Project Portfolio for COMPal - Yue Jun Yi

About this Project

My team and I were tasked with enhancing a simple **Personal Assistant**, Duke, which helps a user to manage tasks via a **Command-Line Interface**. We decided to morph it into a calendar application called **COMPal**, which targets NUS students who prefer to use a desktop application to manage their busy student lives.

COMPal integrates the existing **Command-Line Interface (CLI)** mode of interaction with a sleek and clean **Graphical User Interface (GUI)**. This caters to most students' preference to type fast, and also presents their tasks in a more intuitive and user-friendly way.

We have added the following features to **COMPal**:

- Simple commands for easy task management, such as addition of tasks and setting of reminders
- A system of flexible and intuitive keywords to have more control over the user's schedule, such as priority levels, start and end times
- A pleasant interface to view schedule for a particular day/week/month.

This is what our project looks like:

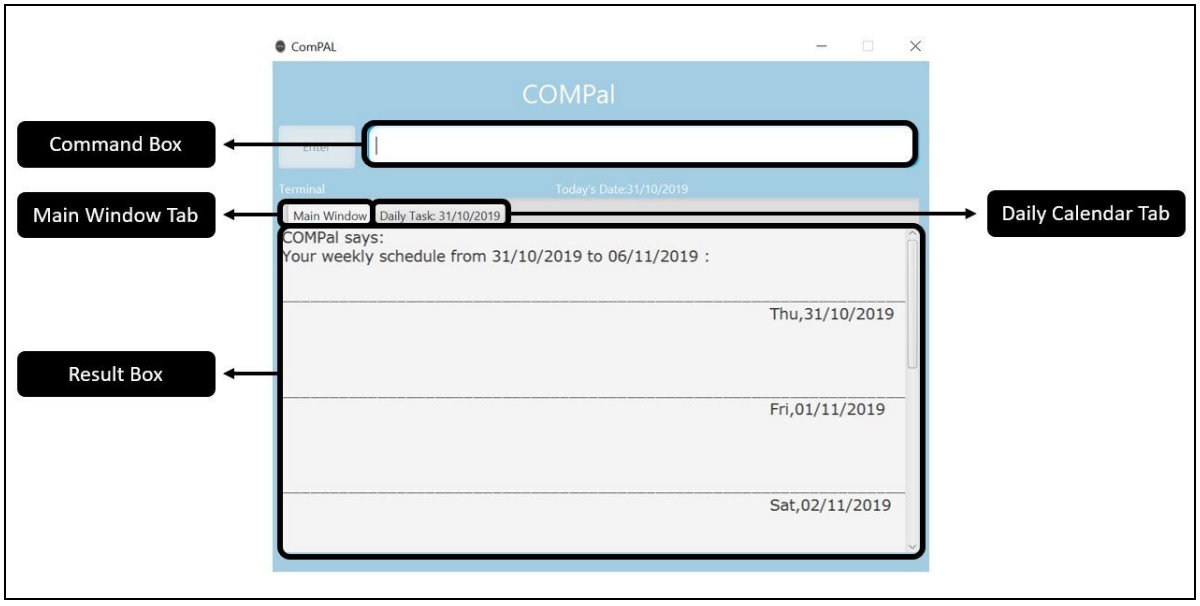

Figure 1: The GUI for **COMPal**.

Project Portfolio for COMPal - Yue Jun Yi

Summary of Contributions

This section will list my coding, documentation, and other helpful contributions to the team project.

Enhancements added: I was heavily involved in the **Task Management** feature of **COMPal**.

- **COMPal** manages two types of **tasks**: **deadlines**, which have an **end date** and an **end time**, and **events**, which have to be done during a **fixed duration** on a **specific date**.
- I developed the system of parameter keywords for **events**. As it shared many similarities with **deadlines**, I worked closely with my teammate in charge of **deadlines**, Liu Peize.
- `event`: The `event` command allows the user to add an **event**. This command has the following keywords for the user to control the characteristics of the **event**:
    - `/start`: This keyword allows the user to enter the start time of the **event**.
    - `/end`: This keyword allows the user to enter the end time of the **event**.
    - `/priority`: This keyword allows the user to assign the priority of the **event**, with acceptable values of `low`, `medium`, and `high`.
    - `/final-date`: This keyword allows the user to add multiple **events** that happen at regular intervals. The final date that the user specifies after this keyword is the final possible date of the final iteration of the user's **event**.
    - `/interval`: This keyword allows the user to specify the interval of days in between each recurring **event**.
    - `/date`: This keyword allows the user to specify the date that Event is occurring on.

