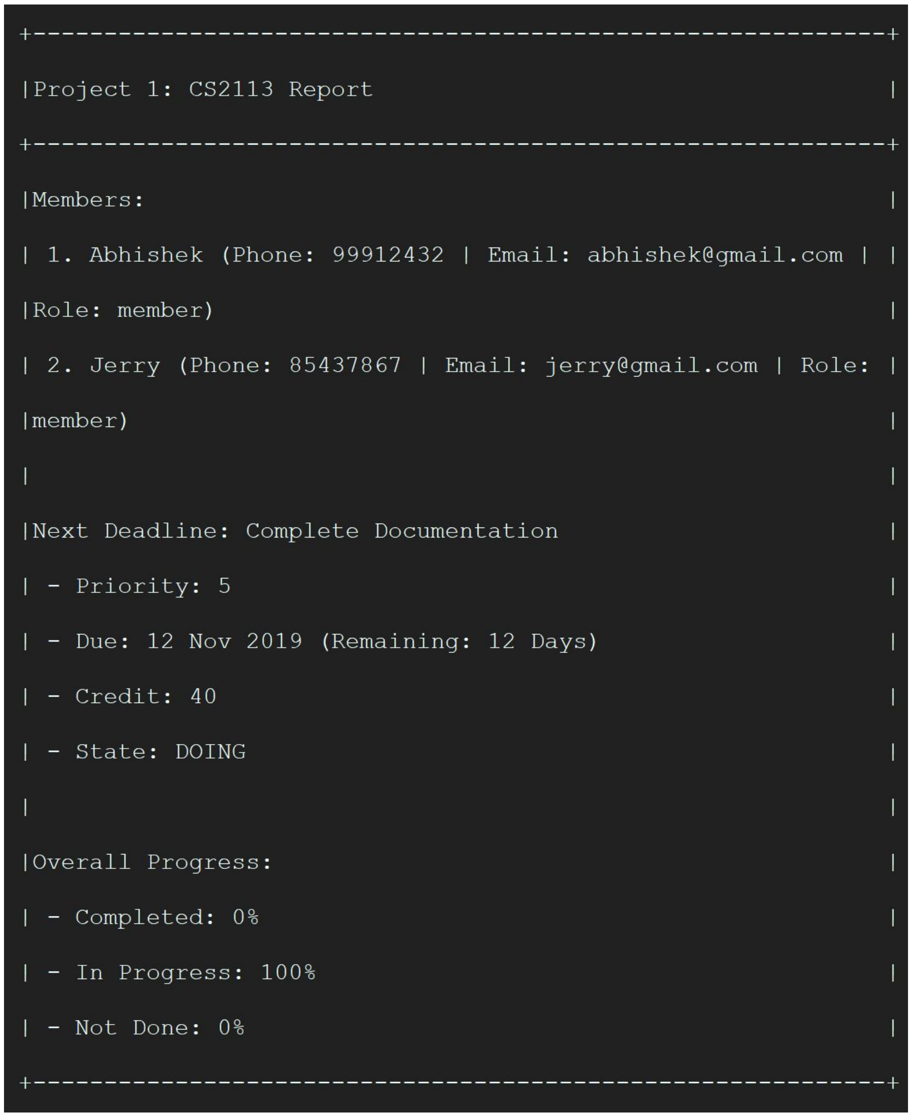
ArchDuke Project Manager - User Guide

By: Team Technical Difficulties Since: 30 September 2019 License: MIT

1. Introduction

Ever wondered how good it will be to have a Kanban board for all your projects instead of using multiple Kanban boards for different group projects? ArchDuke is the application for you! ArchDuke is a simple **desktop application** designed by team leaders, for team leaders to **manage their multiple projects**! ArchDuke is **optimized for team leaders who wish to work using a Command Line Interface (CLI)** while still being able to view certain features such as a calendar through a Graphical User Interface (GUI).

Overview of ArchDuke's User Interface



```
+----------------------------------------------------------+
|Project 1: CS2113 Report                                  |
+----------------------------------------------------------+
|Members:                                                  |
| 1. Abhishek (Phone: 99912432 | Email: abhishek@gmail.com | |
|Role: member)                                             |
| 2. Jerry (Phone: 85437867 | Email: jerry@gmail.com | Role: |
|member)                                                   |
|                                                          |
|Next Deadline: Complete Documentation                     |
| - Priority: 5                                            |
| - Due: 12 Nov 2019 (Remaining: 12 Days)                  |
| - Credit: 40                                             |
| - State: DOING                                           |
|                                                          |
|Overall Progress:                                         |
| - Completed: 0%                                          |
| - In Progress: 100%                                      |
| - Not Done: 0%                                           |
+----------------------------------------------------------+
```

2. Quick Start

1. Ensure **Java 11** is installed in your computer. To check the version of Java installed in your work environment, type `java -version` into your preferred terminal (such as Command Prompt, Git Bash, or MacOS terminal)
2. Download the latest **.jar** file from here.
3. Copy the **.jar** file to your desired folder.
4. Run the file using `java -jar ArchDuke.jar`.
5. Type your commands in the command box at the very top and press Enter to execute it.
    a. Typing `help` and executing it will open the help window for a complete overview of all available commands, their usage and their functions.
6. Refer to the **Features** section for detailed documentation and usage of each command
7. Refer to **Command Summary** section for a quick summary of all available commands.

