share, open event, open menu, username, password
User outputs: event, confirmation, error message, options menu
User queries: change view request, history request, person information request
Data files & relational tables: 3 maps for events, 3 viewing files, 1 personal information file (privacy)
External interfaces: device screen

1.) Does the system require reliable backup and recovery? 3
2.) Are data communications required? 2
3.) Are there distributed processing functions? 0
4.) Is performance critical? 4
5.) Will the system run in an existing, heavily utilized operational environment? 2
6.) Does the system require online data entry? 4
7.) Does the online data entry require the input transaction to be built over multiple screens or operations? 2
8.) Are the master files updated online? 2
9.) Are the inputs, outputs, files, or inquiries complex? 2
10.) Is the internal processing complex? 1
11.) Is the code designed to be reusable? 2
12.) Are conversion and installation included in the design? 2
13.) Is the system designed for multiple installations in different organizations? 4
14.) Is the application designed to facilitate change and ease of use by the user? 5

GFP = 170 FP

PCA = 0.65 + 0.01(3 + 2 + 0 + 4 + 2 + 4 + 2 + 2 + 2 + 1 + 2 + 2 + 4 + 5)
= 0.65 + 0.01(35)
= 1.00

FP = GFP * PCA
= 170 * 1.00
= 170 FP

(Assumptions: 60 function points per person-week)
E = FP / productivity
  = 170 / 60
  = 2.8333
  ≈ 3 person-weeks

Team Size = 7
D = E / Team Size
  = 2.8333 / 7
  = 0.4048
  ≈ 1 week

4.3. Estimated cost of hardware products (such as servers, etc.)

Estimated cost of hardware products including app infrastructure services (servers, hosting, domains), databases used for storage such as user data, and back-end infrastructure to allow for app scalability and integration: ~$250/month

An app hosting server can range in price anywhere from $70/month - $320/month [6]. For our application which is considered a medium complexity application with an above average number of features and capabilities compared to other traditional calendar applications, the estimated costs of hardware products would be around $250/month.

4.4. Estimated cost of software products (such as licensed software, etc.)

Due to the number of free IDE's available to choose from to develop our software application on and various open-source software to use, we would not incur any additional costs for software products such as fees for using licensed software.

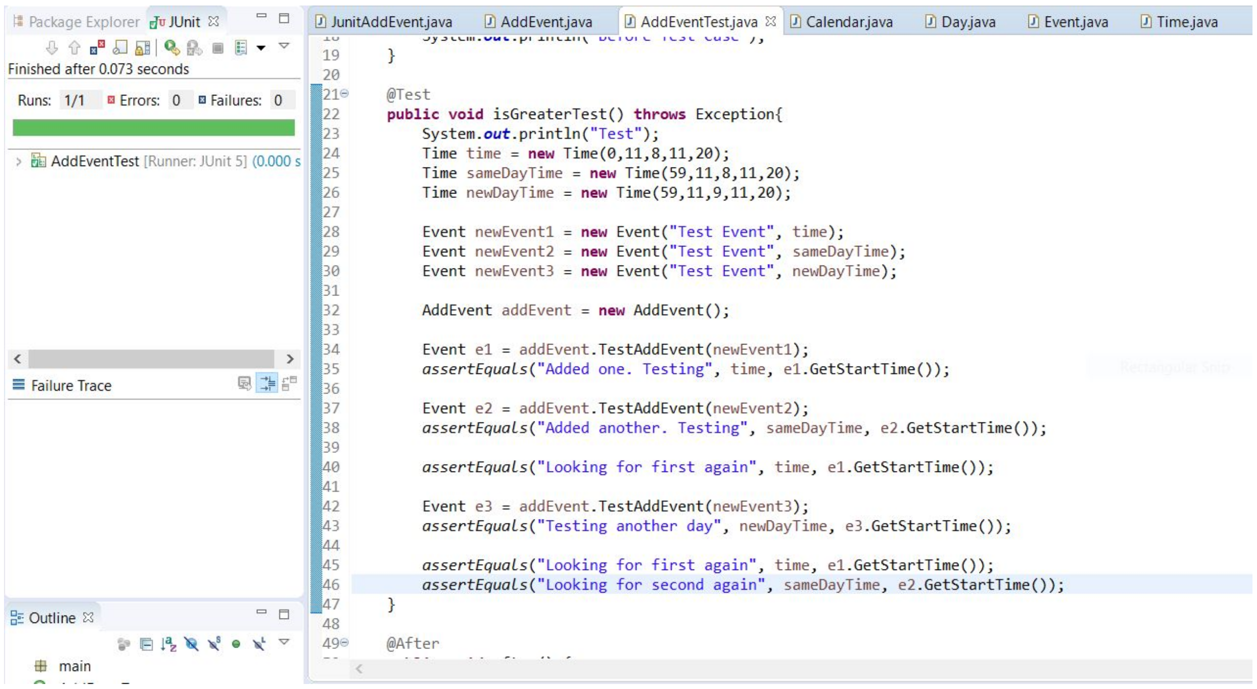
4.5. Estimated cost of personnel (number of people to code the end product, training cost after installation)

