Lee Yueyu – Project Portfolio for Gazeeebo

Introduction

Overview of project

My group of five computer engineering students were tasked with either enhancing or morphing a desktop personal assistant command line application called Duke. We chose to enhance the application into an integrated student planner, called Gazeeebo, for National University of Singapore (NUS) computer engineering (CEG) students to plan their student life. It enables them to plan their modules for each semester during their four years in NUS; keep track of their progress in completing a specialisation; calculate their cumulative average point (CAP); it also provides information on module prerequisites to aid them in their planning.

This is what the main page of our project looks like after entering the correct password to enter the application:

```
Welcome to Gazeeebo
_____

  __   __   __   __   __   __
 |    |  |  |   / |    |    |    |  \  |    |
 |  _ |_ |  | /  |__ |__ |__ |__| |    |
 |__| |    | /_ |__ |__ |__ |__/ |__|

_____

Today is Sunday, 10 November 2019
Upcoming deadlines:
Upcoming events:
1.[E][ND]eat(at:12 Dec 2019 03:03:03-04:04:04)

Content Page:
------------------
1. help
2. contacts
3. expenses
4. places
5. tasks
6. cap
7. spec
8. moduleplanner
9. notes
10. change password
To exit: bye
```

Figure 1. The user interface of Gazeeebo

My role

My role in the project was to implement and write code for the schedule feature, which lists out the user's schedule for a particular day, week or month and their notes for that period; note page feature, where users can record their goal and information about their modules such as assessment details; detecting scheduling clashes for events feature; and the help feature, which provides help to users. The following sections will provide more detail about these features and the documentation I have written for them. There will also be details about my other contributions to the project such as project management.

Mark ups used in the document

Grey highlight	Words that are highlighted in grey are classes and components.
`Courier New font`	This font is used for commands to be keyed into the command line of the product.

Summary of contributions

This section provides a summary of my contributions to the project in terms of code, documentation and other useful contributions.

Enhancements

Feature 1: List out the user's schedule

→ *What it does:* This feature lists out the user's schedule for a particular day, week or month.

→ *Reason for this feature:* It is difficult to see all the tasks for a particular period if there is a long list of tasks and sometimes users might want to do that. This feature makes it easy for them to do so.

→ *Highlight:* The weekly period is especially useful for the target users, which are students, as NUS CEG students' timetable is by the week. This means that the time period of a week is significant for them.

→ *Links with other features:* This feature is linked to Feature 2. When the user types in the command to use this feature, the notes for the specified period is also shown to the user.

Feature 2: Notes linked to a particular period

→ *What is does:* This feature enables users to add, edit, delete and list out their notes for a day, week or month.

→ *Reason for this feature:* This feature enables users to link noteworthy information that they want to record to a specific time period and it also can serve as a digital diary.

→ *Highlight:* Same as Feature 1.

→ *Links with other features:* This feature is linked to Feature 1 as explained above.

Feature 3: Note page

→ *What is does:* This note page enables users to record their goal and information about their modules. A sub feature of this is the module page where users can view their notes for that module.

→ *Reason for this feature:* Students are likely to need to record important information about modules so that they can take note of what needs to be done for each module.

→ *Highlight:* There is a special section for users to record their assessments in terms of the name of the assessment and its weightage. This is likely to be my biggest feature.

Feature 4: Help for users

→ *What is does:* It provides users with help in the form of a help page with all the commands and also allows users to call out help for specific pages in our application.

→ *Reason for this feature:* Users might need help when using the product if they encounter problems. This feature makes it convenient for them to get help without having to look for the user guide and without requiring internet connection to find the user guide.

→ *Links to other features:* This help feature can be called anywhere in the application.

Feature 5: Detect schedule clashes for events

→ *What is does:* Tells users if the event they are adding clashes with an existing event they have.

→ *Reason for this feature:* To alert users to clashes in events so that they can re-plan their schedule if necessary.

→ *Links to other features:* Works together with the Event Feature (implemented by my other team mate) by increasing the usefulness of the Event feature.