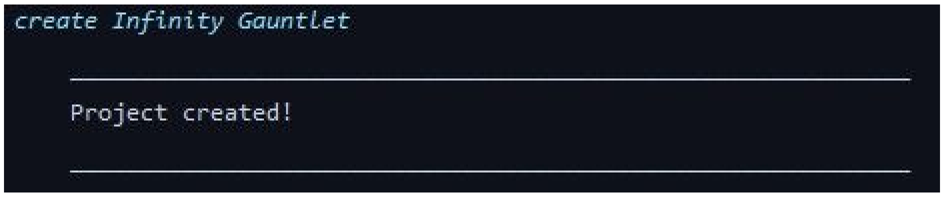
3. Features

Command Format

- Words in `UPPER_CASE` are the parameters that needs to be specified by each user.
  E.g. `create PROJECT_NAME` where `PROJECT_NAME` is a parameter which can be used as `create CS2113_Project`
- Parameters can be typed in any order
  E.g. if the command specifies `-n NAME -d DATE`, `-d DATE -n NAME` is acceptable as well
- Parameters defined in square brackets are optional.
  E.g. `-n NAME [-i PHONE_NUMBER]` can be used as `-n Cynthia -i 91234567` or as `-n Cynthia`
- `-n` indicates a name or string flag
- `-e` indicates a flag for email
- `-a` indicates a flag for an address of a member
- `-i` indicates an integer flag
- `-d` indicates a flag for a date in the DD/MM/YYYY format
- `-p` indicates a flag for task priority, represented in integers.
- `-c` indicates a flag for the amount of credit awarded for completion of a certain task, represented in integers.
- `-to` indicates the flag for the index of the member to assign the selected task to
- `-s` indicates the flag where user can specify the state of the specific task

3.1. Create a project : `create`

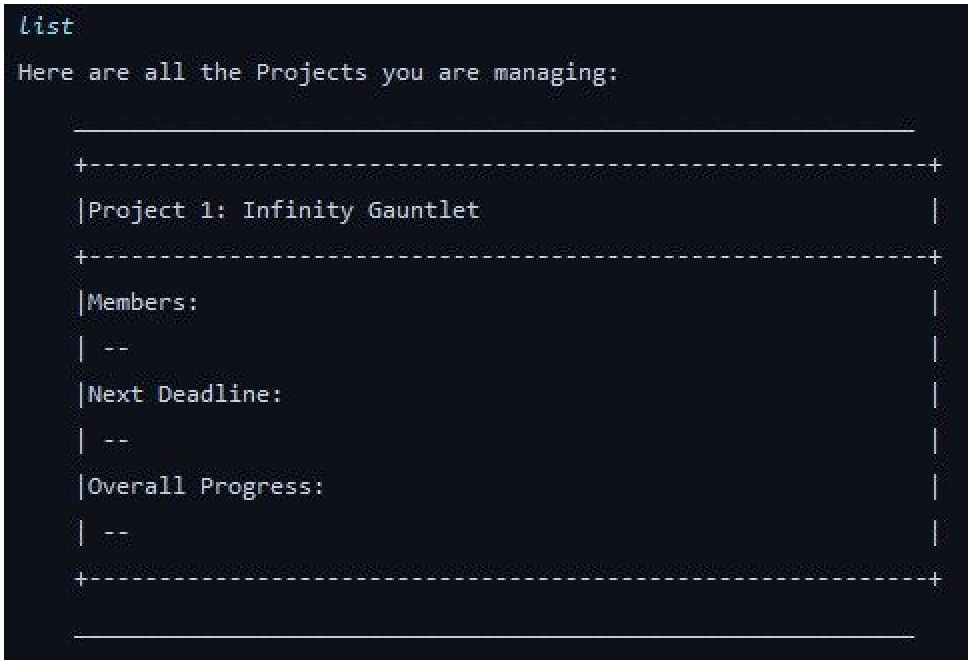
This command will create a new project with a given name, which is automatically saved in JSON.
Format: `create PROJECT_NAME`

Example: `create Infinity Gauntlet`


```
create Infinity Gauntlet

    ____________________________________________________________

    Project created!

    ____________________________________________________________
```

3.2. View all projects : `list`

Shows a list of all projects and project details: project index number, project name, the names of all project members, nearest deadline and overall progress of project
Format: `list`

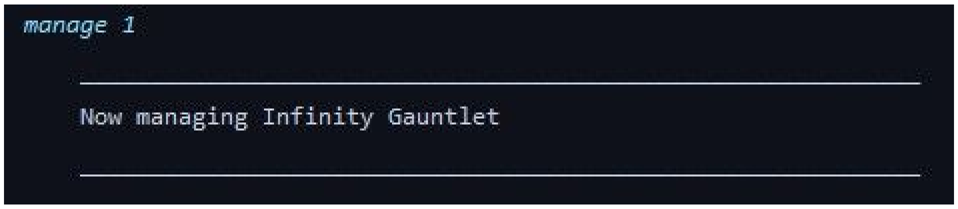


```
list
Here are all the Projects you are managing:
    ____________________________________________________________
    +--------------------------------------------------------------+
    |Project 1: Infinity Gauntlet                                  |
    +--------------------------------------------------------------+
    |Members:                                                      |
    | --                                                           |
    |Next Deadline:                                                |
    | --                                                           |
    |Overall Progress:                                             |
    | --                                                           |
    +--------------------------------------------------------------+
    ____________________________________________________________
```

3.3. Manage a project : `manage`

Allows user to manage a project at the specified index. The index refers to the index number shown in the list of all projects and project details through the command view. Index number are generated using 1-based indexing.
Format: `manage PROJECT_INDEX`

- All commands labelled 3.3.x are only executable after user has selected a project that he or she wishes to manage

