ResuMe - User Guide

By: `CS2103T F10-1`   Since: `Jan 2020`   Licence: `MIT`

{ start of `introduction` section written by: Nguyen Chi Hai }

1. Introduction

Hello fellow University friends! Have you ever found writing resumes for job and internship applications a hassle? Have you ever dreamt about an app that can help you manage different resumes for different companies? Have you ever wished for a more streamlined **Command Line Interface** application that can match your efficiency and typing prowess?

If this sounds like you, **ResuMe** is *the* resume managing application for you! Made by University students for University students with *looovee*, this wonder understands your struggle and strives to cater to your specific resume needs and job application woes.

{ end of `introduction` section written by: Nguyen Chi Hai }

1.1. Overview

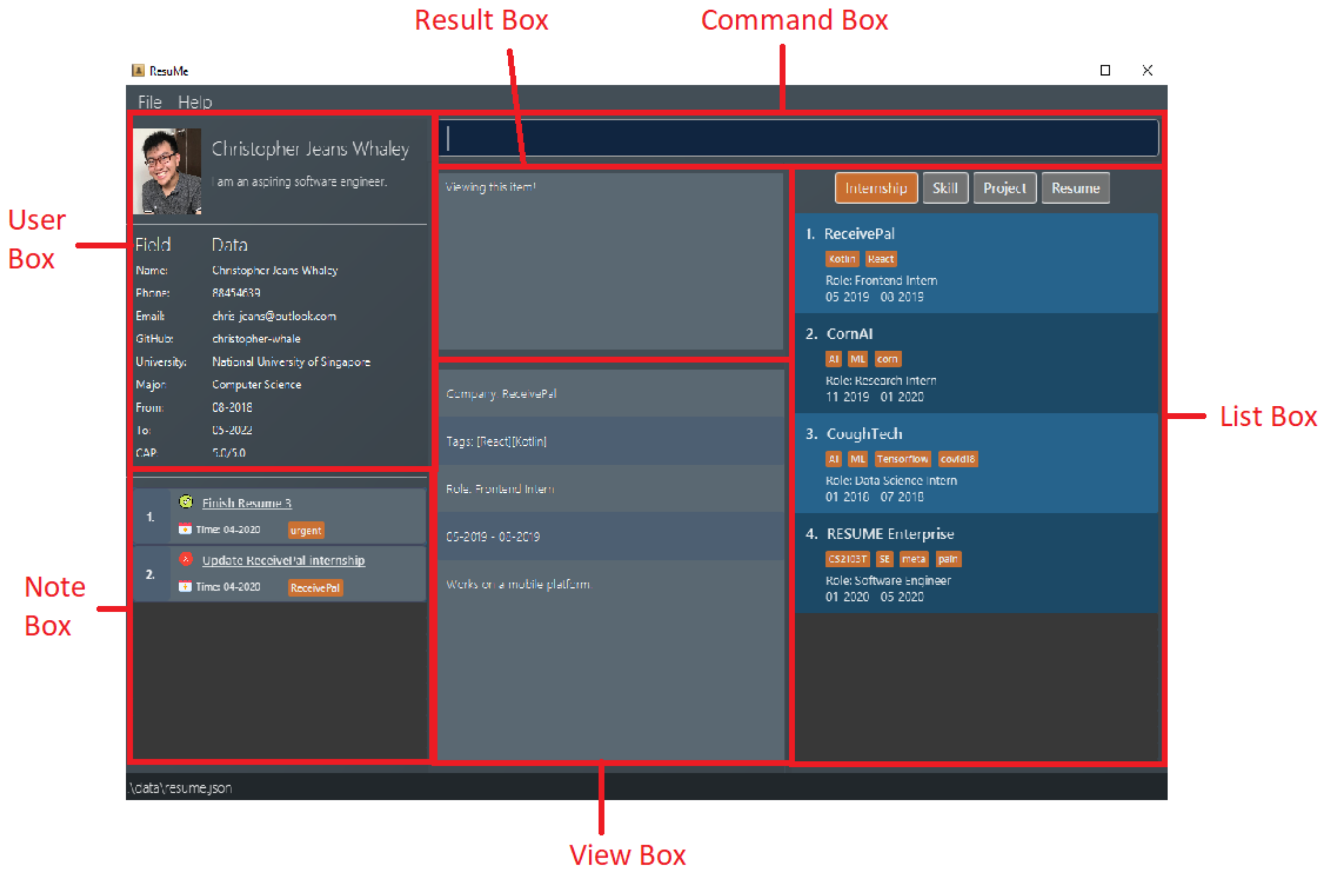
ResuMe allows the management of five different item types:

No.	Item type	Alias	Attributes
1	Internship	int	name, role, from, to, description, tag
2	Project	proj	name, time, website, description, tag
3	Skill	ski	name, level, tag
4	Resume	res	name, tag
5	Note	note	name, time, tag

As you may have noticed from number 4, we are able to manage different `resume` items as well! Items from number 1 to 3 can be shared across different `resume` items.

1.2. Preview

The following is the GUI for our application:



Christopher Jeans Whaley
I am an aspiring software engineer.

Field	Data
Name:	Christopher Jeans Whaley
Phone:	88454639
Email:	chris.jeans@outlook.com
GitHub:	christopher-whale
University:	National University of Singapore
Major:	Computer Science
From:	08-2018
To:	05-2022
CAP:	5.0/5.0

1. Finish Resume 3
   Time: 04-2020 urgent
2. Update ReceivePal internship
   Time: 04-2020 ReceivePal

Viewing this item!

Company: ReceivePal
Tags: [React][Kotlin]
Role: Frontend Intern
05-2019 - 08-2019
Works on a mobile platform

1. ReceivePal
   Kotlin React
   Role: Frontend Intern
   05 2019 08 2019
2. CornAI
   AI ML corn
   Role: Research Intern
   11 2019 01 2020
3. CoughTech
   AI ML Tensorflow covid19
   Role: Data Science Intern
   01 2018 07 2018
4. RESUME Enterprise
   CS2103T SE meta pain
   Role: Software Engineer
   01 2020 05 2020

\data\resume.json

Figure 1. The GUI Components of ResuMe

GUI Components:

1. ***Command box:*** where you enter your commands.
2. ***Result box:*** where the result of a command is shown. It can show a success message or a failure message if something wrong happens.
3. ***View box:*** where the detailed information about an item is shown. For instance, if you are viewing an `internship` item, this box shows in full the `company name`, `role`, `from`, `to`, `description` and `tags`.
4. ***List box:*** where all items of a certain type are listed. Only one type is displayed at a time. You can change the view of this list using `list` command. Additionally, when other commands change items of a certain type, the list box will be set to display that type.
5. ***User box:*** where you can view your personal information like `name`, `email`, or `phone number`. Any changes made to your user profile will be automatically reflected here.
6. ***Note box:*** where you can view your list of `note` items. You can also change the view of this list by using `list` command. When a `note` is marked as done, this box will be automatically updated.

Interested? Jump to Section 3, “Quick Start” to get started. Enjoy!

{ start of `about guide` section written by: Pham Thuy Duong }

2. About this guide

Our aim when writing this document is to help you - a new user - get started with **ResuMe** as quickly as possible. The guide features all the available commands, their syntax, as well as specific examples for illustration.

This section is, well… a guide on how to use this guide :) We will be explaining some of the symbols and formatting used, key terms, as well notes and tips for a better user experience.

2.1. Symbols and Formatting

Most of us are visual learners, and we figured, what is a better way to present information than using some shapes and colours?

Symbol / Format	Explanation
ℹ	Important information to take note
💡	Tips
⚠	Warnings on potential misuse
`consola font`	Keywords representing an `item type` or their `parameters`, or a `command` that could be executed

2.2. Key terms

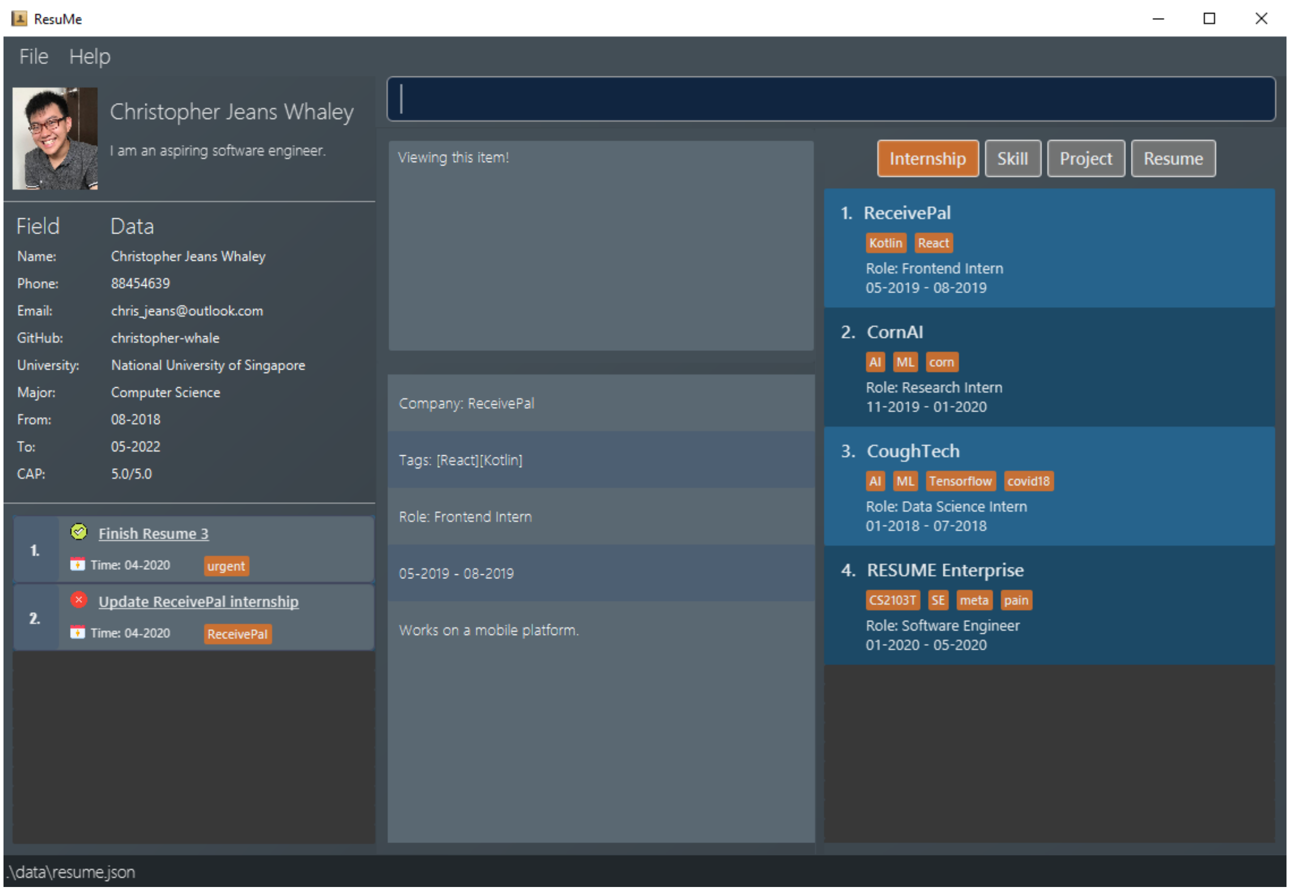
You will come across these little guys a lot in this document, so let's get to know them first!

Term	Explanation
`type`	Consisting of five different types: `internship`, `project`, `skill`, `resume` and `note`
`item`	An item stored in the application, which could be any one of the five types above
`command`	A line of text that you could type into the command box and execute
`parameter`	Specific information about an `item` supplied by you
`index`	The position of the `item` in the currently displayed list

{ end of `about guide` section written by: Pham Thuy Duong }

3. Quick Start

1. Ensure you have Java `11` or above installed in your Computer.
2. Download the latest `CS2103T-F10-1.ResuMe.jar` here.
3. Copy the file to the folder you want to use as the home folder for your **ResuMe** application.
4. Double-click the file to start the app. The GUI should appear in a few seconds.



Christopher Jeans Whaley

I am an aspiring software engineer.

Field	Data
Name:	Christopher Jeans Whaley
Phone:	88454639
Email:	chris_jeans@outlook.com
GitHub:	christopher-whale
University:	National University of Singapore
Major:	Computer Science
From:	08-2018
To:	05-2022
CAP:	5.0/5.0

1. Finish Resume 3
   Time: 04-2020 urgent
2. Update ReceivePal internship
   Time: 04-2020 ReceivePal

Viewing this item!

Company: ReceivePal

Tags: [React][Kotlin]

Role: Frontend Intern

05-2019 - 08-2019

Works on a mobile platform.

1. ReceivePal
   Kotlin React
   Role: Frontend Intern
   05-2019 - 08-2019
2. CornAI
   AI ML corn
   Role: Research Intern
   11-2019 - 01-2020
3. CoughTech
   AI ML Tensorflow covid18
   Role: Data Science Intern
   01-2018 - 07-2018
4. RESUME Enterprise
   CS2103T SE meta pain
   Role: Software Engineer
   01-2020 - 05-2020

.\data\resume.json

Figure 2. The GUI of ResuMe

5. Type a command in the command box and press `Enter` to execute it.
   e.g. typing **`help start`** and pressing `Enter` will open the help window for getting started.
6. Try out these example commands:
   - `list i/ res` : lists all `resume` items.
   - `add i/ res n/ Software Engineering Resume #/ SE #/ frontend`: adds a `resume` named `Software Engineering Resume` with `tags` *"SE"* and *"frontend"*.
   - `delete 1 i/ res` : deletes the 1st `resume` shown in the resume list.
   - `exit` : exits the app.
7. Refer to Section 4, “Features” for details of each command.

4. Features

For this section, we will introduce you to all of the available features of our application.