Code contributed: [code]

Other contributions

- Project management: I setup our GitHub repository and helped to manage the GitHub issue tracker by creating milestones and setting deadlines for them. Also, I ensured that the naming of weekly deliverables was correct. Additionally, I ensured that the user interface (UI) image on my group's GitHub page was up to date.
- Developer guide: I improved the format of the developer guide to improve its aesthetics.
- Adopted idea: I proposed to switch from the Date class to LocalDate and LocalTime classes to store date and time data. This was adopted and used across the whole project.
- Releases: I managed releases 1.2.1 to 1.3.1.

Contributions to the User Guide

Table of Sections Contributed to the User Guide

Section number	Name of Section
2	Quick Start
3.1	Viewing help
3.2.3.1	Detect scheduling anomalies
3.2.8	Viewing the schedule for a particular period
3.2.28	Note section for a particular period
3.4	Note page

Command Format

Words in UPPER_CASE are the parameters to be supplied by the user e.g. in todo TASK_DESCRIPTION, TASK_DESCRIPTION is a parameter which can be used as todo eat.

Purpose of Section

The following are excerpts from the documentation that I wrote for the user guide, which showcase my ability to write explanatory text for end users. For the full documentation, please refer to my group's user guide.

Part of the Note Section Documentation

3.2.28. Note section for a particular period

    3.2.28.1 Adding a note to a particular day/week/month: addNote

        Adds a note to the note section of the specified day, week or month.

Format (day): `addNote day YYYY-MM-DD`
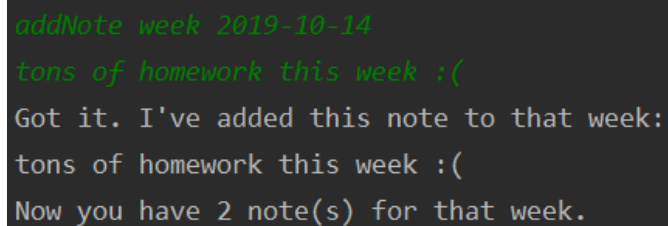
Format (week): `addNote week YYYY-MM-DD`

Format (month): `addNote month YYYY-MM`

Format details:

- The date specified in Format (week) above has to be a Monday.

Steps for adding a note (this example is for a particular week):

1. Type in the command in the format specified above.
   a. Eg. `addNote week 2019-10-14`
2. Press ENTER.
3. Type in the note you want to add on that new line.
   a. Eg. `tons of homework for this week :(`
4. Press ENTER.
5. An example is shown in the figure below.


```
addNote week 2019-10-14
tons of homework this week :(
Got it. I've added this note to that week:
tons of homework this week :(
Now you have 2 note(s) for that week.
```

Figure 3.2.28.1.1 Adding a note to a particular week

Examples for command format:

- Adding a note to a particular day:
  - `addNote day 2020-10-11`
- Adding a note to a particular week:
  - `addNote week 2019-10-14`
- Adding a note to a particular month:
  - `addNote month 2019-01`

Contributions to the Developer Guide

Table of Sections Contributed to the Developer Guide

Section number	Name of Section
3.2	UI Component
3.4	Command Component
3.5	Storage Component
4.8	Note Page Feature
4.9	Help Feature
5	Documentation
Appendix B 14	Deleting an assessment from a module in the module page
Appendix B 15	Adding a module to the note page
Appendix B 16	Editing the goal in the note page
Appendix E.1.	Launch
Appendix E.2.	Note Feature
Appendix E.2.	Note Section for a Particular Period Feature

Purpose of Section

The following are excerpts from the documentation that I wrote for the developer guide, which showcase my ability to write technical documentation and draw technical diagrams. For the full documentation, please refer to my group's developer guide.