Example: `manage 1`


```
manage 1
    ____________________________________________________________

    Now managing Infinity Gauntlet
    ____________________________________________________________
```

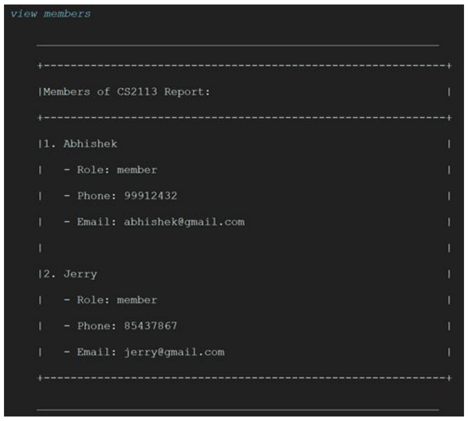
3.3.1. Add members to specific project: `add member`

Adds members to the specific project selected using manage.
Format: `add member -n NAME [-i PHONE_NUMBER] [-e EMAIL_ADDRESS] [-r ROLE]`
Example: `add member -n Jerry Zhang -i 9123456 -e jerryzhang@gmail.com -r Lead`

- Email must be a valid email address with the format "<String>@<String>.com"

3.3.2. View members : `view members`

Displays all members' details like their name, phone number, email and their role in the project.
Format: `view members`

`view members`


```
view members
    ____________________________________________________________
    +-----------------------------------------------------------+
    |Members of CS2113 Report:                                  |
    +-----------------------------------------------------------+
    |1. Abhishek                                                |
    |   - Role: member                                          |
    |   - Phone: 99912432                                       |
    |   - Email: abhishek@gmail.com                             |
    |                                                           |
    |2. Jerry                                                   |
    |   - Role: member                                          |
    |   - Phone: 85437867                                       |
    |   - Email: jerry@gmail.com                                |
    +-----------------------------------------------------------+
    ____________________________________________________________
```

3.3.3. Edit members : `edit member`

Edits details of the members. Only fields that need editing need to be entered
Format: `edit member INDEX [-n NAME] [-i PHONE_NUMBER] [-e EMAIL]`
Example: `edit member 2 -n Jerry Zhang -e zhangjerry@u.nus.edu`

- The member index must be a positive integer

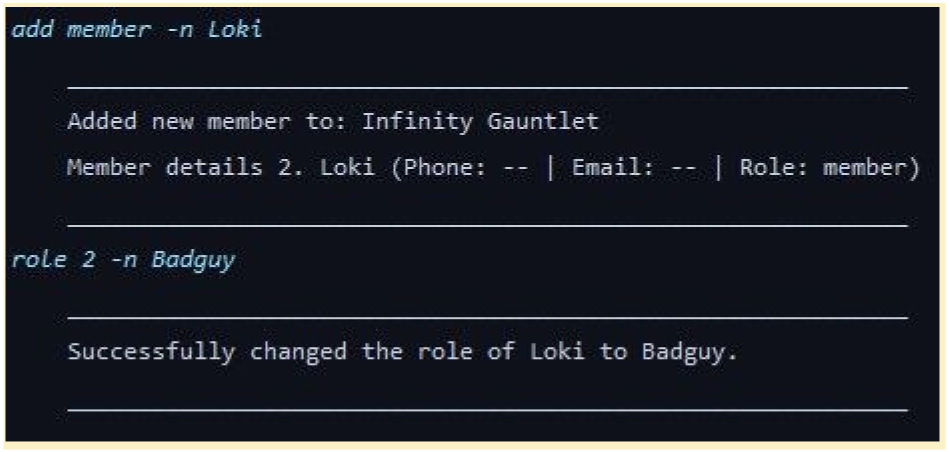
3.3.4. Delete members : `delete member`

Remove **a** selected member part of the project based on their index number. Index number is generated using 1-based indexing and can be viewed using the `view members` command.
Format: `delete member INDEX`
Example: `delete member 4`

3.3.5. Give roles to specific members in a project : `role`

This command allows a user to assign roles to specific members, where a role can be any String. Members must be specified using the index of a specific member in a group project.
Format: `role INDEX -n ROLE_NAME`

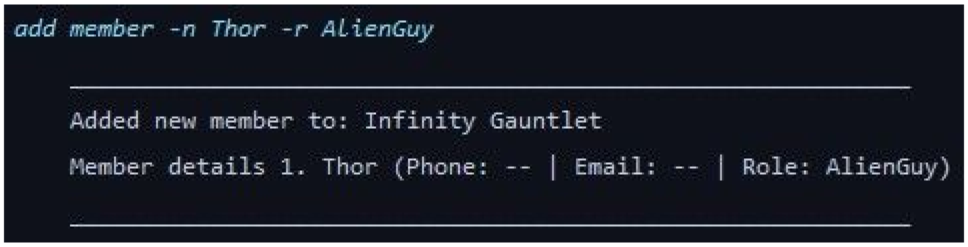
Example: `role 2 -n Badguy`



```
add member -n Loki
    ____________________________________________________________
    Added new member to: Infinity Gauntlet
    Member details 2. Loki (Phone: -- | Email: -- | Role: member)
    ____________________________________________________________
role 2 -n Badguy
    ____________________________________________________________
    Successfully changed the role of Loki to Badguy.
    ____________________________________________________________
```

The role of a new member can also be changed during the addition, by using the `-r` flag