Command Format

- Words in `UPPER_CASE` are the parameters to be supplied by you.
  For example, in `add i/ TYPE n/ NAME`, `TYPE` and `NAME` are parameters which can be customised to be `add i/ proj n/ Orbital`.
- Parameters in square brackets are optional.
  For example, `n/ NAME [#/TAG]` can be used as `n/ Orbital #/ SE` or as `n/ Orbital`.
- Parameters with `…` after them can be used multiple times including zero times.
  For example, `[#/ TAG]…` can be used as:
  - (i.e. 0 times)
  - `#/ SE`
  - `#/ SE #/ frontend` etc.
- Parameters can be in any order.
  For example, if the command specifies `n/ NAME p/ PHONE_NUMBER`, `p/ PHONE_NUMBER n/ NAME` is also acceptable.

{ start of `help` section written by: Pham Thuy Duong }

4.1. Viewing help : `help`

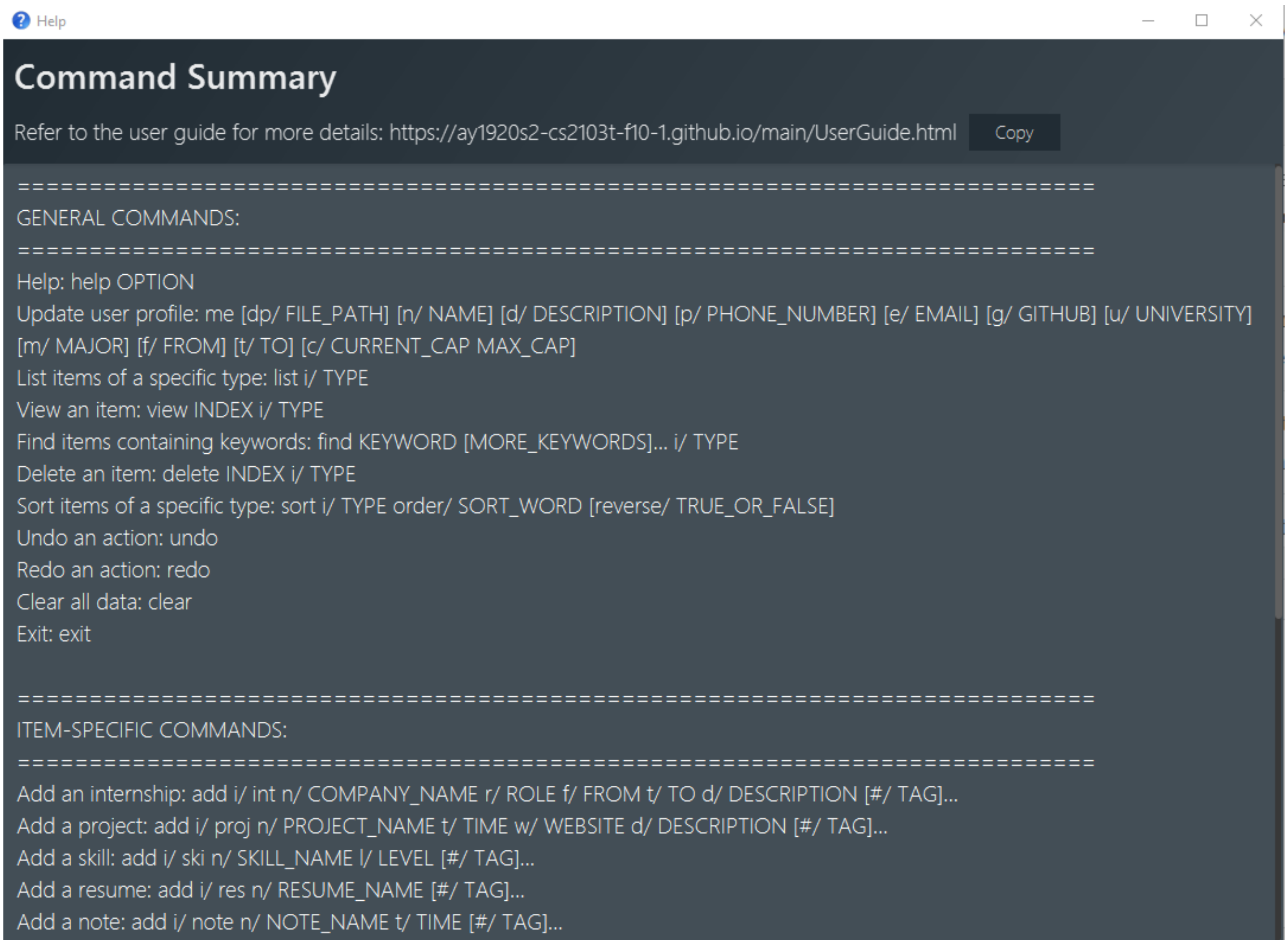
Lists out the function and usage of each command.

Format: `help OPTION`

There are two possible `help` options that you can choose from, as shown in the table below:

Option	Format	Result
command	`help command`	Opens a pop-up window with a summary of available commands, their syntax and usages.
start	`help start`	Opens a pop-up window with detailed guide on how to get started.

These are screenshots of the two pop-up windows:



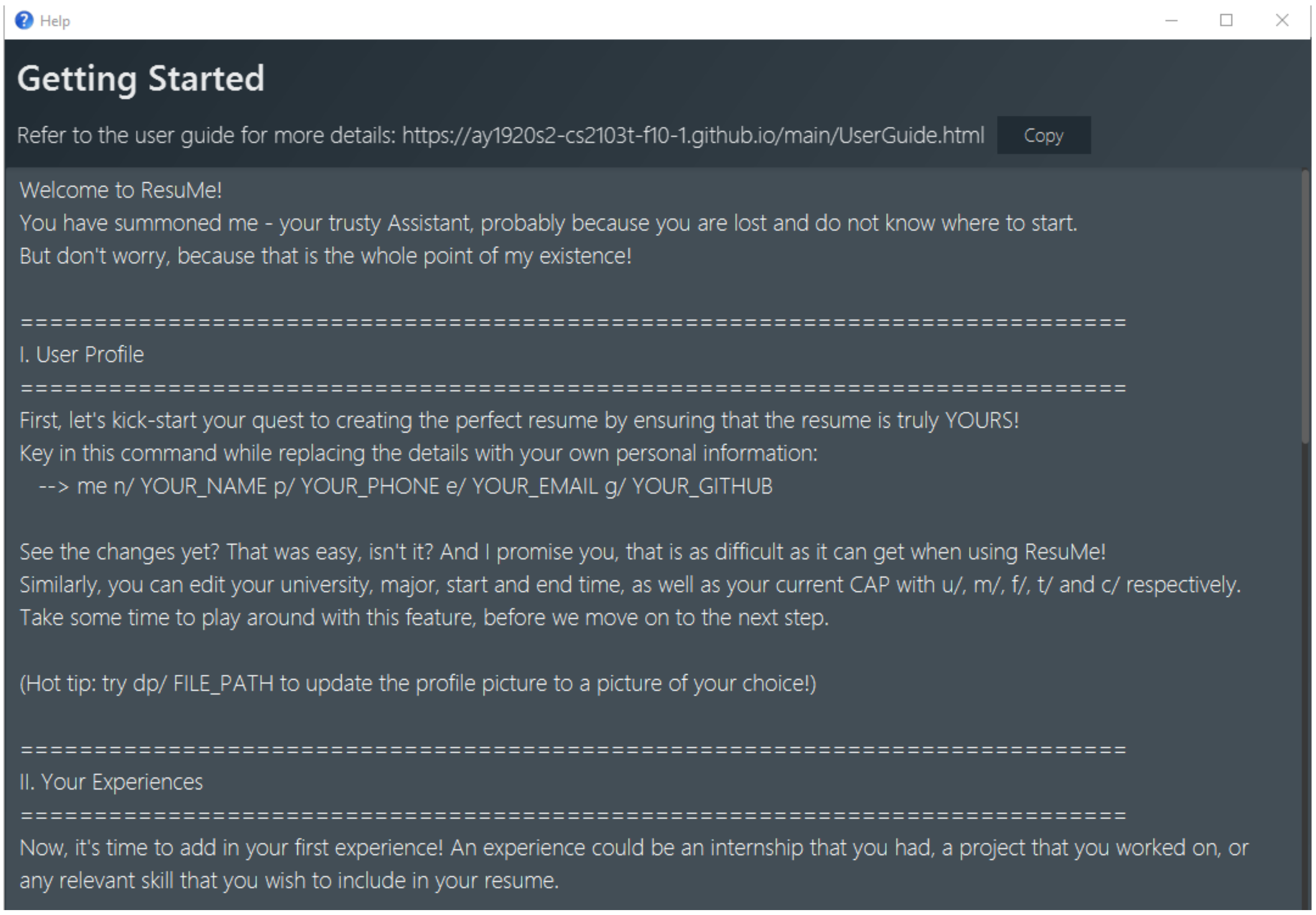
Command Summary

Refer to the user guide for more details: https://ay1920s2-cs2103t-f10-1.github.io/main/UserGuide.html

```
==========================================================================
GENERAL COMMANDS:
==========================================================================
Help: help OPTION
Update user profile: me [dp/ FILE_PATH] [n/ NAME] [d/ DESCRIPTION] [p/ PHONE_NUMBER] [e/ EMAIL] [g/ GITHUB] [u/ UNIVERSITY]
[m/ MAJOR] [f/ FROM] [t/ TO] [c/ CURRENT_CAP MAX_CAP]
List items of a specific type: list i/ TYPE
View an item: view INDEX i/ TYPE
Find items containing keywords: find KEYWORD [MORE_KEYWORDS]... i/ TYPE
Delete an item: delete INDEX i/ TYPE
Sort items of a specific type: sort i/ TYPE order/ SORT_WORD [reverse/ TRUE_OR_FALSE]
Undo an action: undo
Redo an action: redo
Clear all data: clear
Exit: exit

==========================================================================
ITEM-SPECIFIC COMMANDS:
==========================================================================
Add an internship: add i/ int n/ COMPANY_NAME r/ ROLE f/ FROM t/ TO d/ DESCRIPTION [#/ TAG]...
Add a project: add i/ proj n/ PROJECT_NAME t/ TIME w/ WEBSITE d/ DESCRIPTION [#/ TAG]...
Add a skill: add i/ ski n/ SKILL_NAME l/ LEVEL [#/ TAG]...
Add a resume: add i/ res n/ RESUME_NAME [#/ TAG]...
Add a note: add i/ note n/ NOTE_NAME t/ TIME [#/ TAG]...
```

Figure 3. Command Summary Pop-up Window



Refer to the user guide for more details: https://ay1920s2-cs2103t-f10-1.github.io/main/UserGuide.html

```
Welcome to ResuMe!
You have summoned me - your trusty Assistant, probably because you are lost and do not know where to start.
But don't worry, because that is the whole point of my existence!

=========================================================================
I. User Profile
=========================================================================
First, let's kick-start your quest to creating the perfect resume by ensuring that the resume is truly YOURS!
Key in this command while replacing the details with your own personal information:
  --> me n/ YOUR_NAME p/ YOUR_PHONE e/ YOUR_EMAIL g/ YOUR_GITHUB

See the changes yet? That was easy, isn't it? And I promise you, that is as difficult as it can get when using ResuMe!
Similarly, you can edit your university, major, start and end time, as well as your current CAP with u/, m/, f/, t/ and c/ respectively.
Take some time to play around with this feature, before we move on to the next step.

(Hot tip: try dp/ FILE_PATH to update the profile picture to a picture of your choice!)

=========================================================================
II. Your Experiences
=========================================================================
Now, it's time to add in your first experience! An experience could be an internship that you had, a project that you worked on, or
any relevant skill that you wish to include in your resume.
```

Figure 4. Getting Started Pop-up Window

{ end of `help` section written by: Pham Thuy Duong }

{ start of `me` section written by: Nham Quoc Hung }

4.2. Editing user profile: `me`

You can edit your display profile here! It will then be updated accordingly in the user box. Feel free to edit only one or multiple fields at a time :)

Format: `me [dp/ FILE_PATH] [n/ NAME] [d/ DESCRIPTION] [p/ PHONE_NUMBER] [e/ EMAIL] [g/ GITHUB] [u/ UNIVERSITY] [m/ MAJOR] [f/ FROM] [t/ TO] [c/ CURRENT_CAP MAX_CAP]`

A user profile contains the following fields: `DisplayPicture`, `Name`, `Description`, `Phone`, `Email`, `Github`, `University`, `Major`, `From`, `To`, `CAP`.

- We note that these fields are your essential information which will be displayed in the PDF file of every generated resume. As such, each field comes with a format check which you need to fulfill in order to make a valid change.

Field	Format
Display Picture	Has to be a valid image file ending with .jpg, .png or .jpeg.

Field	Format
Name	Should only contain alphanumeric characters and spaces, and it should not be blank.
Description	Can take any characters and it should not be blank.
Phone	Should only contain numbers, and it should be at least 3 digits long.
Email	Must follow a valid email format.
Github	Must follow a valid github username format.
University	Should only contain alphanumeric characters and spaces, with a maximum of 50 characters and should not be blank.
Major	Should only contain alphanumeric characters and spaces, with a maximum of 50 characters and should not be blank.
From	Should be in the format of MM-yyyy.
To	Should be in the format of MM-yyyy and must not be an earlier date than From.
Cap	Must be in the format of "CURRENT_CAP MAX_CAP" where "CURRENT_CAP" is not greater than "MAX_CAP". Must contain only non-negative numbers.