Code Contributed:

- Please click this link to view my Contributed Code
- Please click this link to view my Contributed Pull Requests

Other Contributions:

- Project Management:
    - I vetted our Developer Guide for submission.
- Enhancements to existing features
    - The initial structure of our project was not Object-Oriented. I collaborated with my teammates in restructuring our code (pull request #84)
    - We implemented more structural refinements later on (pull request #159, collaborating on another teammates computer)
    - The original `/date` keyword from **Duke** only took in a single date. I modified it to take in multiple dates, so that the user can enter multiple **events** that do not occur at regular intervals.
    - I changed the `/priority` keyword to an optional keyword, so that users can automatically assign a priority of `low` when they enter their **event** without that keyword.

Project Portfolio for COMPal - Yue Jun Yi

- Documentation
    - I created our project's README page to introduce potential users to our product with a more personal touch. (README.md)
    - I updated all documentation related to the features I am in charge of. These can all be found in the **User Guide** and **Developer Guide** sections below.
- Community
    - The pull requests that I reviewed from my teammates can be viewed here: Pull Requests

Throughout this **Project Portfolio**, there will be various icons as described below:

Icon	Description
ℹ	Additional important information about a term/concept
💡	A tip that can improve your understanding about a term/concept
⚠	A warning that you should take note of

Project Portfolio for COMPal - Yue Jun Yi

Contributions to User Guide

These are the sections that I am in charge of in the User Guide:

- 4.1.2. Deleting a Task: `delete`
- 4.2.1. Deadline Management
- 4.2.2. Event Management

I will only include part of Section 4.2.2. on Event Management (up to Section 4.2.2.1.1.) and remove all page breaks, to stay under the page limit.

4.2.2. Event Management

Your best friend's birthday party. Your sibling's graduation. Your cousin's wedding.

Your 8 am Lecture. Your 8-hour code sprint for your software development project. Your compulsory torture session in the frigid exam hall.

Your student life is an endless merry-go-round of things to do - some joyful, some agonising. To better manage your time, you can set a preferred duration for each **task** you have to accomplish, and let **COMPal** track them as **events**. In this section, you will be introduced to commands and parameters that help you manage these **events**.

Below is a list of **parameters** and **keywords** that you can expect to use for the commands in this section.

Table 2: Parameters and **keywords** for **events**.

Keyword	Parameter	Usage
-	`DESCRIPTION`. All characters except underscores can be used.	You can **describe** your **task** in any detail. No keyword is required to be typed before your input - just describe your **event**!!
`/date`	`DATE`, in the format `DD/MM/YYYY`	You can enter the ***date*** that your **event** is happening ***on***
`/start`	`START_TIME`, in the format `HHmm`	You can enter the ***time*** that the **event** is starting ***at***
`/end`	`END_TIME`, in the format `HHmm`	You can enter the ***time*** that the **event** is ending ***at***

`/priority`	`PRIORITY` (`low`, `medium`, `high`)	You can assign a ***priority*** to multiple/single **event(s)**.
`/final-date`	`FINAL_DATE`, in the format `DD/MM/YYYY`	You can use this to add ***multiple* events** that occur at ***regular intervals***. `FINAL_DATE` will be taken as the latest possible date for your ***final* event**.
`/interval`	`INTERVAL`, positive number greater than `zero`	You can use this to add ***multiple* events** that occur at ***regular intervals***. `Interval` will be taken as the interval between each recurring **event**, in terms of the number of days.

i	`/final-date`, `/interval` and `/priority` are ***optional keywords***. You can use them for more control over your **tasks** but can leave them out if you want to. **COMPal** will then revert to ***default values***, which will be specified below.

⚠	Any dates that you enter ***has*** to be in the format `DD/MM/YYYY`, or **COMPal** will not understand your dates!
⚠	Any time that you enter ***has*** to be in the format of `HHmm`, or **COMPal** will be confused!

4.2.2.1. Adding Events

You can use the `event` command to get **COMPal** to add impending **events** to its impressive memory, and keep track of them for you. You'll never miss an **event** this way!

Basic Command Format:

```
event DESCRIPTION /start START_TIME /end END_TIME /date DATE
```

Example:

- `event Dance Practice /start 1800 /end 2000 /date 02/10/2019`

  Adds an **event** with `Dance Practice` as `DESCRIPTION`, `1800` as the `START_TIME`, `2359` as the `END_TIME` and `02/10/2019` as the `DATE`.

- `event Late Night Study Session /start 2200 /end 0100 /date 02/10/2019`

  Adds an **event** with `Late Night Study Session` as `DESCRIPTION`, `2200` as the `START_TIME`, `0100` as the `END_TIME` and `02/10/2019` as the `DATE`.