Example: `add member -n Thor -r AlienGuy`

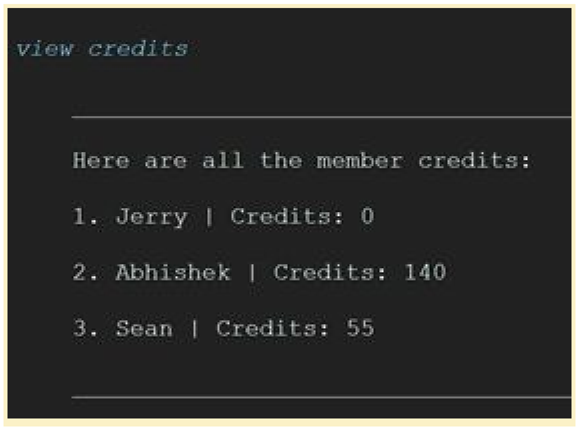


```
add member -n Thor -r AlienGuy
    ____________________________________________________________
    Added new member to: Infinity Gauntlet
    Member details 1. Thor (Phone: -- | Email: -- | Role: AlienGuy)
    ____________________________________________________________
```

3.3.6. View total credits completed by each member : `view credits`

Shows all members' credits, their index number, name, and name of tasks completed.
Format: `view credits`

`view credits`



```
view credits

    ____________________________________________
    Here are all the member credits:
    1. Jerry | Credits: 0
    2. Abhishek | Credits: 140
    3. Sean | Credits: 55
    ____________________________________________
```

3.3.7. Add a task to project : `add task`

Adds a task to the project.
Format: `add task -t TASK_NAME -p TASK_PRIORITY [-d TASK_DUEDATE-(dd/mm/yyyy)] -c TASK_CREDIT [-s STATE] [-r TASK_REQUIREMENT1] [-r TASK_REQUIREMENT2]`
Example: `add task -t Documentation for product -p 2 -d 21/09/2019 -c 40 -r do something -r do another thing`

- Task priority is represented by an integer from 1-5 which denotes how important a task is, with a higher number meaning higher priority.
- Task credit is represented by an integer from 0 to 100, which denotes the amount of credit a person would receive for completing a task (eg. a more difficult task which requires more work would receive higher credit.)
    - Task credit is accumulated throughout the selected project
- Due date may or may not need to be added depending on the nature of the task. The input for date is in the **dd/mm/yyyy** format. E.g. 21/09/2019
- Each task will be given an index number based on the amount of tasks in the list
- State refers to whether the task is in **OPEN, TODO, DOING, DONE**. If no state is specified, task created will automatically be assigned to **OPEN.**
- Any number of task requirements can be added to give specific requirements for the task so that the member will be clear of what to do

3.3.8. Edit a task : `edit task`

Edits the details of a task: task name, priority, credit and due date. Only fields that need to be edited must be filled in. Fields that are not filled will not be changed. Task requirements cannot be edited through this command

Format: `edit task TASK_INDEX [-t TASK_NAME] [-p TASK_PRIORITY] [-d TASK_DUEDATE] [-c TASK_CREDIT] [-s STATE]`
Example: `edit task 12 -p 5 -c 80`
Example: `edit task 4 -s doing`

3.3.7. Assign/Remove tasks to/from members: `assign`

Delegates the task to a member/several members in the group. Assignments to the task can be added or removed based on the command given.
Format: `assign task -i TASK_INDEX -to [MEMBER1_INDEX] [MEMBER2_INDEX] -rm [MEMBER3_INDEX]`
Example: `assign task -i 1 -to 3 4` `//assigns members 3 and 4`
Example: `assign task -i 1 -to 5 -rm 3` `//assigns member 5, removes 3`
Example: `assign task -i 1 -rm 2 4` `//removes member 2 and 4`

- A task may be assigned to multiple members. The task credit will then be split evenly between them.
- Multiple tasks may be assigned or unassigned for multiple members in 1 command.
  Example: `assign task -i 1 2 -to 1 2 -rm 3 4`
  `//for tasks 1 and 2, assign members 1, 2 and unassign members 3,4`

3.3.9. Delete a task : `delete task`

Removes a task from the current selected project. All details regarding the task will be removed, and should it be assigned to any member, the member will automatically be unassigned.
Format: `delete task INDEX`
Example: `delete task 1`

3.3.10. View tasks : `view tasks`

Shows user a list of all tasks in the current project, sorted based on certain criteria as chosen by the user:

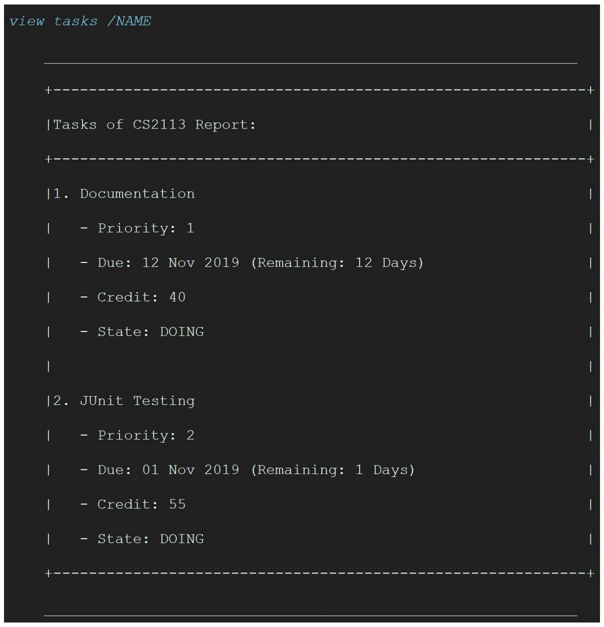
Sorting criteria	**Modifier**
Sort based on task name	`NAME`
1-based indexing based on the data structure tasks are stored in	`INDEX`
Based on the deadline/due date of the task: from earliest to latest. Tasks without a deadline will be classified as the latest	`DATE`
Sort based on task priority from highest to lowest priority (ascending)	`PRIORITY`
Sort based on task credit from highest to lowest credit (descending)	`CREDIT`