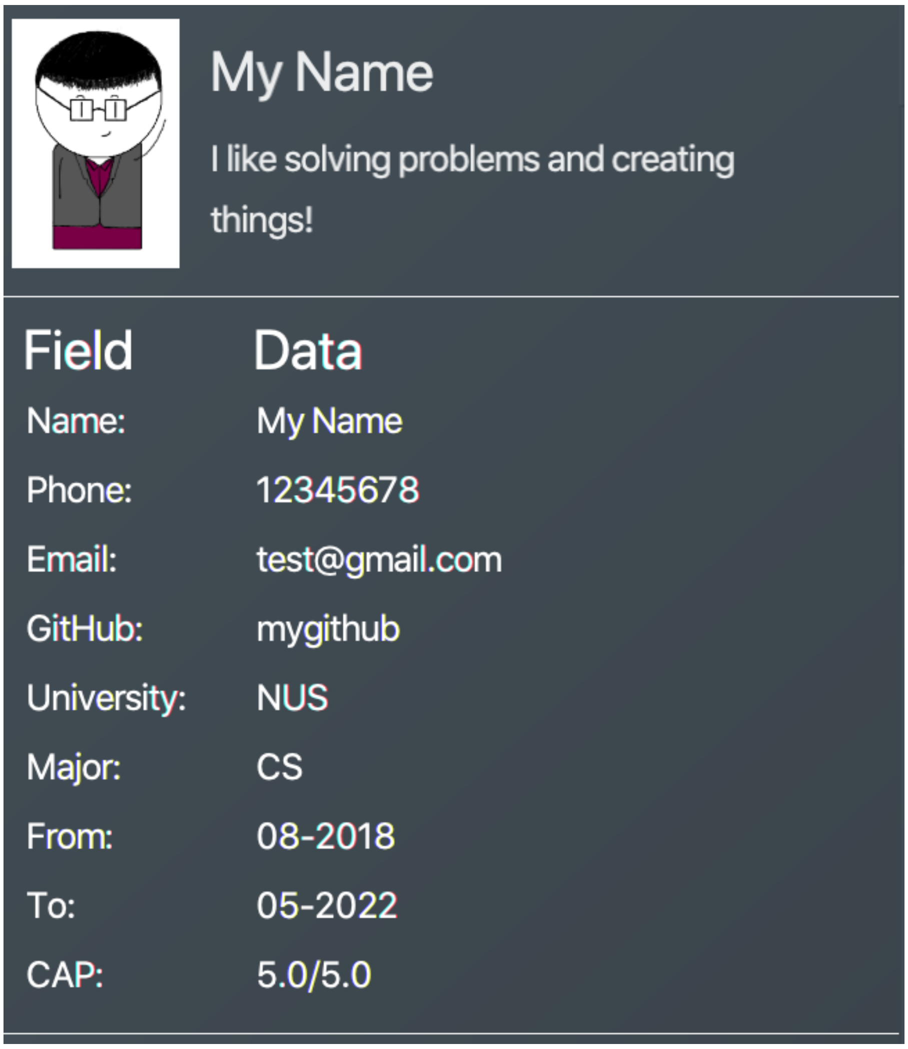
Example 1: Update user profile details

Try typing in the command box this command:

```
me n/ My Name p/ 12345678 e/ test@gmail.com d/ I like solving problems and creating things! g/ mygithub u/ NUS m/ CS f/ 08-2018 t/ 05-2022 c/ 5.0 5.0
```

Outcome:

The user box will be swiftly updated to showcase all your essential information:



My Name

I like solving problems and creating things!

Field	Data
Name:	My Name
Phone:	12345678
Email:	test@gmail.com
GitHub:	mygithub
University:	NUS
Major:	CS
From:	08-2018
To:	05-2022
CAP:	5.0/5.0

Figure 5. User Box with updated personal details

Example 2: Update user profile picture

Now, let's make your profile even more customisable. Follow the steps in one of these two links below if you do not know how to copy an full file path from your computer.

1. Mac: https://osxdaily.com/2013/06/19/copy-file-folder-path-mac-os-x/
2. Windows: https://www.laptopmag.com/articles/show-full-folder-path-file-explorer

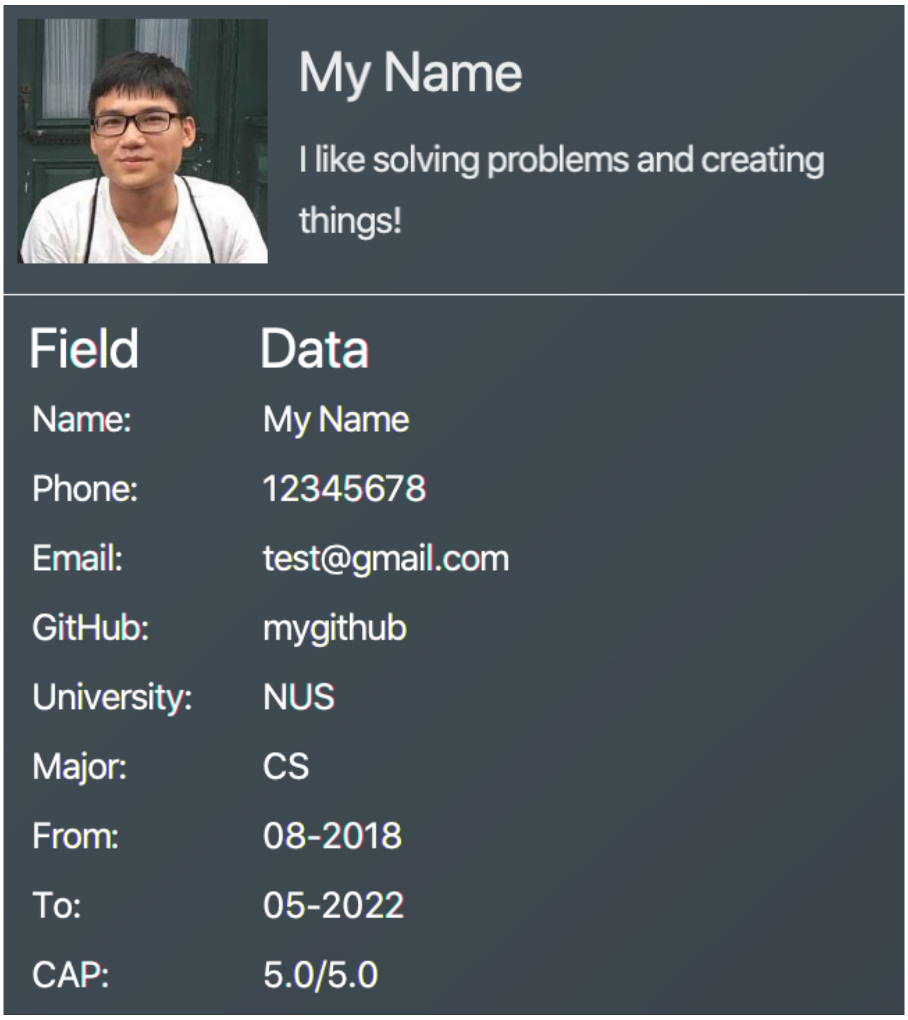
Afterwards, try a command similar to the one of the two below, depending on your operating system:

`me dp/ /Users/nhamquochung/Desktop/test.png` (for Mac)

`me dp/ C:\Users\Christian J. Welly\Pictures\chrisjwelly.jpg` (for Windows)

Outcome:

Tada! The user box will be updated to display your beautiful picture:



My Name

I like solving problems and creating things!

Field	Data
Name:	My Name
Phone:	12345678
Email:	test@gmail.com
GitHub:	mygithub
University:	NUS
Major:	CS
From:	08-2018
To:	05-2022
CAP:	5.0/5.0

Figure 6. User Box with updated profile picture

{ end of `me` section written by: Nham Quoc Hung }

{ start of `list` section written by: Nguyen Chi Hai }

4.3. Listing all items : `list`

Lists items in the storage.

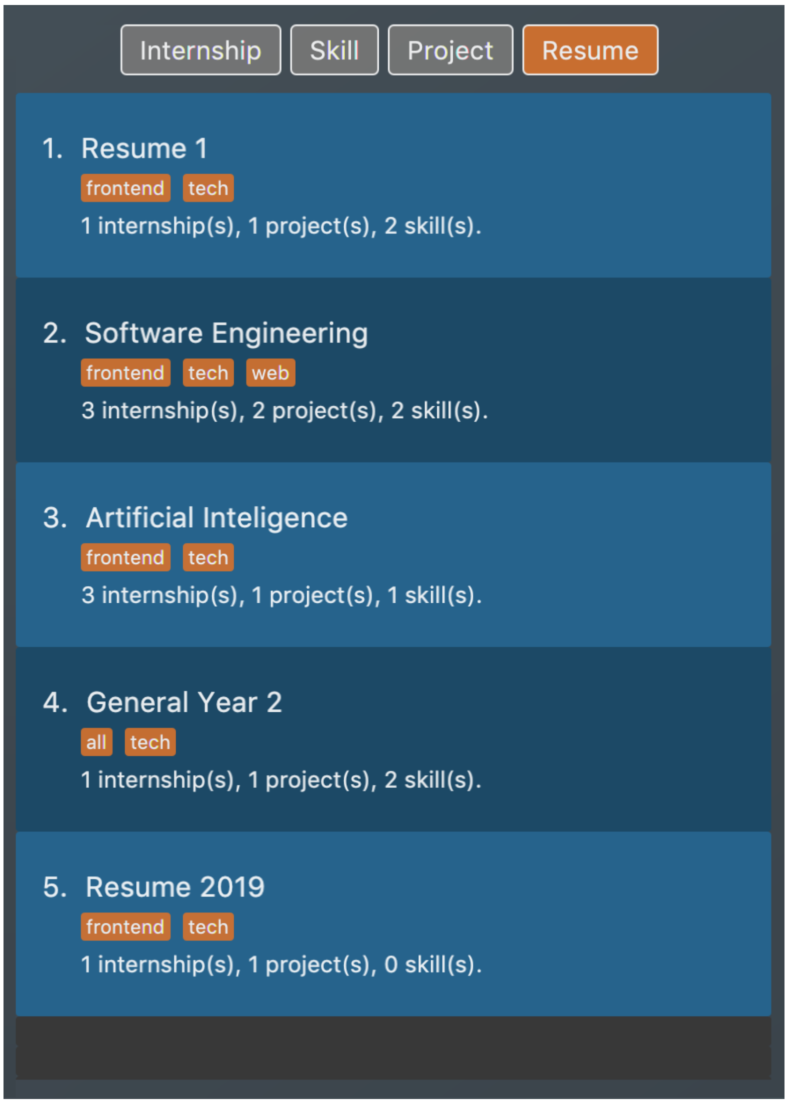
Format: `list i/ TYPE`

- Listed items are in short form, only showing their index, `name`, `tags` and a short summary. To view items in full details, use view.
- The type of items listed will light up in orange.

`note` list is always showing on the bottom left of the application. `list i/ note` command will show the entire `note` lis, this is to be used in conjunction with `find` for `note` to show all `note` items after you have filtered the list.

Examples:

- `list i/ res`: Lists all `resume` items.



1. Resume 1
frontend tech
1 internship(s), 1 project(s), 2 skill(s).

2. Software Engineering
frontend tech web
3 internship(s), 2 project(s), 2 skill(s).

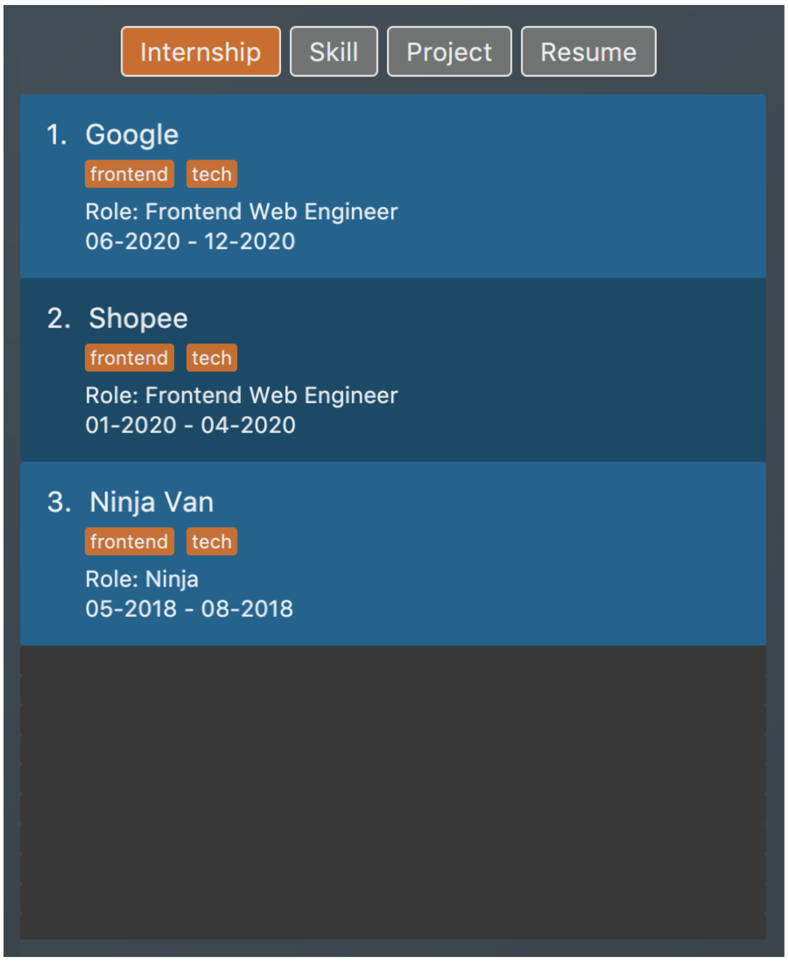
3. Artificial Inteligence
frontend tech
3 internship(s), 1 project(s), 1 skill(s).

4. General Year 2
all tech
1 internship(s), 1 project(s), 2 skill(s).

5. Resume 2019
frontend tech
1 internship(s), 1 project(s), 0 skill(s).

Figure 7. List of all resume items

- `list i/ int`: Lists all `internship` items.