Note that in this case, the absolute value of the `END_TIME` is ***not after*** the absolute value of the `START_TIME`. **COMPal** interprets this as your Late Night Study Session starting at 2200 on 02/10/2019, and ending at 0100 on 03/10/2019 (the following day).

⚠	An **event** has a maximum duration of 24 hours, i.e. if you enter the same value for both `START_TIME` and `END_TIME`, **COMPal** interprets it as a 24-hour long event. E.g. `event LAN Party /start 2300 /end 2300 /date 02/10/2019` means that your LAN Party starts at 2300 on 02/10/2019 and ends at 2300 on 03/10/2019 (the following day).

However, the above is merely the ***basic format***. As students, we have to juggle schoolwork, friends, family, and perhaps even a side job. As our lives become increasingly hectic, we invariably have to pick some **events** to prioritise above others. To handle these concerns, **COMPal** lets you assign ***priorities*** and also create ***recurring* events**.

4.2.2.1.1. Assigning Priorities to Events

The optional `/priority` keyword lets you assign an **event** with a `PRIORITY`. If you have an **event** that you ***absolutely have*** to complete, you can enter a `PRIORITY` of `high`. If your **event** isn't that urgent, you can use the value of `medium`, or if it isn't something worth worrying about, you can assign it as `low`.

Alternatively, if you do not use the `/priority` keyword, **COMPal** will set the `PRIORITY` of your **event** as `low` by ***default***.

Command Format (Priority):

```
event DESCRIPTION /start START_TIME /end END_TIME /date DATE /priority PRIORITY
```

Examples:

- `event Netflix and Chill /start 2300 /end 0200 /date 02/10/2019 /priority high`

  Adds an **event** with `Netflix and Chill` as `DESCRIPTION`, `2300` as the `START_TIME`, `0200` as the `END_TIME`, `02/10/2019` as the `DATE`, and `high` as `PRIORITY`**.**

- `event Birthday Bash /start 1800 /end 2300 /date 02/10/2019 /priority medium`

  Adds an **event** with `Birthday Bash` as `DESCRIPTION`, `1800` as the `START_TIME`, `2300` as the `END_TIME`, `02/10/2019` as the `DATE`, and `medium` as `PRIORITY`**.**

- `event Study Session /start 1900 /end 2300 /date 02/10/2019 /priority low`

Adds an **event** with `Study Session` as `DESCRIPTION`, `1900` as the `START_TIME`, `2300` as the `END_TIME`, `02/10/2019` as the `DATE`, and `low` as `PRIORITY`.

- `event Study Session /start 1900 /end 2300 /date 02/10/2019`

  This adds an **event** with `Study Session` as `DESCRIPTION`, `1900` as the `START_TIME`, `2359` as the `END_TIME`, `02/10/2019` as the `DATE`.

  However, omitting the `/priority` keyword prompts **COMPal** to automatically assign `low` as `PRIORITY`, meaning that the result of this command is ***identical*** to the previous command. This example illustrates the ***optional*** nature of the `/priority` keyword.

Contributions to Developer Guide

These are the sections that I am in charge of in the Developer Guide:

- Section 3. Setting Up
- Section 4.4. Logic Component
- Section 5.2. Task Management

I will only include Section 4.4. Task Management. All page breaks are removed, to stay under the page limit.