Sort based on the name of member assigned to all tasks in alphabetical order	**`WHO-{name}`**
State: List from **OPEN, TODO, DOING, DONE** in a Kanban board style	**`KANBAN-{state}`**

Format: `view tasks /MODIFIER`

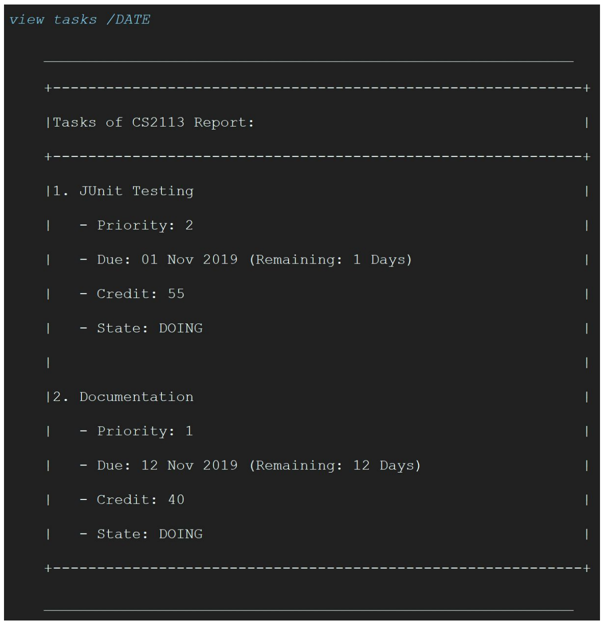
Example: `view tasks /NAME` `//shows all the tasks sorted lexicographically`
Example: `view tasks /DATE` `//show all the dates sorted by the due date`
Example: `view tasks /PRIORITY` `//shows all the dates sorted by priority`
Example: `view tasks /CREDIT` `//shows all the tasks sorted by credits`
Example: `view tasks /WHO-Dillen` `//shows all the tasks assigned to "Dillen"`
Example: `view tasks /KANBAN-DOING` `//shows all the tasks with the state "DOING"`

`view tasks /NAME`



```
view tasks /NAME

    ____________________________________________________________

    +----------------------------------------------------------+
    |Tasks of CS2113 Report:                                   |
    +----------------------------------------------------------+
    |1. Documentation                                          |
    |    - Priority: 1                                         |
    |    - Due: 12 Nov 2019 (Remaining: 12 Days)               |
    |    - Credit: 40                                          |
    |    - State: DOING                                        |
    |                                                          |
    |2. JUnit Testing                                          |
    |    - Priority: 2                                         |
    |    - Due: 01 Nov 2019 (Remaining: 1 Days)                |
    |    - Credit: 55                                          |
    |    - State: DOING                                        |
    +----------------------------------------------------------+
    ____________________________________________________________
```

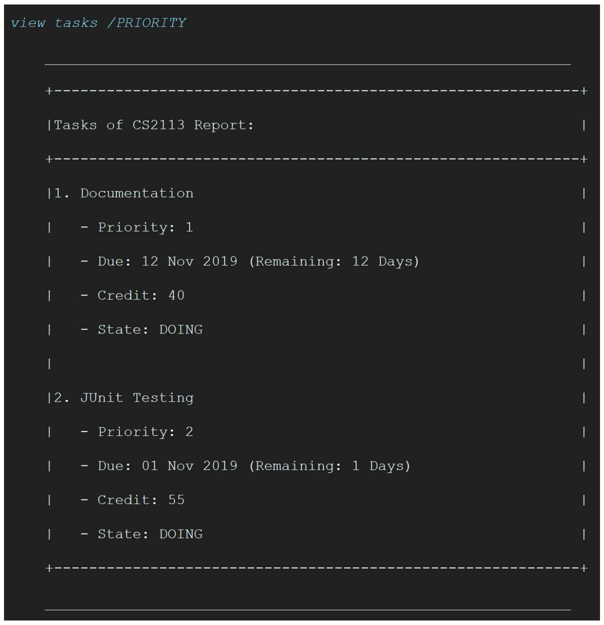
`view tasks /DATE`



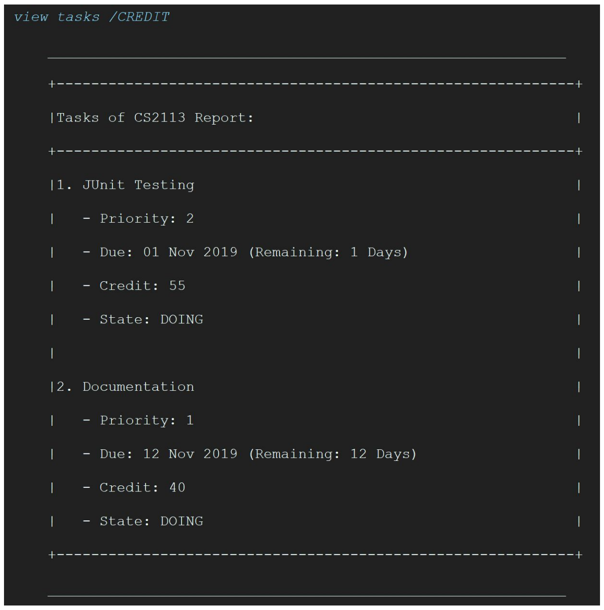
```
view tasks /DATE

    ____________________________________________________________

    +----------------------------------------------------------+
    |Tasks of CS2113 Report:                                   |
    +----------------------------------------------------------+
    |1. JUnit Testing                                          |
    |    - Priority: 2                                         |
    |    - Due: 01 Nov 2019 (Remaining: 1 Days)                |
    |    - Credit: 55                                          |
    |    - State: DOING                                        |
    |                                                          |
    |2. Documentation                                          |
    |    - Priority: 1                                         |
    |    - Due: 12 Nov 2019 (Remaining: 12 Days)               |
    |    - Credit: 40                                          |
    |    - State: DOING                                        |
    +----------------------------------------------------------+
    ____________________________________________________________
```