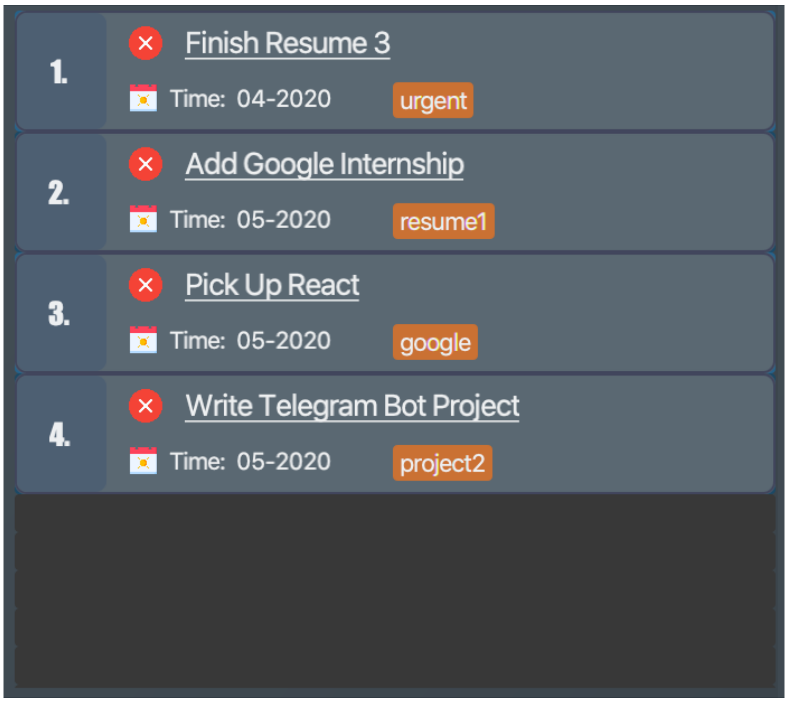
1. Google
frontend tech
Role: Frontend Web Engineer
06-2020 - 12-2020

2. Shopee
frontend tech
Role: Frontend Web Engineer
01-2020 - 04-2020

3. Ninja Van
frontend tech
Role: Ninja
05-2018 - 08-2018

Figure 8. List of all internship items

- `list i/ note`: Lists all `note` items.



1. Finish Resume 3
Time: 04-2020 urgent

2. Add Google Internship
Time: 05-2020 resume1

3. Pick Up React
Time: 05-2020 google

4. Write Telegram Bot Project
Time: 05-2020 project2

Figure 9. List of all note items

{ end of `list` section written by: Nguyen Chi Hai }

{ start of `add` section written by: Pham Thuy Duong }

4.4. Adding an item: `add`

Adds an item to the **ResuMe** application.

Format: `add i/ TYPE n/ NAME [ATTRIBUTE/ VALUE]… [#/ TAG]…`

The specific command syntax could be found in the table below:

Type	Format
Internship	`add i/ int n/ COMPANY_NAME r/ ROLE f/ FROM t/ TO d/ DESCRIPTION [#/ TAG]…`
Project	`add i/ proj n/ PROJECT_NAME t/ TIME w/ WEBSITE d/ DESCRIPTION [#/ TAG]…`
Skill	`add i/ ski n/ SKILL_NAME l/ LEVEL [#/ TAG]…`
Resume	`add i/ res n/ RESUME_NAME [#/ TAG]…`
Note	`add i/ note n/ NOTE_NAME t/ TIME [#/ TAG]…`

- An item could have any number of tags *(including 0)*.
- There are three possible `levels` for a `skill` item: `ADVANCED`, `INTERMEDIATE` and `BASIC`.
- A newly added note will be automatically marked as `undone`.

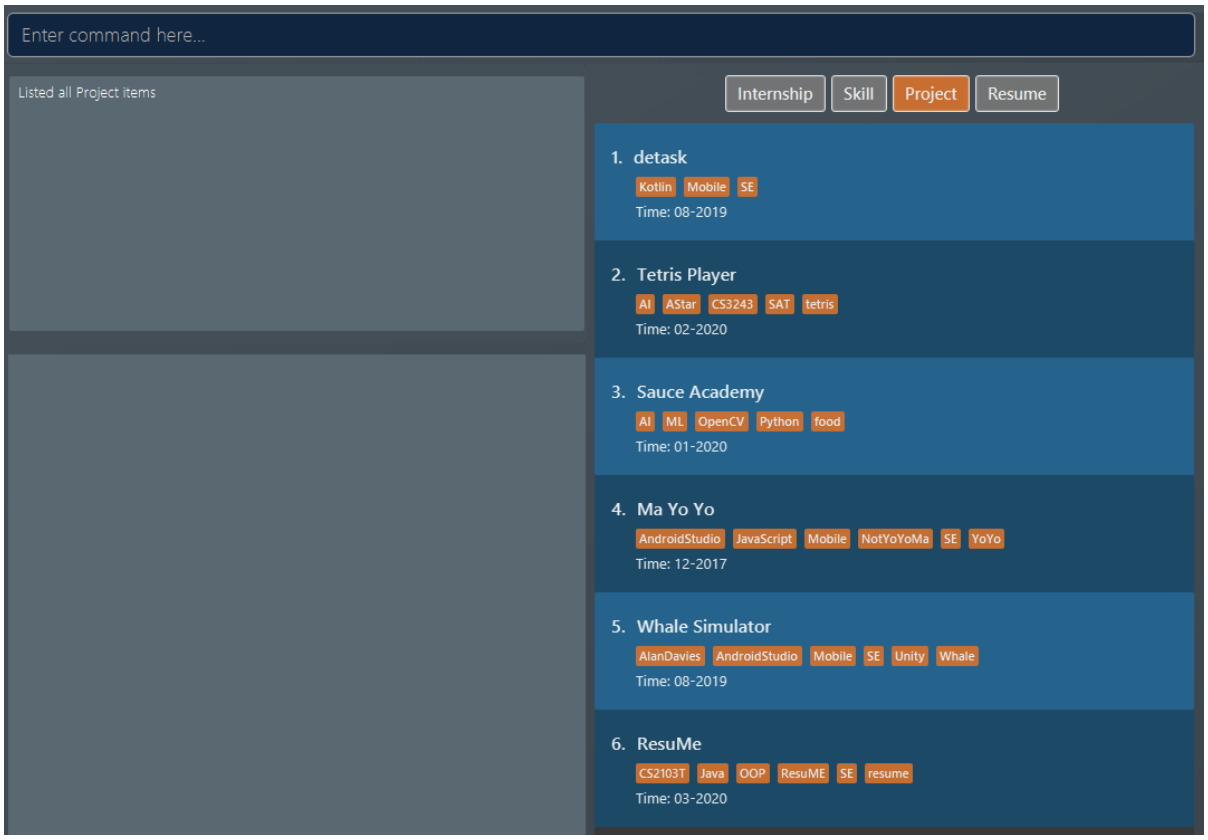
The description of an item could be written in one paragraph (with **full stop** and a **space** after each sentence). These sentences, when exported to `.pdf` format, will be automatically shown as individual bullet points! *(refer to rgen for more details)*

Example: Try typing in the command box these two commands one by one!

1. `list i/ proj`
2. `add i/ proj n/ Duke t/ 06-2020 w/ abc.github.io d/ For a little module named CS2103T. #/ java #/ tech`

Outcome:

1. All `project` items are listed in the list box.

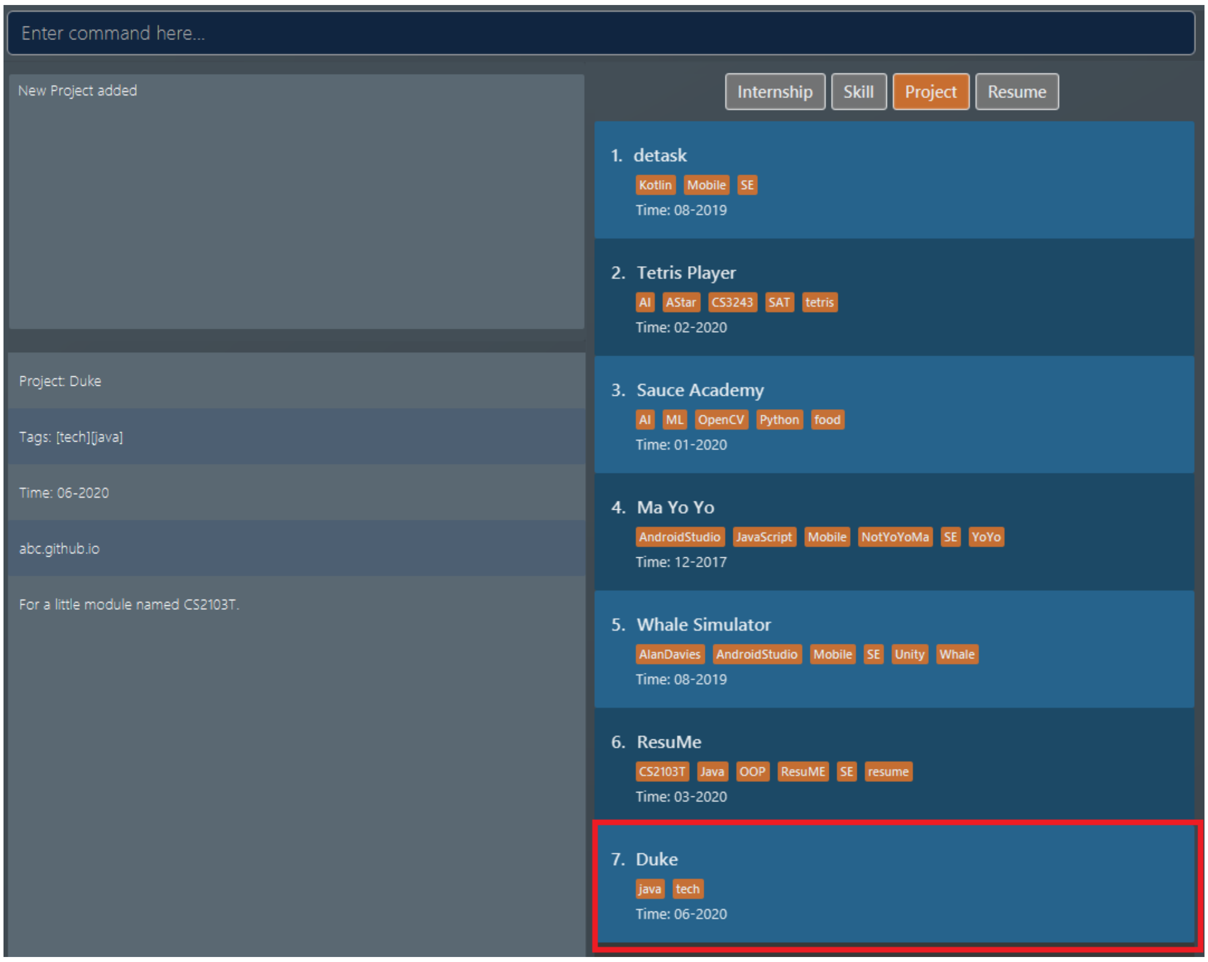

Enter command here...

Listed all Project items

1. detask
   Kotlin Mobile SE
   Time: 08-2019
2. Tetris Player
   AI AStar CS3243 SAT tetris
   Time: 02-2020
3. Sauce Academy
   AI ML OpenCV Python food
   Time: 01-2020
4. Ma Yo Yo
   AndroidStudio JavaScript Mobile NotYoYoMa SE YoYo
   Time: 12-2017
5. Whale Simulator
   AlanDavies AndroidStudio Mobile SE Unity Whale
   Time: 08-2019
6. ResuMe
   CS2103T Java OOP ResuME SE resume
   Time: 03-2020

Figure 10. List of all project items

2. A new `project` item named "Duke" with the specified fields is added. This item is automatically reflected in the list box.



New Project added

Project: Duke

Tags: [tech][java]

Time: 06-2020

abc.github.io

For a little module named CS2103T.

1. detask
Kotlin Mobile SE
Time: 08-2019

2. Tetris Player
AI AStar CS3243 SAT tetris
Time: 02-2020

3. Sauce Academy
AI ML OpenCV Python food
Time: 01-2020

4. Ma Yo Yo
AndroidStudio JavaScript Mobile NotYoYoMa SE YoYo
Time: 12-2017

5. Whale Simulator
AlanDavies AndroidStudio Mobile SE Unity Whale
Time: 08-2019

6. ResuMe
CS2103T Java OOP ResuME SE resume
Time: 03-2020

7. Duke
java tech
Time: 06-2020

Figure 11. Application view after project "Duke" is added

{ end of `add` section written by: Pham Thuy Duong }

{ start of `edit` section written by: Christian James Welly }

4.5. Editing an item : `edit`

Edits an existing item in the **ResuMe** application.

`edit` is a different command from `redit`. Please visit FAQ for more information.