Help Feature Documentation

4.9 Help feature

4.9.1 Implementation

The help feature is implemented by HelpCommand and HelpText. HelpText stores the data required for the help feature. These data include descriptions of the various commands as well as information on how to interpret the syntax in the help page. HelpCommand deduces what help information has been requested by the user and shows the correct information to the user.
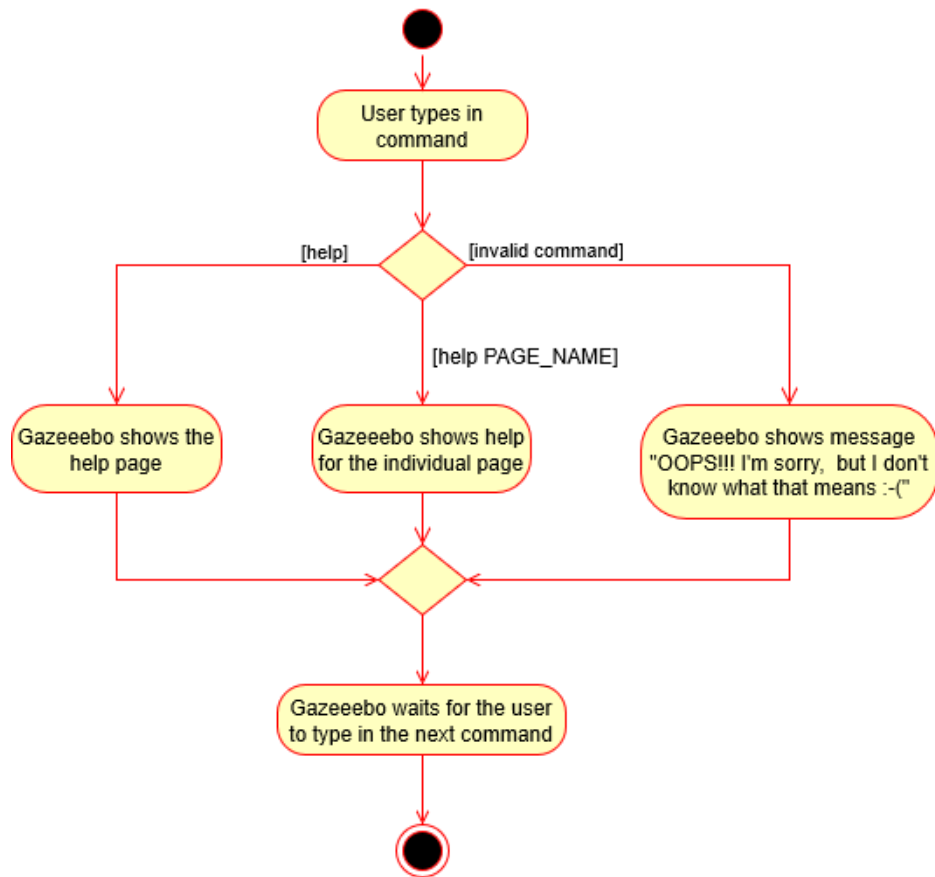
Figure 4.9.1.1 Activity diagram for Help Feature

4.9.2 Design Considerations

Aspect: Data structure to store the descriptions of the commands.

- Alternative 1 (current implementation): Store the description of each individual command as a String without any container.
  - Pros: Fast retrieval of the descriptions of each command as there is no need to search for the required description.
  - Cons: The code becomes long as the code to enumerate all the command descriptions is long.
- Alternative 2: Use a HashMap to store the description of the commands as objects with the command name as the key.
  - Pros: Searching for the description of the command is fast in O(1) and code to enumerate the container is short.
  - Cons: Order of the commands in the help page might not be the same over time as the order of the map is not guaranteed to be the same over time.
- Alternative 3: Use a TreeMap to store the description of the commands as objects with the command name as the key.

- Pros: The order of the commands in the help page will not change over time and code to enumerate the container is short.
- Cons: Retrieval of a command description is the slowest out of all the 3 alternatives in O(log n), where n is the number of commands in the TreeMap.

Aspect: The storage location of the descriptions of the commands.

- Alternative 1: Store the descriptions in a text file.
  - Pros: Good organisation of program components as all data would be stored in the same place.
  - Cons: Program slows down as reading data from the text file takes up time.
- Alternative 2 (current implementation): Store the descriptions directly in a class in the Java file.
  - Pros: Program is faster as time required to read from a text file is eliminated.
  - Cons: Poor organisation of program components as data is mixed into the source code.