Based on details/features for a medium sized app with medium complexity, a polished UI level, email/password sign-up and Google sign-up, cloud syncing capabilities, calendaring, display of maps data/geolocation, ability for bookings within the app, social media sharing, push notifications, multilingual support, ability to connect to one or more third party services, performance monitoring, an API for others to integrate with our app, two-factor authentication [7].

Web App	iOS App	Android App
21 Designer Days (4.2 Weeks)	21 Designer Days (4.2 Weeks)	21 Designer Days (4.2 Weeks)
77 Developer Days (15.4 Weeks)	68 Developer Days (13.6 Weeks)	68 Developer Days (13.6 Weeks)
$44,100	$40,050	$40,050
		Total Cost: $124, 200

5. A test plan for your software: Describe the test plan for testing minimum one unit of your software. As an evidence, write a code for one unit (a method for example) of your software in a programming language of your choice, then use an automated testing tool (such as JUnit for a Java unit) to test your unit and present results. Clearly define what test case(s) are provided for testing purposes and what results are obtained (Ch 8). Include your test code as additional document in your zip file submitted.

For the test, we chose to implement a basic version of adding an event to a calendar. The implementation of the calendar in the final version will use hash tables for efficiency. For this example we used ArrayLists, so the time complexity for will go from O(1) to O(n), but this won't matter much for this. We tested to ensure the following worked: adding a single event, adding two events on the same day, and adding an event on different days. After each event is added, the GetEvent method is automatically called to ensure the events could be found. The calendar is set up as a list of lists (or hash table of hash tables), so testing adding on different days is essential because the performance is different. After adding new events, we verified we could still find the previous added events.



Finished after 0.073 seconds

Runs: 1/1 Errors: 0 Failures: 0

AddEventTest [Runner: JUnit 5] (0.000 s

```
    }

    @Test
    public void isGreaterTest() throws Exception{
        System.out.println("Test");
        Time time = new Time(0,11,8,11,20);
        Time sameDayTime = new Time(59,11,8,11,20);
        Time newDayTime = new Time(59,11,9,11,20);

        Event newEvent1 = new Event("Test Event", time);
        Event newEvent2 = new Event("Test Event", sameDayTime);
        Event newEvent3 = new Event("Test Event", newDayTime);

        AddEvent addEvent = new AddEvent();

        Event e1 = addEvent.TestAddEvent(newEvent1);
        assertEquals("Added one. Testing", time, e1.GetStartTime());

        Event e2 = addEvent.TestAddEvent(newEvent2);
        assertEquals("Added another. Testing", sameDayTime, e2.GetStartTime());

        assertEquals("Looking for first again", time, e1.GetStartTime());

        Event e3 = addEvent.TestAddEvent(newEvent3);
        assertEquals("Testing another day", newDayTime, e3.GetStartTime());

        assertEquals("Looking for first again", time, e1.GetStartTime());
        assertEquals("Looking for second again", sameDayTime, e2.GetStartTime());
    }

    @After
```

6. Comparison of your work with similar designs. This step requires a thorough search in the field of your project domain. Please cite any references you make.

The 4th Dimension (4D) is an intuitive calendar application that allows users to experience the same task management skills as the world's best chief administrative officers. 4D divides the user's schedule into three main categories: daily, weekly, and monthly view, including an optional view called agenda. Each view is composed of events that are compared to each other to avoid time conflicts. Although other calendar applications are very useful, our software engineering project provides users of all makes and backgrounds a comprehensive dose of organization. There are a lot of similarities and differences when we compare our software with the similar designs such as apple calendar, google calendar, outlook calendar etc.

For example, if we use a Mac calendar we can have multiple accounts on one calendar and can manage different events in the calendar even if they are in different accounts like iCloud or Google [2]. When it comes to User Interface or UI Design, apple uses color code to differentiate the work, family, or personal events [2]. It helps users to navigate and find the important event easily.