Format: `edit INDEX i/ TYPE [ATTRIBUTE/ VALUE]… [#/ TAG]…`

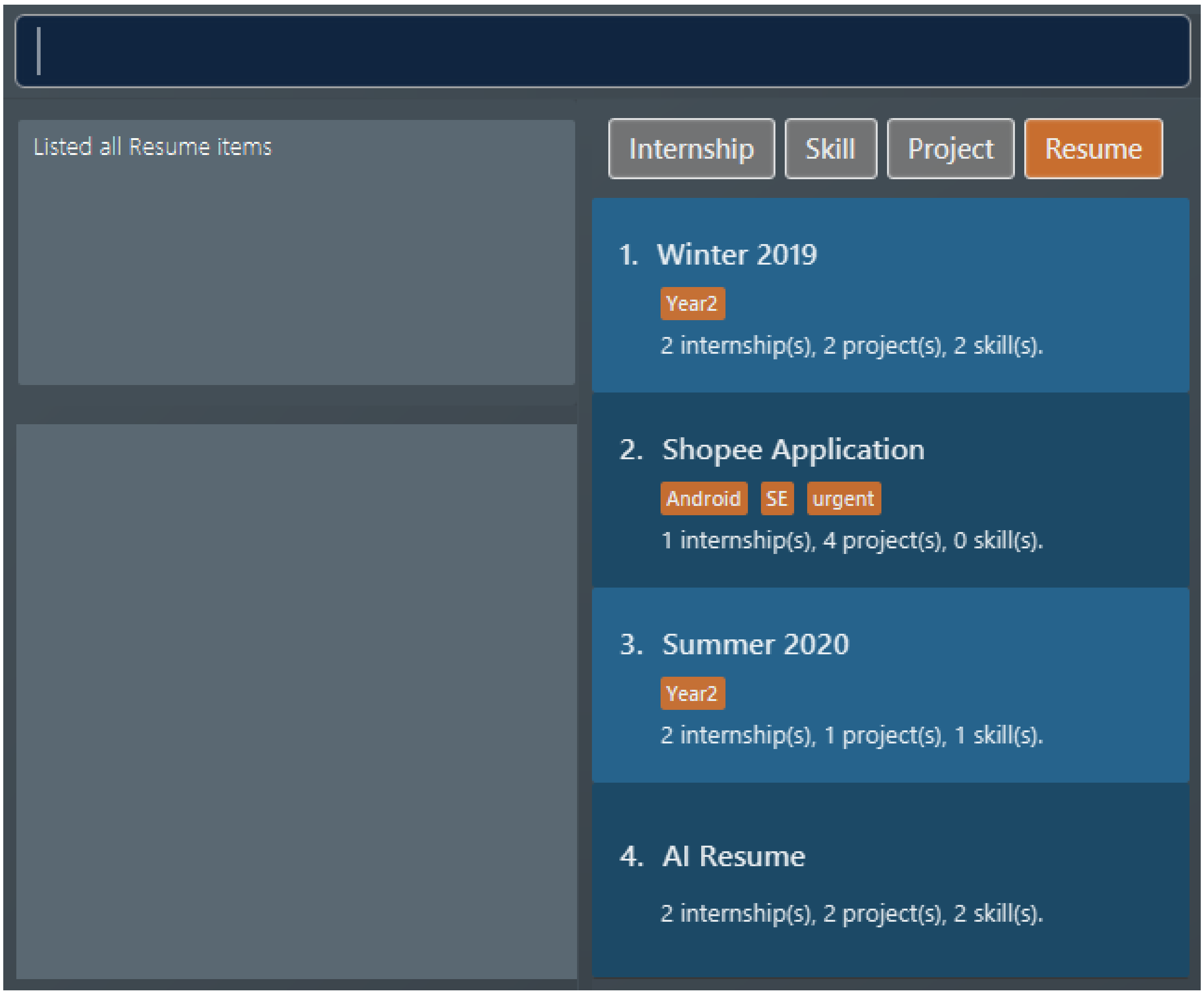
Type	Format
Internship	`edit INDEX i/ int [n/ COMPANY_NAME] [r/ ROLE] [f/ FROM] [t/ TO] [d/ DESCRIPTION] [#/ TAG]…`
Project	`edit INDEX i/ proj [n/ PROJECT_NAME] [t/ TIME] [w/ WEBSITE] [d/ DESCRIPTION] [#/ TAG]…`
Skill	`edit INDEX i/ ski [n/ SKILL_NAME] [l/ LEVEL] [#/ TAG]…`
Resume	`edit INDEX i/ res [n/ RESUME_NAME] [#/ TAG]…`
Note	`edit INDEX i/ note [n/ NOTE_NAME] [t/ TIME] [#/ TAG]…`

Example 1: Try typing in the command box these two commands one by one!

1. `list i/ res`
2. `edit 1 i/ res n/ Software Engineering Resume #/ agile`

Outcome:

1. The first command lists all `resume` items in the list box. Assuming you want to edit the `resume` item at index 1.



Listed all Resume items

1. Winter 2019
Year2
2 internship(s), 2 project(s), 2 skill(s).

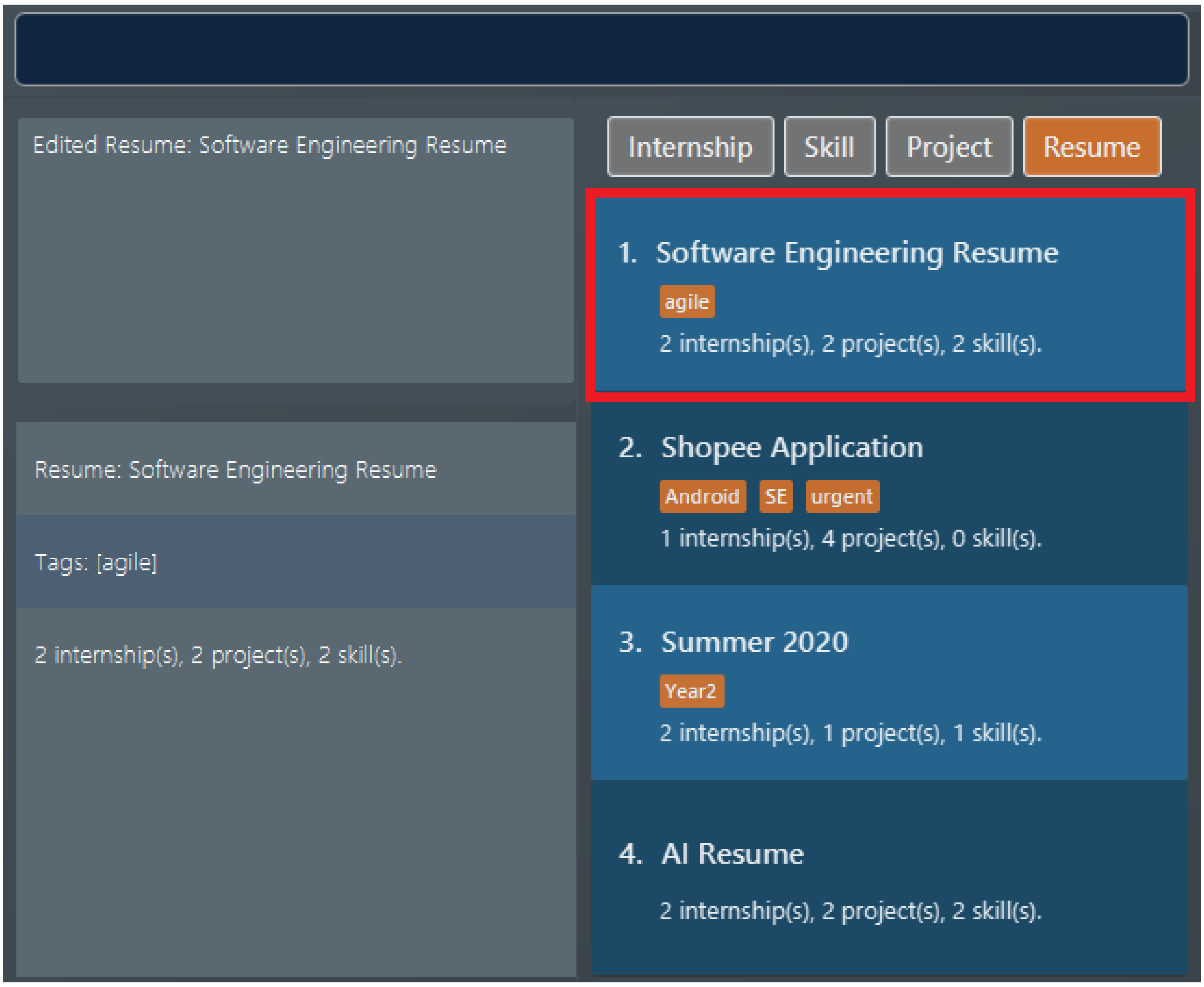
2. Shopee Application
Android SE urgent
1 internship(s), 4 project(s), 0 skill(s).

3. Summer 2020
Year2
2 internship(s), 1 project(s), 1 skill(s).

4. AI Resume
2 internship(s), 2 project(s), 2 skill(s).

Figure 12. List of all resume items

2. The second command edits the `resume` item at index 1. It has its name changed from *"Winter 2019"* to *"Software Engineering Resume"* and its `tag` set to *"agile"*.


Edited Resume: Software Engineering Resume

Resume: Software Engineering Resume

Tags: [agile]

2 internship(s), 2 project(s), 2 skill(s).

1. Software Engineering Resume
agile
2 internship(s), 2 project(s), 2 skill(s).

2. Shopee Application
Android SE urgent
1 internship(s), 4 project(s), 0 skill(s).

3. Summer 2020
Year2
2 internship(s), 1 project(s), 1 skill(s).

4. AI Resume
2 internship(s), 2 project(s), 2 skill(s).

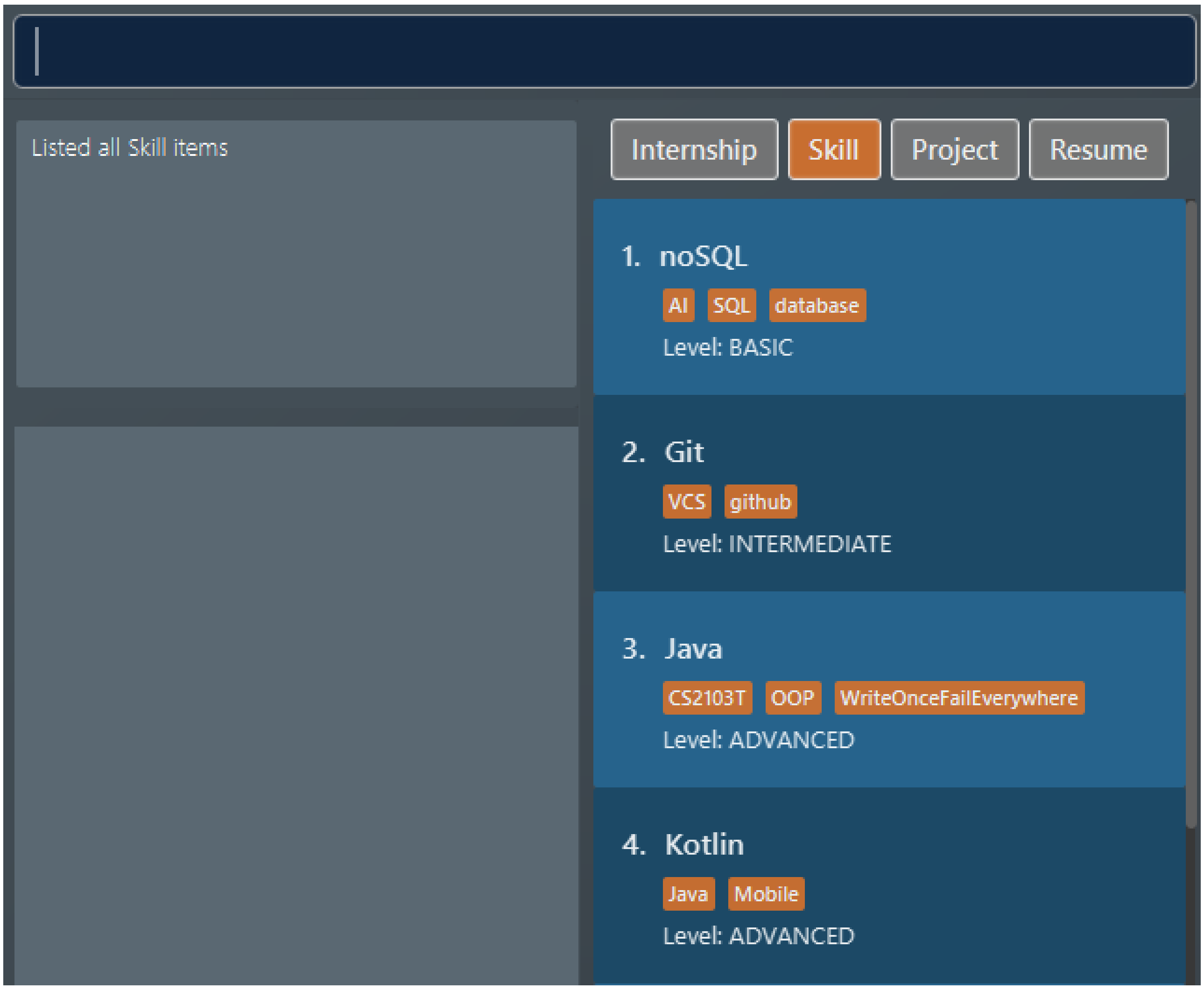
Figure 13. Application view after editing the resume item at index 1

Example 2: Here is another way you can use `edit`. This time, we will try an example using `skill` items. Try typing in the command box these two commands one by one!