`view tasks /PRIORITY`

`view tasks /CREDIT`

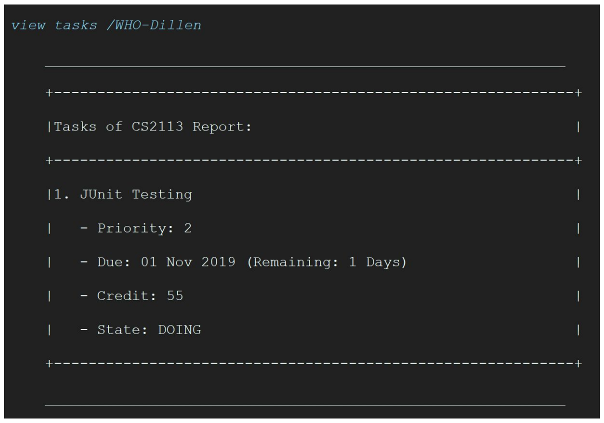


```
view tasks /PRIORITY
 ____________________________________________________________
 +-----------------------------------------------------------+
 |Tasks of CS2113 Report:                                    |
 +-----------------------------------------------------------+
 |1. Documentation                                           |
 |   - Priority: 1                                           |
 |   - Due: 12 Nov 2019 (Remaining: 12 Days)                 |
 |   - Credit: 40                                            |
 |   - State: DOING                                          |
 |                                                           |
 |2. JUnit Testing                                           |
 |   - Priority: 2                                           |
 |   - Due: 01 Nov 2019 (Remaining: 1 Days)                  |
 |   - Credit: 55                                            |
 |   - State: DOING                                          |
 +-----------------------------------------------------------+
 ____________________________________________________________
```

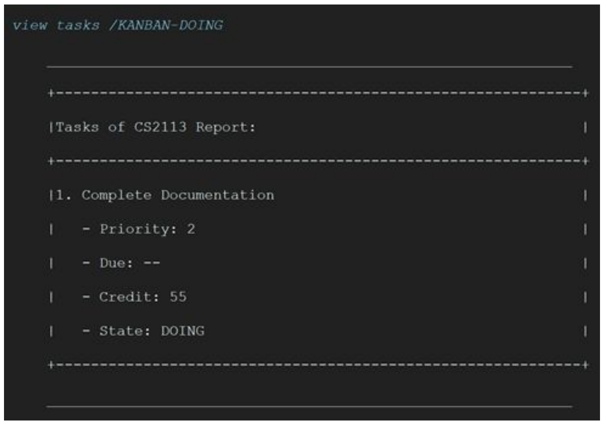


```
view tasks /CREDIT
 ____________________________________________________________
 +-----------------------------------------------------------+
 |Tasks of CS2113 Report:                                    |
 +-----------------------------------------------------------+
 |1. JUnit Testing                                           |
 |   - Priority: 2                                           |
 |   - Due: 01 Nov 2019 (Remaining: 1 Days)                  |
 |   - Credit: 55                                            |
 |   - State: DOING                                          |
 |                                                           |
 |2. Documentation                                           |
 |   - Priority: 1                                           |
 |   - Due: 12 Nov 2019 (Remaining: 12 Days)                 |
 |   - Credit: 40                                            |
 |   - State: DOING                                          |
 +-----------------------------------------------------------+
 ____________________________________________________________
```

`view tasks /WHO-Dillen`	`view tasks /KANBAN-DOING`



```
view tasks /WHO-Dillen
 ____________________________________________________________
 +-----------------------------------------------------------+
 |Tasks of CS2113 Report:                                    |
 +-----------------------------------------------------------+
 |1. JUnit Testing                                           |
 |   - Priority: 2                                           |
 |   - Due: 01 Nov 2019 (Remaining: 1 Days)                  |
 |   - Credit: 55                                            |
 |   - State: DOING                                          |
 +-----------------------------------------------------------+
 ____________________________________________________________
```



```
view tasks /KANBAN-DOING
 ____________________________________________________________
 +-----------------------------------------------------------+
 |Tasks of CS2113 Report:                                    |
 +-----------------------------------------------------------+
 |1. Complete Documentation                                  |
 |   - Priority: 2                                           |
 |   - Due: --                                               |
 |   - Credit: 55                                            |
 |   - State: DOING                                          |
 +-----------------------------------------------------------+
 ____________________________________________________________
```