Lastly, in Mac all the information you need about an event is at your fingertips. When you add the location of your event—like the name of a restaurant—Calendar fills in the address, shows you a map and the weather, and lets you know when it's time to leave. When we compare our design with the outlook calendar, as mentioned with a Mac design, we can create appointments and events, organize meetings, view group schedules [4], and have two calendars side by side that users created and shared by other outlook users [3]. What makes our software different is users have the option to add driving routes to inform them of the best time to leave for their event. We didn't find that feature on any other design which made our design different and feasible to use for our users.

7. Conclusion - Please make an evaluation of your work, describe any changes that you needed to make (if any), if things have deviated from what you had originally planned for and try to give justification for such changes.

Overall, our project fundamentally remains the same as our original design. However, for our product to stand out from existing technologies, our team included driving routes when planning events. Although other calendars may provide warnings when events overlap, our product will ensure driving routes do not interfere with other events.

After careful consideration of our project's design, estimated cost and development time our team concluded that the project was feasible and potentially a future endeavor we may pursue.

8. References: Please include properly cited references in IEEE paper referencing format. Please review the IEEE referencing format document at the URL: https://ieeedataport.org/sites/default/files/analysis/27/IEEE%20Citation%20Guidelines.pdf). It means that your references should be numbered, and these numbers properly cited in your project report.

[1] A. Goyal, "Calendar App Development- Cost & Key Features," *Octal IT Solution*. [Online]. Available: https://www.octalsoftware.com/blog/calendar-app-development. [Accessed: 07-Nov-2020].

[2] "Calendar User Guide for Mac," *Apple Support*. [Online]. Available: https://support.apple.com/guide/calendar/welcome/mac. [Accessed: 07-Nov-2020]

[3] "Introduction to the Outlook Calendar," *Outlook*. [Online]. Available: https://support.microsoft.com/en-us/office/introduction-to-the-outlook-calendar-d94c5203-77c7-48ec-90a5-2e2bc10bd6f8.[Accessed: 07-Nov-2020.

[4] "Introduction to the Outlook Calendar," *Outlook*. [Online]. Available: https://support.microsoft.com/en-us/office/introduction-to-the-outlook-calendar-d94c5203-77c7-48ec-90a5-2e2bc10bd6f8. [Accessed: 07-Nov-2020].

[5] K. O.Copywriter, K. O., and Copywriter, "Software Development Costs and Factors Affecting Its Price," *Cleveroad Inc. - Web and App development company*, 20-Mar-2019. [Online]. Available: https://www.cleveroad.com/blog/software-development-costs. [Accessed: 07-Nov-2020].

[6] M. Lahn, "How much does a server cost for app hosting?," *Knowledge Base ServerMania*, 20-Sep-2019. [Online]. Available: https://www.servermania.com/kb/articles/server-cost-for-apps/. [Accessed: 08-Nov-2020].

[7] Oozou, *Estimate My App*. [Online]. Available: https://estimatemyapp.com/. [Accessed: 08-Nov-2020].

[8] T. Editors, "How much does software development cost?," *Thumbtack*, 26-Sep-2017. [Online]. Available: https://www.thumbtack.com/p/software-development-costs. [Accessed: 07-Nov-2020].

[9] *Understanding App Development Cost in 2020 [Full Guide]*. [Online]. Available: https://mlsdev.com/blog/app-development-cost. [Accessed: 07-Nov-2020].

9. Non-recorded (no-voice) presentation slides. No min/max number of slides enforced. Please make sure that you can complete the presentation within **15 (fifteen)** minutes. Following templates could be a good start to prepare your presentations. As each project topic is different, a variety in presentation style is expected and welcome.

- **Title of your project together with participants**
- **Objective of the project designed**
- **Cost estimation**
- **Project timeline**
- **Functional and non-functional requirements.**
- **Use case diagram**
- **Sequence diagram for a selected representative operation of the project.**
- **Class diagram**
- **Architectural design:**
  - **Model-View-Controller (MVC) pattern (similar to Figure 6.6)**
- **Preferably a demo of user interface design that shows screen to screen transitions though no full functionality is required.**

11. GitHub requirement:
Make sure at least one member of your group commits everything for project deliverable 2 to your GitHub repository, i.e.

- Your final project deliverable2 report
- Unit test code for a sample unit of your project
- Implementation code (if you have implemented your project)
- Non-recorded (no voice) presentation slides

Still, one member of your team should also submit the required project deliverable 2 materials to eLearning.

Project Submission:

Each group should designate one team member to submit his work via **eLearning** only.
PLEASE SUBMIT ONE PROJECT PER TEAM..

What to submit?

Please zip the following as one single file:

- Final project deliverable2 report (Please note that your deliverable2 report should also include your deliverable1 report as required in section 2 above.)
- Test code (section 5 above)
- Non-recorded (no voice) presentation slides (section 9 above)