1. `list i/ ski`
2. `edit 2 i/ ski l/ ADVANCED #/`

Outcome:

1. The first command lists all `skill` items in the list box. Assuming you want to edit the `skill` item at index 2.



Listed all Skill items

1. noSQL
AI SQL database
Level: BASIC

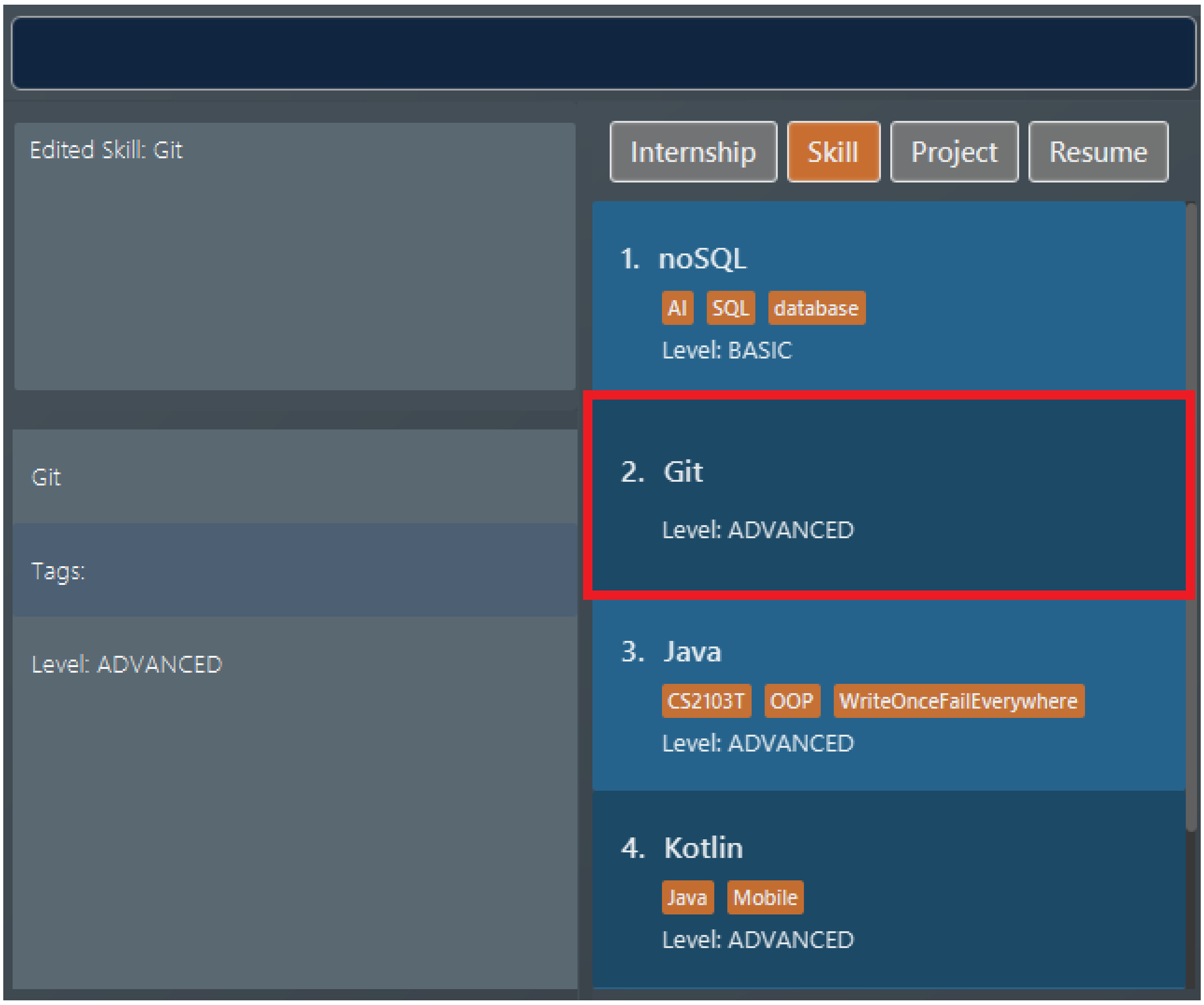
2. Git
VCS github
Level: INTERMEDIATE

3. Java
CS2103T OOP WriteOnceFailEverywhere
Level: ADVANCED

4. Kotlin
Java Mobile
Level: ADVANCED

Figure 14. List of all skill items

2. The second command edits the `skill` item at index 2. It has its `level` changed to from `INTERMEDIATE` to `ADVANCED` and all its existing `tags` cleared.



Edited Skill: Git

Git

Tags:

Level: ADVANCED

1. noSQL
AI SQL database
Level: BASIC

2. Git
Level: ADVANCED

3. Java
CS2103T OOP WriteOnceFailEverywhere
Level: ADVANCED

4. Kotlin
Java Mobile
Level: ADVANCED

Figure 15. Application view after editing the skill item at index 2

{ end of `edit` section written by: Christian James Welly }

{ start of `find` section written by: Nham Quoc Hung }

4.6. Finding items by name: `find`

Finds items of a specific `type` in the corresponding list of items whose names contain the specified keyword(s).

Format: `find KEYWORD [MORE_KEYWORDS]… i/ TYPE`

The specific command syntax could be found in the table below:

Type	Format
Internship	`find KEYWORD [MORE_KEYWORDS]… i/ int`
Project	`find KEYWORD [MORE_KEYWORDS]… i/ proj`
Skill	`find KEYWORD [MORE_KEYWORDS]… i/ ski`

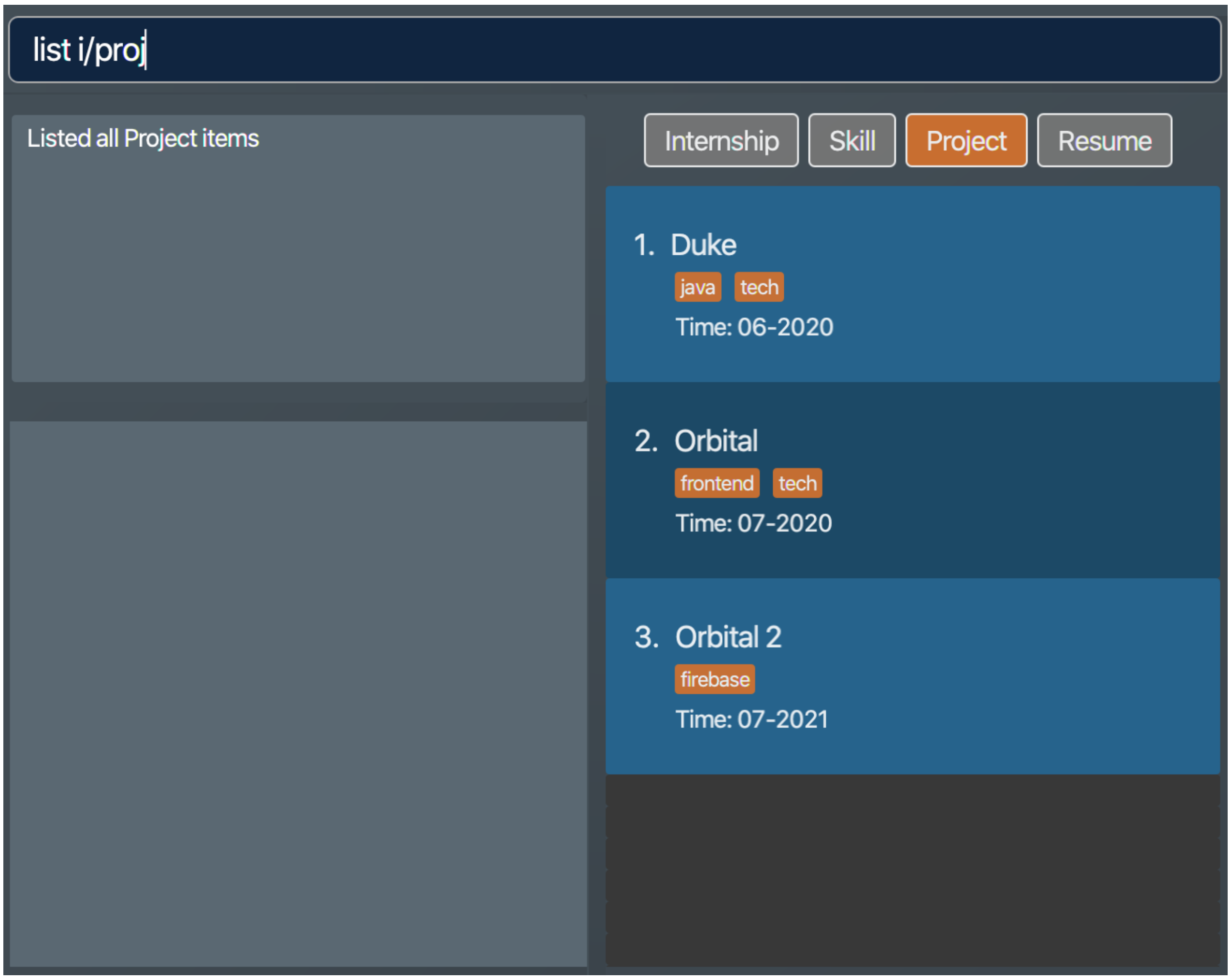
Type	Format
Resume	`find KEYWORD [MORE_KEYWORDS]… i/ res`
Note	`find KEYWORD [MORE_KEYWORDS]… i/ note`

Example: Try typing in the command box these commands one by one!

1. `list i/ proj`
2. `find Orbital i/ proj`

Outcome

1. All `project` items are listed in the list box. A sample project list is shown below.

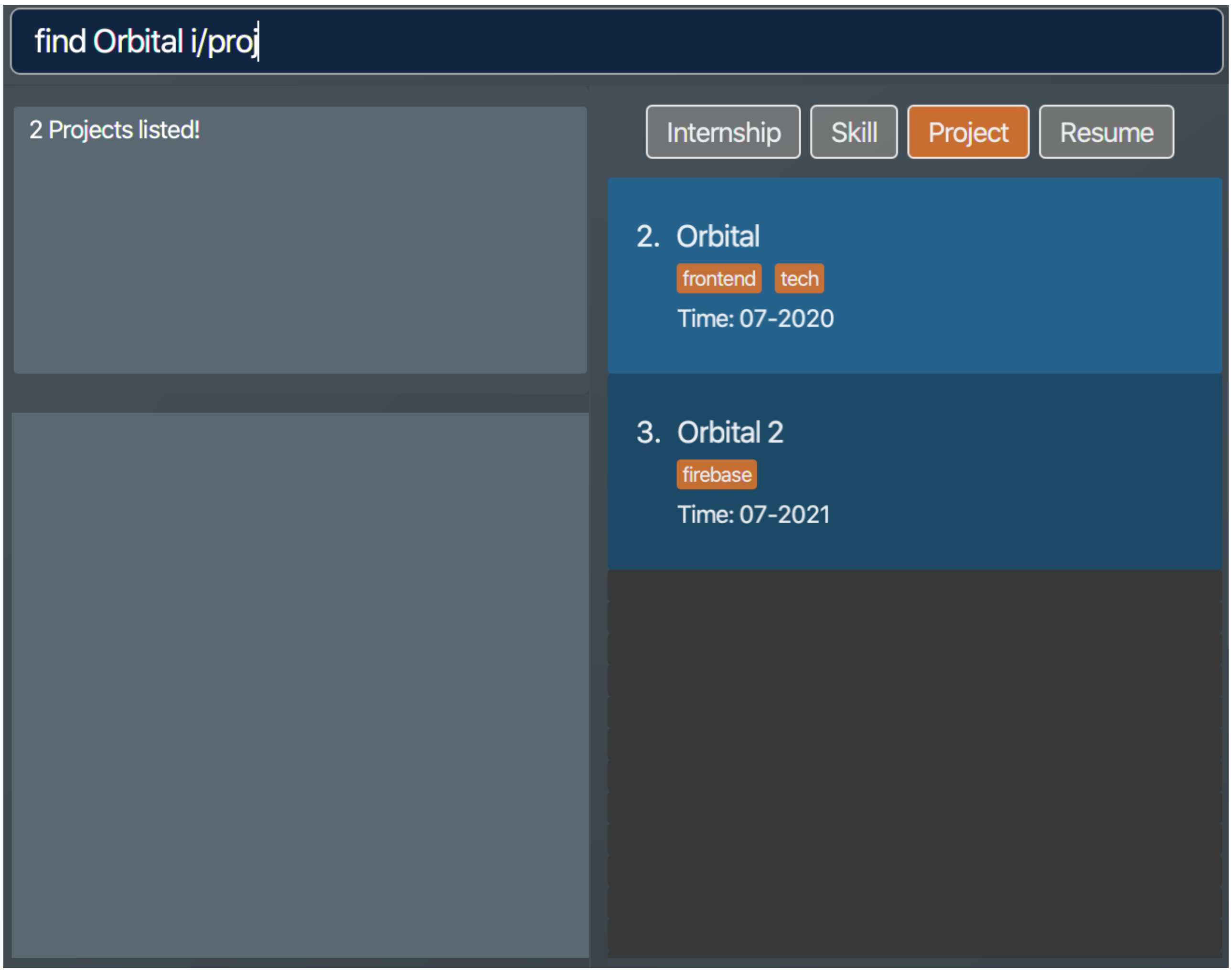


```
list i/proj
```

Listed all Project items

Figure 16. List of all project items

2. `Projects` whose names match keywords are listed in the list box.


find Orbital i/proj

2 Projects listed!

2. Orbital
frontend tech
Time: 07-2020

3. Orbital 2
firebase
Time: 07-2021

Figure 17. Filtered list of project items whose names contain "Orbital"

{ end of `find` section written by: Nham Quoc Hung }

{ start of `delete` section written by: Nguyen Minh Hoang}

4.7. Deleting an item : `delete`

Deletes an existing item from the **ResuMe** application.

