<u>FOP Group Assignment 2025/26</u>

<u>Project 2: Calendar and Schedular App</u>

A. Introduction

Time management is an essential skill for students and professionals alike. Thus, people have been finding ways to organize their time and commitments in a legible way, in order to be able to optimize the limited daytime that every single person gets.

One of the popular solutions is calendar applications. Its simple interface of a calendar, and everything that is going on during a date, makes it intuitive to use and makes it benefit our daily lives. Thus, we will be building a rather barebones calendar and schedular app for this assignment.

This project will help you apply fundamentals of programming in Java, with focus on logic flow, Object-Oriented Programming, and file I/O handling. GUI integration is optional but encouraged.

B. Problem Statement

You are required to build a Calendar and Schedular App intended for personal use. The application will be able to create and manage events, produce views for the user, and perform backup and restore functionality.

You are not allowed to use a database/DBMS for this assignment. We will be keeping our data in local .csv files, with the formatting as below:

event.csv for storing all events

```
eventId, title, description, startDateTime, endDateTime
1, Assignment Meeting, FOP assignment, 2025-10-05T11:00:00, 2025-10-
05T12:00:00
```

** Note that eventId should be auto-generated by increment.*

recurrent.csv for recurring events linked by eventId

```
eventId, recurrentInterval, recurrentTimes, recurrentEndDate
1, 1d, 0, 2025-10-08
```

** This means eventId=1 recurs every 1 day until 8 October 2025 inclusive. We do not need recurrentTimes, so put a 0 here for ease of programming and parsing.*

```
eventId, recurrentInterval, recurrentTimes, recurrentEndDate
1, 1w, 3, 0
```

** This means eventId=1 recurs every 1 week for 3 times.*

C. Basic Requirements (8 marks)

Feature	**Description**	**Notes / Suggestion**
Event Creation (1 mark)	The user can create events into the calendar application. eventId should be auto-generated by increment. **The created event should persist across runs, i.e. it won't disappear after terminating/restarting the program.**	Utilize reading and writing event.csv file. Having a solid understanding of Java File I/O will help you a lot in this entire project.
Event Update & Delete (1 mark)	The user can update AND delete existing events from the calendar application.	
Recurring Events (1 mark)	The user can configure events to repeat in intervals, or times. For practical purposes of this project, don't let recurring events repeat indefinitely. *Example: repeat daily / every N days / weekly / biweekly / monthly etc., for N times, or until a specified date.*	You might need to take some time thinking about updating or deleting *recurring events,* especially when it's not the root (first) event in the series.
Backup and Restore (1 mark)	The user can create backups **into one single backup file** <u>AND</u> restore all events into the calendar from the said single backup file. This is different from just going into event.csv and other .csv files and copy-pasting it, but rather the capability of copying data, e.g., into a new install / PC, and writing into the contents of application-using .csv files.	As you will be exporting the file to a separate location, an expected output will be something like: *"Backup has been completed to <location in your PC>"* One thing to consider during import is if you want to provide the user with the option to append to existing events, or delete all and only apply restored events.

View Calendar (2 marks)	The application can display events in various forms, including calendar view (by month or week) and list view (by day, week or month) This section covers view **in CLI format.** *Some examples:*	

```
// Weekly list view
=== Week of 2025-10-05 ===
Sun 05: Assignment Meeting (11:00)
Mon 06: No events
Tue 07: Project Discussion (15:00)

// Calendar month view
Oct 2025
Su Mo Tu We Th Fr Sa
         1  2  3  4
5* 6  7  8  9  10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30 31
* 5: Assignment Meeting (11:00)
```

Event Basic Search (1 mark)	This section will implement basic searching for events **by date / custom date range**. Other advanced filters will be under "Event Advanced Search and Filter" section in additional features.	
Programming Skills, File System and Version Control System Usage (1 mark)	To get your one mark here, 1. Objects should be written in their classes and use OOP principles for coding. 2. The application can be used across different PCs without changing the code, especially file locations, while in each different PC. 3. The team demonstrates significant collaboration, and Git usage capabilities through presenting their Git commit logs, issue tracker / pull request and code review logs throughout the progress of the project.	

D. Additional Features (4 marks)

There are many customization options for you to do in this system, but here are a few suggestions for you to look at if you get stuck brainstorming for ideas. Most of these additional features give you space to express yourself and "show-off" your programming capabilities.

Do remember to code your project primarily in Java.

Feature	**Description**	**Notes / Suggestion**
Reminders / Notifications (1 mark)	The application can display event reminders or notifications, whether on realtime, or for practical purposes here, during program launch. **AND** The user is also able to set how long before an event to send a reminder/notification, eg. 30 minutes before / the day before etc.	If you opt for program launch notification, I might expect something like: *"Your next event is coming soon in <duration_to_next_event>"*
Event Advanced Search & Filter (1 mark)	You can implement more searching filters for the section here.	Some other metrics you can do include event title search, or more filters based on the Additional Event Fields you may add.
Event Statistics (1 mark)	This section will be for you to express your creativity. You may think of some insights that you are able to get from the calendar and try to calculate and visualize it. Simple examples include statistics like busiest day of the week. Do sufficient statistics to be able to gain this mark here, one or two will not suffice.	
Graphical User Interface (1 mark)	The graphical user interface, or GUI is a form of user interface that allows users to interact with electronic devices through graphical icons and audio indicators such as primary notation, instead of text-based UIs, typed command labels or text navigation. (Wikipedia)	Examples include using JavaFX or Swing.

Additional Event Fields (1 mark)	This section allows you to customize additional fields for your event that is not included in any other feature scope. The additional fields **must** be kept in a separate .csv file from the basic fields for marking purposes, and the additional fields should be searchable and able to be backed up and restored. additional.csv `eventId, field1, field2, field3`	Some additional fields might include location, attendees, category etc.
Conflict Detection (1 mark)	The application can ensure that your scheduled events do not clash with each other in case you should really not be clashing them, like meetings or having dinner at two different places.	

E. Tips and Tricks

1. Version control systems
   Use this project as your prime opportunity to master Git, including making mistakes and trying to fix them. **Start doing your assignment early,** so you have a lot of learning leeway instead of resorting to AI tools towards the end. **The goal of this assignment is to hone your programming skills,** not the AI's.
2. LLM / Coding copilot usage
   Yes, everyone is using them (including me). **How you use it is key:** If you use it as a substitute for you coding, you will be substituted by them in the future when you don't have the skills to design a solution due to blindly copy-pasting the questions into your copilot. If you use it as your handy tool, your coding capabilities will be enhanced by your copilot while you learn and retain programming knowledge. **Ensure you know what you are trying to do** / design your solution before asking AI to generate the syntax for you. **Understand what is the AI generating:** ask it if you don't understand so you learn along the way.
3. Modularity
   Writing all code in one file is messy, convoluted, unreadable, and unmanageable in the long run. Write separate classes in separate files.
4. File I/O references
   Don't put full file locations (i.e. from C:\Users\user\.....). Use relative file paths so that the external files can be used on multiple devices. What are the chances that everyone's stuff is on C:, and their PC username is the same as yours?

F. Contact Me and my last words

For any questions or clarifications, contact me (Jonas Chuan) via WhatsApp 019-5187978 or e-mail at jc.chuan.0303@gmail.com . This assignment is aimed at ensuring mastery of core concepts of flow control, File I/O and Object Oriented Programming. I hope you guys are able to explore and have a good study of programming fundamentals and be a better programmer by the end of this assignment.