4.4. Logic component

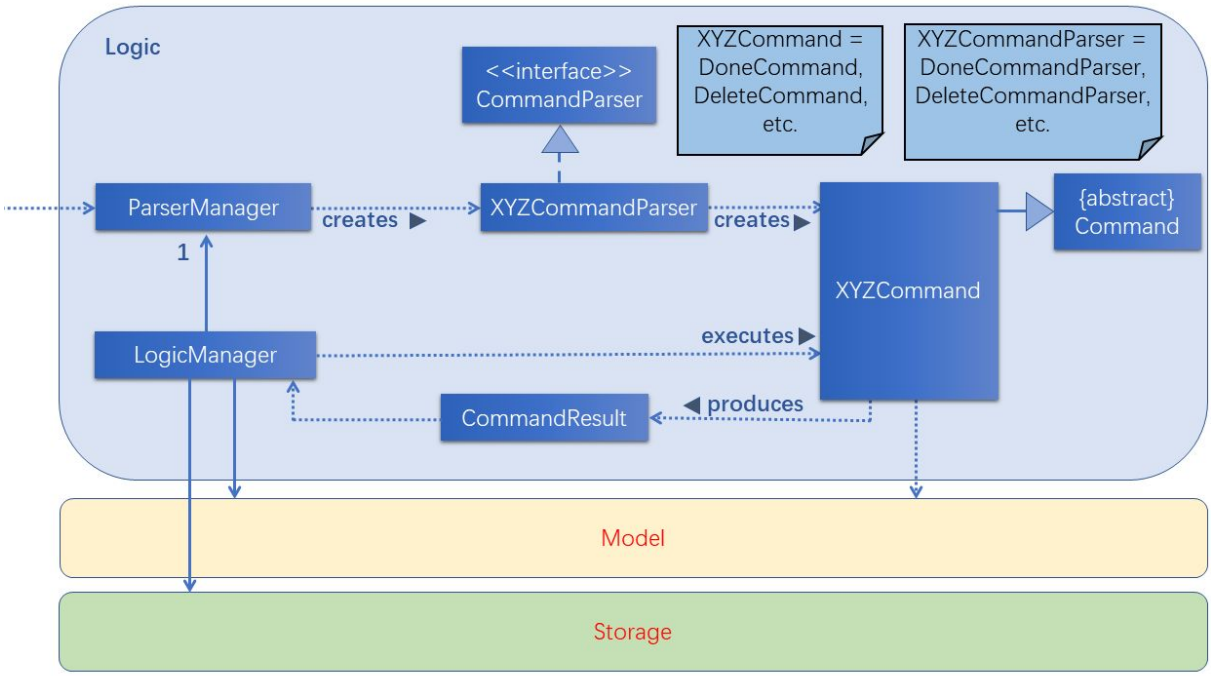

Figure 5. Structure of the **Logic** component

Project Portfolio for COMPal - Yue Jun Yi

API for `LogicManager`: `LogicManager.java`

API for `ParserManager`: `ParserManager.java`

API for `CommandParser`: `CommandParser.java`

API for `Command`: `Command.java`

The **Logic** component handles the parsing of user input and interacts with the **task** objects.

1. Uses the `CommandParser` class to parse user input.
    a. This results in a `Command` object which is executed.
2. The execution of `Command` can affect a **task** object (e.g. adding a **task** to the `TaskList`)
3. The result of the `Command` execution is encapsulated as a `CommandResult` object which is passed to the UI to be rendered as output for the user.

Given below is the Sequence Diagram for interactions within the **Logic** Component for the `logicExecute` (`delete /id 1`) API call.

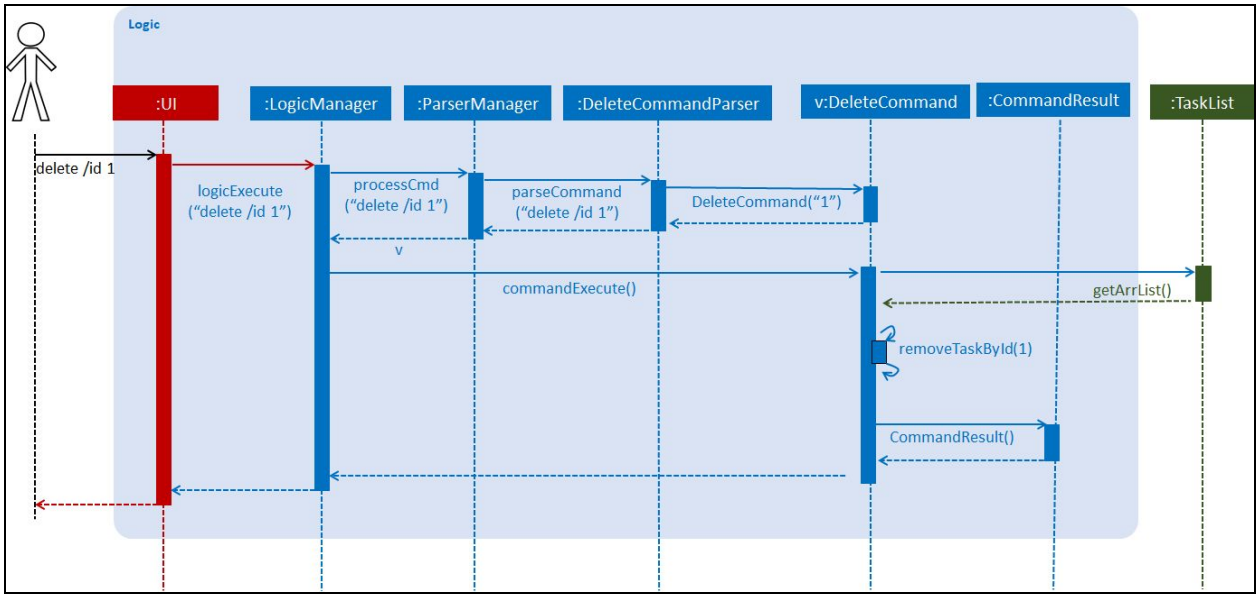

Figure 6: Interactions inside **Logic** Component for the `delete /id 1` command