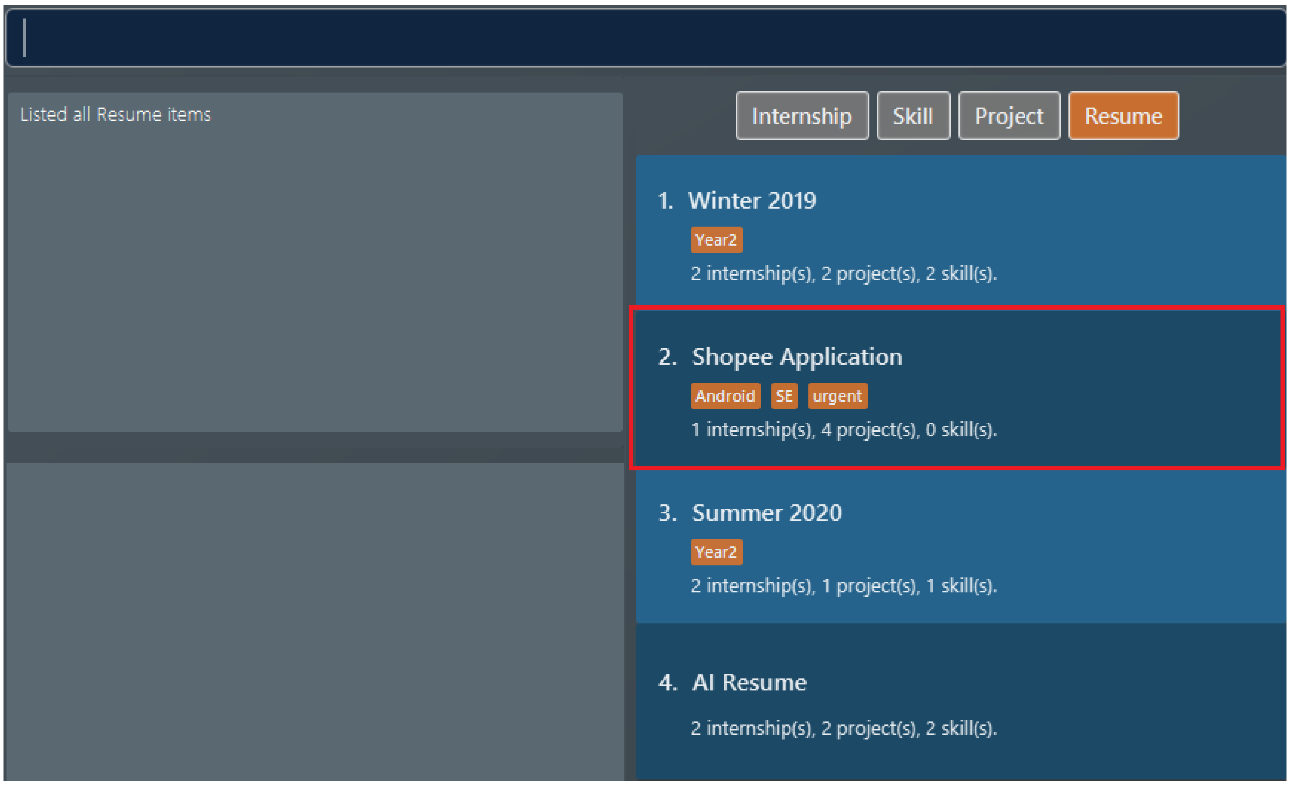
Format: `delete INDEX i/ TYPE`

- The `INDEX` argument comes before the `TYPE` argument.
- Deleting an item will set the list box to display items of that type.
- All `resume` items that contain the deleted item will also be updated to reflect the change.

Use `list` command to navigate to the correct list to ensure that 1) the item exists, and 2) you know the correct `index` for deletion. Or just fall back on `undo` if you accidentally delete the wrong item.

Example:

1. Suppose we start with the following original list of `resume` items, as seen using the command `list i/ res`. Pay attention to the second `resume` item.



Listed all Resume items

1. Winter 2019
Year2
2 internship(s), 2 project(s), 2 skill(s).

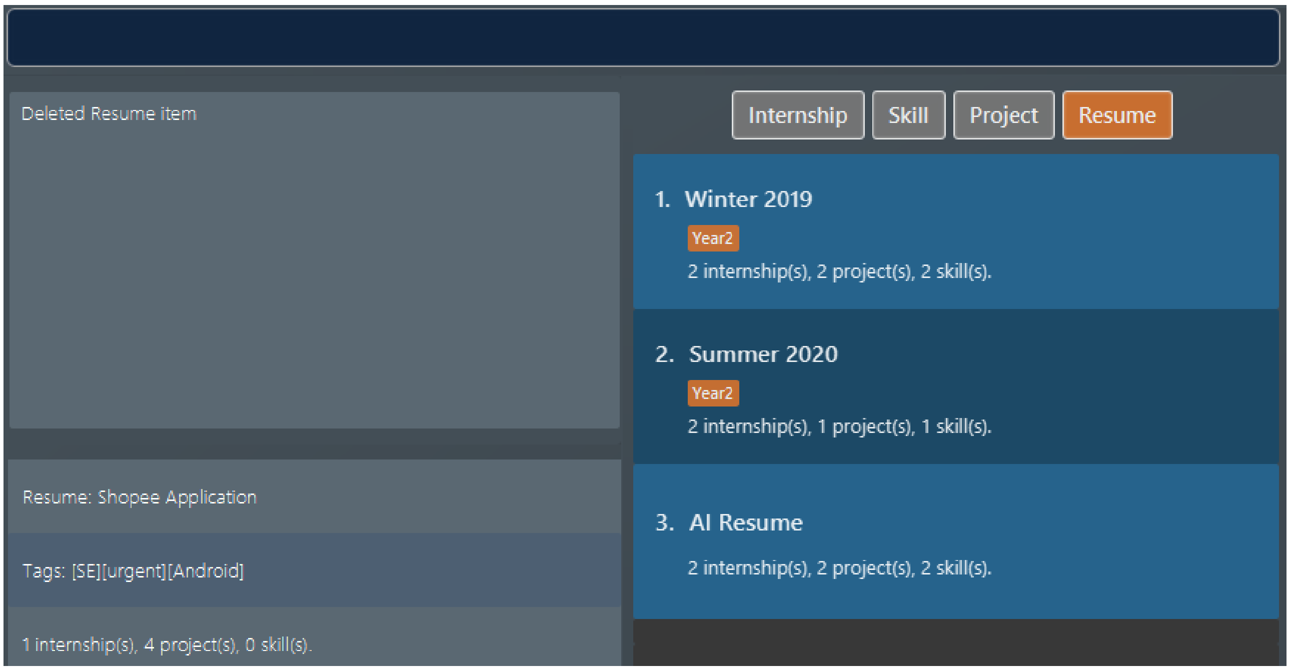
2. Shopee Application
Android SE urgent
1 internship(s), 4 project(s), 0 skill(s).

3. Summer 2020
Year2
2 internship(s), 1 project(s), 1 skill(s).

4. AI Resume
2 internship(s), 2 project(s), 2 skill(s).

Figure 18. List of all resume items

2. We delete the 2nd `resume` with `delete 2 i/ res` .



Deleted Resume item

Resume: Shopee Application

Tags: [SE][urgent][Android]

1 internship(s), 4 project(s), 0 skill(s).

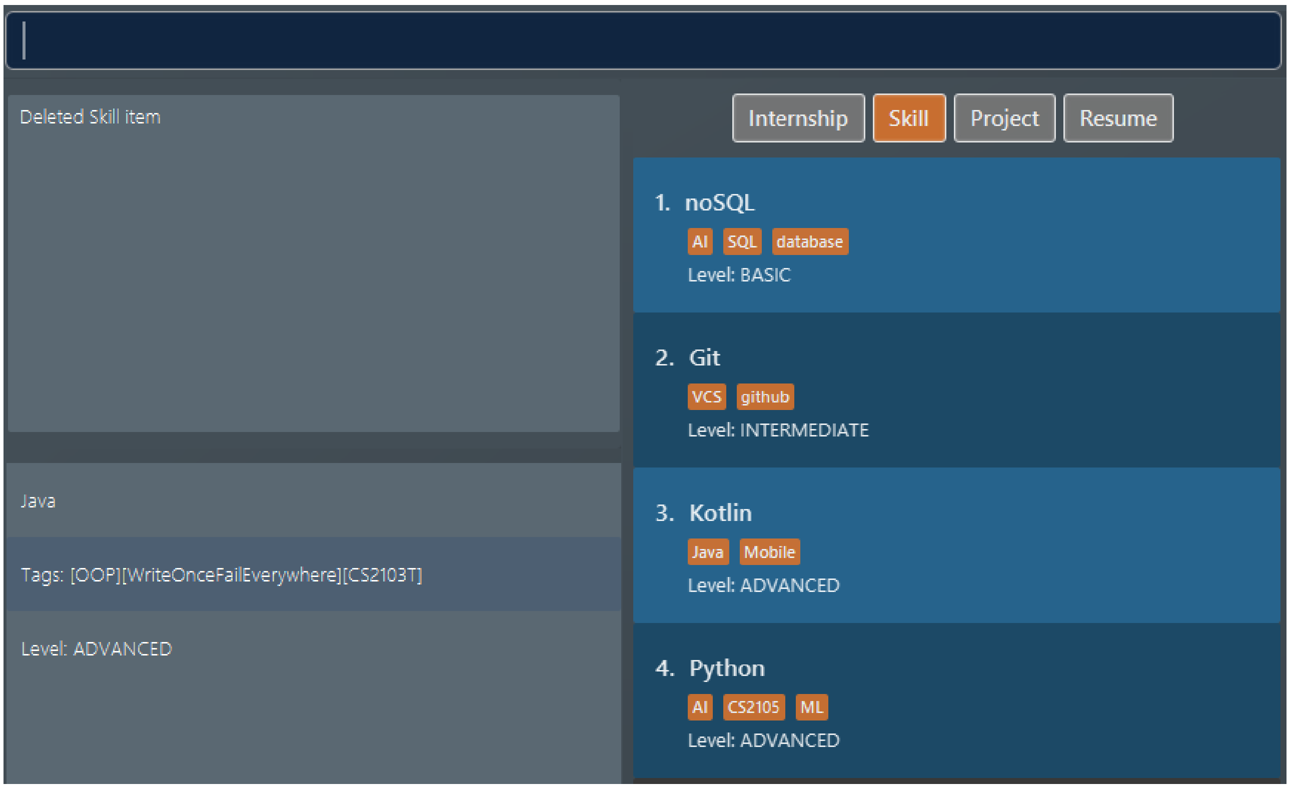
1. Winter 2019
Year2
2 internship(s), 2 project(s), 2 skill(s).

2. Summer 2020
Year2
2 internship(s), 1 project(s), 1 skill(s).

3. AI Resume
2 internship(s), 2 project(s), 2 skill(s).

Figure 19. Application view after deleting the resume item at index 2

3. We delete the 3rd `skill` with `delete 3 i/ ski` . Do note how it is not a must for the list box to display `skill` items for deletion to proceed. After the deletion the list box is set to show `skill` items.


Deleted Skill item

Java

Tags: [OOP][WriteOnceFailEverywhere][CS2103T]

Level: ADVANCED

1. noSQL
AI SQL database
Level: BASIC

2. Git
VCS github
Level: INTERMEDIATE

3. Kotlin
Java Mobile
Level: ADVANCED

4. Python
AI CS2105 ML
Level: ADVANCED

Figure 20. Application view after deleteing the skill item at index 3

{ end of `delete` section written by: Nguyen Minh Hoang}

{ start of `sort` section written by: Nguyen Minh Hoang}

4.8. Sorting an item list : `sort`

Sorts an item list in the **ResuMe** application.

Format: `sort i/ TYPE order/ SORT_ORDER [reverse/ TRUE_OR_FALSE]`

- The default sort order is ascending. You can specify `reverse/` to reverse this order. So use `reverse/ true` if you want to sort in descending order, and `reverse/ false` if you just like to type.
- Sorting an item list will set the list box to display items of that type.

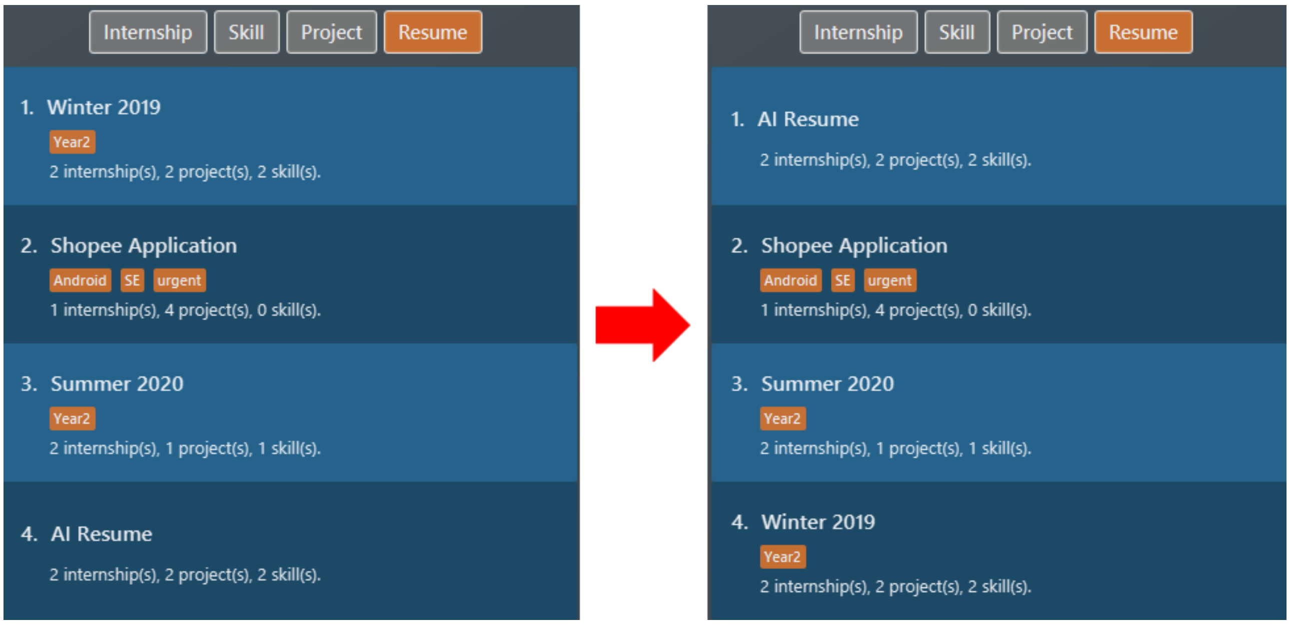
- `redit` can be very potent when used in conjunction with `sort`. Sort internships by latest, then call `redit` with `int/ 1 2 3` to add the three most recent internships to a resume. So easy!