3.3.11. View task assignments : `view assignments`

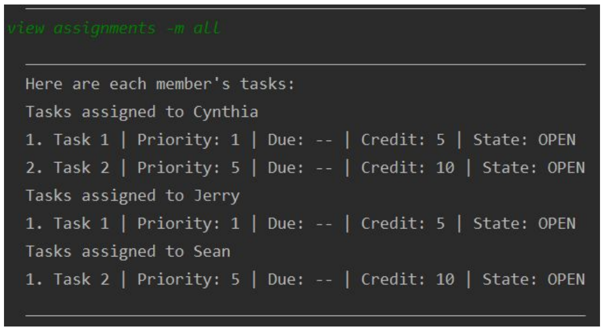
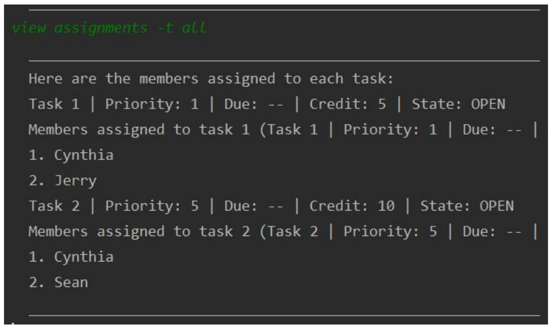
Shows user the task assignments based on certain specifications. This allows the user to check which task is assigned to who, and vice versa.

Specification	**Modifier**
View selected members' individual assigned tasks	**`-m INDEX_NUMBERS`**
View all members' individual assigned tasks	**`-m all`**

View the members assigned to selected tasks	**`-t INDEX_NUMBERS`**
View the members assigned to all tasks	**`-t all`**

Format: `view assignments /MODIFIER`
Example: `view assignments -m all` `//show task list of all members`
Example: `view assignments -m 1 2` `//show task list of members 1 and 2`
Example: `view assignments -t all` `//show members assigned to all tasks`
Example: `view assignments -t 3 4 5` `//show members assigned to tasks 3, 4, 5`

In case of confusion, the following are examples that demonstrate the difference between -m and -t, and how they can be used to show assignments in different formats

`view assignments -m all`	`view assignments -t all`
	

```
view assignments -m all

 Here are each member's tasks:
 Tasks assigned to Cynthia
 1. Task 1 | Priority: 1 | Due: -- | Credit: 5 | State: OPEN
 2. Task 2 | Priority: 5 | Due: -- | Credit: 10 | State: OPEN
 Tasks assigned to Jerry
 1. Task 1 | Priority: 1 | Due: -- | Credit: 5 | State: OPEN
 Tasks assigned to Sean
 1. Task 2 | Priority: 5 | Due: -- | Credit: 10 | State: OPEN
```

```
view assignments -t all

 Here are the members assigned to each task:
 Task 1 | Priority: 1 | Due: -- | Credit: 5 | State: OPEN
 Members assigned to task 1 (Task 1 | Priority: 1 | Due: -- |
 1. Cynthia
 2. Jerry
 Task 2 | Priority: 5 | Due: -- | Credit: 10 | State: OPEN
 Members assigned to task 2 (Task 2 | Priority: 5 | Due: -- |
 1. Cynthia
 2. Sean
```

3.3.12. Show an agenda of tasks that are due within a time period : `agenda`

Displays a list of tasks within a certain time period, much like a calendar. Time period that are allowed to be specified are `DAY`, `WEEK`, `MONTH`, `YEAR`
Format: `agenda -s TIME_PERIOD`
Example: `agenda -s MONTH`

- `agenda -s MONTH` as opposed to `view tasks /DATE` is more for users to view a breakdown of the list of tasks that are due within a certain time period, while `view tasks /DATE` is more suitable for users to view every task, with the most pressing task first.

3.3.13. View task requirements : `view task requirements`

Shows user a list of task requirements for a specific task in the current project:
Format: `view task requirements TASK_INDEX`
Example: `view task requirements 2`

3.3.14. Edit task requirements : `edit task requirements`

Adds or removes a list of task requirements to a specific task in the current project:

Format: `edit task requirements TASK_INDEX [-rm TASK_REQUIREMENT_INDEX] [-r TASK_REQUIREMENT]`
Example: `edit task requirements 2 -r do something -r do another thing`
Example: `edit task requirements 2 -rm 1 2 4 -r do something`

- User is able to input multiple requirements to be added and multiple requirements to be removed

3.3.15. Create reminder: `add reminder [to be implemented]`

Adds a reminder to the project.
Format: `add reminder -n reminder_NAME [-d TASK_DUEDATE-(dd/mm/yyyy) -l REMINDER_LIST_NAME]`
Example: `add task -n Fix important bug -d 21/09/2019 -l Software Reminder List`

- Due date may or may not need to be added depending on the nature of the reminder. The input for date is in the **dd/mm/yyyy** format. E.g. 21/09/2019

3.3.16. View reminder: `view reminder [to be implemented]`

View the list of reminders in the project.
Format: `view reminder [-l LIST_NAME]`
Example: `view reminder -l Systems Reminder List`

3.3.17. Delete reminder: `delete reminder [to be implemented]`

Deletes a reminder from the project.
Format: `delete reminder INDEX_NUMBER`
Example: `delete reminder 1`

3.3.18. Edit reminder: `edit reminder [to be implemented]`

Edit a reminder in the project.
Format: `edit reminder INDEX_NUMBER -n REMINDER_NAME [-d REMINDER_DUEDATE-(dd/mm/yyyy) -l REMINDER_LIST_NAME]`
Example: `edit reminder 1 -n Fix important bug -d 21/09/2019`

3.4. Delete a project : `delete project`

Allows user to delete a project at the specified index. The index refers to the index number shown in the list of all projects and project details through the command view. Index number is generated using 1-based indexing.
Format: `delete PROJECT_INDEX`
Example: `delete 2`

3.5. Report progress : `report [to be implemented]`

Reports the progress of all projects, and the contributions of each member (the credit each member earned, the dates of tasks completed, whether the tasks were overdue).
Format: `report`

3.6. Exit : `exit`

Exits the current state in application runtime and brings user out of current state by one level. For example, if the user has selected a task in a project, `exit` will exit the state of having a task selected, changing the user state to having a project selected.
If the user is currently editing a selected project, `exit` will bring the user out of a project back so that they can choose to manage another project.
Format: `exit`

3.7. Exiting the program from anywhere: `bye`

Exits the program, regardless of what the user was doing at the current point of application runtime. User will be able to exit the program even when they have selected a project, or when they have selected a task.
Format: `bye`

3.8. Shows list of available commands: `help`

Displays a list of commands that are available for use at the current point of application
Format: `help`

4. FAQ

Q: How can I transfer data from one workstation to another?
A: The application must be installed in the other computer by downloading the same version of the .jar file used in the original work environment. You may then copy the data file from the original work environment and overwrite the empty data file in the new work environment.

Q: How do I use your application if I do not wish to see any GUI?
A: At the beginning of program runtime, ArchDuke will prompt you if you wish to use a GUI version or a CLI version. Both GUI and CLI will have the same functionalities, meaning anything that can be done in the GUI can be done via CLI. For example, the viewing of the Kanban board will be visualised using dots and slashes to draw out a board on the command line.

5. Command Summary

- **Create Project:** `create PROJECT_NAME`
  Example: `create Infinity Gauntlet`

- **View Projects:** `list`

- **Manage:** `manage PROJECT_INDEX`
- Example: `manage 2`

  - **Add Members:** `add member -n NAME [-i PHONE_NUMBER] [-e EMAIL] [-r ROLE]`
    Example: `add member -n Jerry Zhang -i 9123456 -e jerryzhang@gmail.com -r Lead`

  - **View Members:** `view member`

  - **Edit Members:** `edit member INDEX [-n NAME] [-i PHONE_NUMBER] [-e EMAIL]`
    Example: `edit member 2 -n Jerry Zhang -e zhangjerry@u.nus.edu`

  - **Delete Members:** `delete member INDEX`
    Example: `delete member 3`

  - **Give Roles for Members:** `role INDEX -n ROLE_NAME`
    Example: `role 2 -n Badguy`

  - **View total credits completed by each member:** `view credits`
    Example: `view credits`

  - **Add Task:** `add task -t TASK_NAME -p TASK_PRIORITY [-d TASK_DUEDATE-(dd/mm/yyyy)] -c TASK_CREDIT [-s STATE] [-r TASK_REQUIREMENT1] [-r TASK_REQUIREMENT2]`
    Example: `add task -t Documentation for product -p 2 -d 21/09/2019 -c 40 -r do something -r do another thing -r do another thing`

  - **Edit Task:** `edit task TASK_INDEX [-t TASK_NAME] [-p TASK_PRIORITY] [-d TASK_DUEDATE-(dd/mm/yyyy)] [-c TASK_CREDIT] [-s TASK_STATE]`
    Example: `edit task 12 -p 5 -c 80`

- **Assign/Remove members to/from Task:** `assign task -i TASK_INDEX -to [MEMBER1_INDEX] [MEMBER2_INDEX] -rm [MEMBER3_INDEX]`
  Example: `assign task -i 1 -to 2 5` `//assigns members 2 and 5`
  Example: `assign task -i 1 -to 5 -rm 1` `//assigns member 5 and removes 1`

- **Delete Tasks:** `delete task INDEX`
  Example: `delete task 3`

- **View Tasks:** `view tasks /MODIFIER`
  Example: `view tasks /KANBAN-DOING` or `view task /NAME`

- **View Tasks:** `view assignments /MODIFIER`
  Example: `view assignments -m all` or `view assignments -t 3 4 5`

- **Agenda of Tasks:** `agenda -s TIME_PERIOD`
  Example: `agenda -s MONTH`

- **View task requirements:** `view task requirements TASK_INDEX`
  Example: `view task requirements 1`

- **Edit task requirements:** `edit task requirements TASK_INDEX [-rm TASK_REQUIREMENT_INDEX] [-r TASK_REQUIREMENT]`
  Example: `edit task requirements 1 -r new task requirement -r another task requirement -rm 1 2 4`

- **Create reminder:** `add reminder -n reminder_NAME [-d TASK_DUEDATE-(dd/mm/yyyy) -l REMINDER_LIST_NAME] [to be implemented]`
  Example: `add task -n Fix important bug -d 21/09/2019 -l Software Reminder List`

- **View reminders:** `view reminder [-l LIST_NAME] [to be implemented]`
  Example: `view reminder -l Systems Reminder List`

- **Delete reminder:** `delete reminder INDEX_NUMBER [to be implemented]`
  Example: `delete reminder 1`

- **Edit reminder:** `edit reminder INDEX_NUMBER -n REMINDER_NAME [-d REMINDER_DUEDATE-(dd/mm/yyyy) -l REMINDER_LIST_NAME] [to be implemented]`
  Example: `edit reminder 1 -n Fix important bug -d 21/09/2019`

- **Delete a Project:** `delete PROJECT_INDEX`
  Example: `delete 1`

- **Report:** `report [to be implemented]`

- **Moving out of the current state:** `exit`
  For example: `exit` to exit out of the selected project to view all created projects

- **Exiting the program from anywhere:** `bye`

- **Shows a list of available commands:** `help`