- For the selected `TYPE`, the list of items of that type will be sorted according to `SORT_ORDER` (see the note above for what `reverse/` does).
- Below is the list of criteria by which different item types can be sorted:

Item type	`SORT_ORDER`	**Criterion**
`internship`	`name`	Name of `internship`
`internship`	`time`	Start date of `internship`
`project`	`name`	Name of `project`
`project`	`time`	Date of `project`
`skill`	`name`	Name of `skill`
`skill`	`level`	Level of proficiency of `skill`
`resume`	`name`	Name of `resume`
`note`	`name`	Name of `note`
`note`	`time`	Deadline of `note`

Example:

1. We sort all `resume` items by their name in ascending order with `sort i/ res order/ name`.



1. Winter 2019
Year2
2 internship(s), 2 project(s), 2 skill(s).

2. Shopee Application
Android SE urgent
1 internship(s), 4 project(s), 0 skill(s).

3. Summer 2020
Year2
2 internship(s), 1 project(s), 1 skill(s).

4. AI Resume
2 internship(s), 2 project(s), 2 skill(s).

1. AI Resume
2 internship(s), 2 project(s), 2 skill(s).

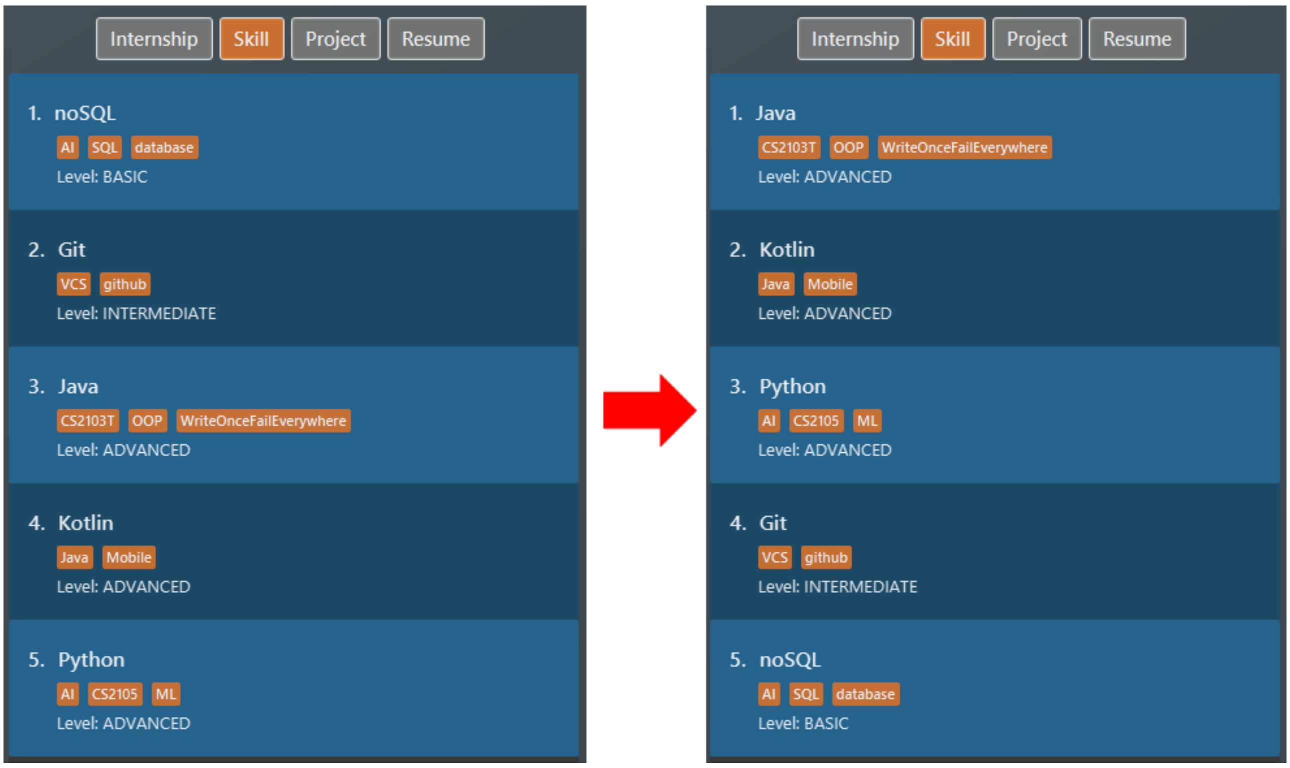
2. Shopee Application
Android SE urgent
1 internship(s), 4 project(s), 0 skill(s).

3. Summer 2020
Year2
2 internship(s), 1 project(s), 1 skill(s).

4. Winter 2019
Year2
2 internship(s), 2 project(s), 2 skill(s).

Figure 21. List of resume items sorted by name in ascending order

2. We sort all `skill` items in descending level of proficiency with `sort i/ ski order/ level reverse/ true`.



1. noSQL
AI SQL database
Level: BASIC

2. Git
VCS github
Level: INTERMEDIATE

3. Java
CS2103T OOP WriteOnceFailEverywhere
Level: ADVANCED

4. Kotlin
Java Mobile
Level: ADVANCED

5. Python
AI CS2105 ML
Level: ADVANCED

1. Java
CS2103T OOP WriteOnceFailEverywhere
Level: ADVANCED

2. Kotlin
Java Mobile
Level: ADVANCED

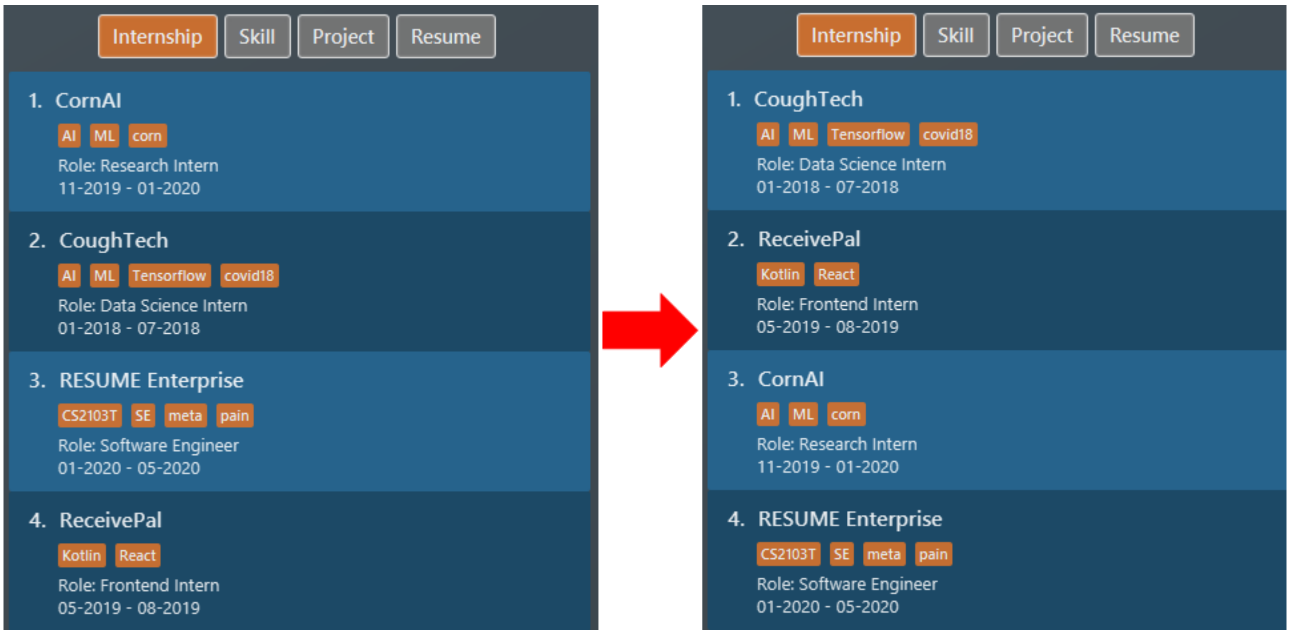
3. Python
AI CS2105 ML
Level: ADVANCED

4. Git
VCS github
Level: INTERMEDIATE

5. noSQL
AI SQL database
Level: BASIC

Figure 22. List of skill items sorted by level in descending order

3. We sort all `internship` by their start date in ascending order with `sort i/ int order/ time reverse/ false`, specifying `reverse/` this time because we really like to type.



1. CornAI
AI ML corn
Role: Research Intern
11-2019 - 01-2020

2. CoughTech
AI ML Tensorflow covid18
Role: Data Science Intern
01-2018 - 07-2018

3. RESUME Enterprise
CS2103T SE meta pain
Role: Software Engineer
01-2020 - 05-2020

4. ReceivePal
Kotlin React
Role: Frontend Intern
05-2019 - 08-2019

1. CoughTech
AI ML Tensorflow covid18
Role: Data Science Intern
01-2018 - 07-2018

2. ReceivePal
Kotlin React
Role: Frontend Intern
05-2019 - 08-2019

3. CornAI
AI ML corn
Role: Research Intern
11-2019 - 01-2020

4. RESUME Enterprise
CS2103T SE meta pain
Role: Software Engineer
01-2020 - 05-2020

Figure 23. List of all note items sorted by deadline in ascending order

{ end of `sort` section written by: Nguyen Minh Hoang}

{ start of `view` section written by: Christian James Welly }

4.9. Viewing an item in details: `view`

Shows the detailed view of an item.

Format: `view INDEX i/ TYPE`

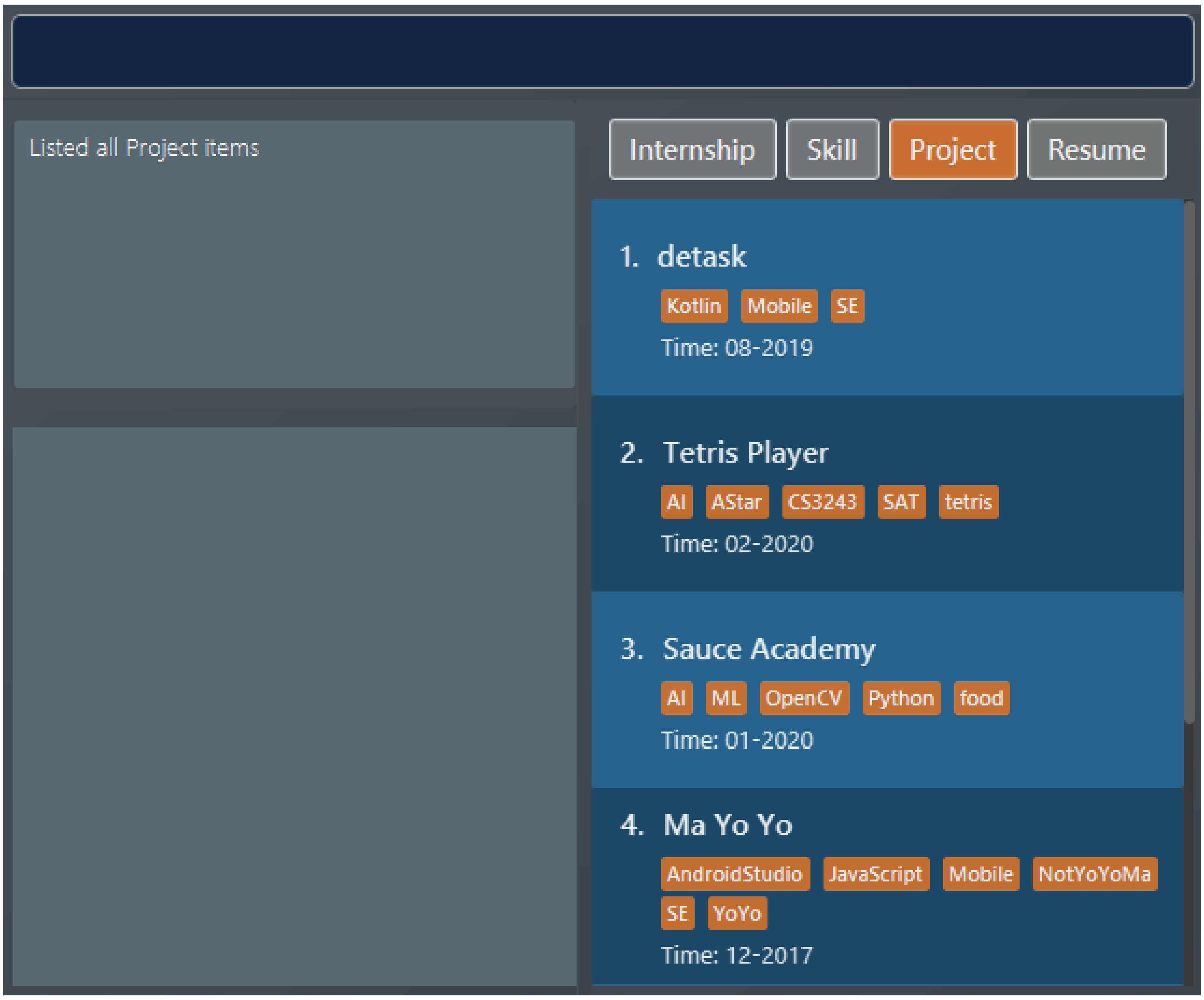
Certain details of specific items, for example the `website` and `description` of a `project` can only be viewed by using this command.

Example: Try typing in the command box these two commands one by one!

1. `list i/ proj`
2. `view 1 i/ proj`

Outcome:

1. The first command lists out all the `project` items in the list box. Assuming you want to view the details of the `project` item at index 1.



Listed all Project items

1. detask
Kotlin Mobile SE
Time: 08-2019

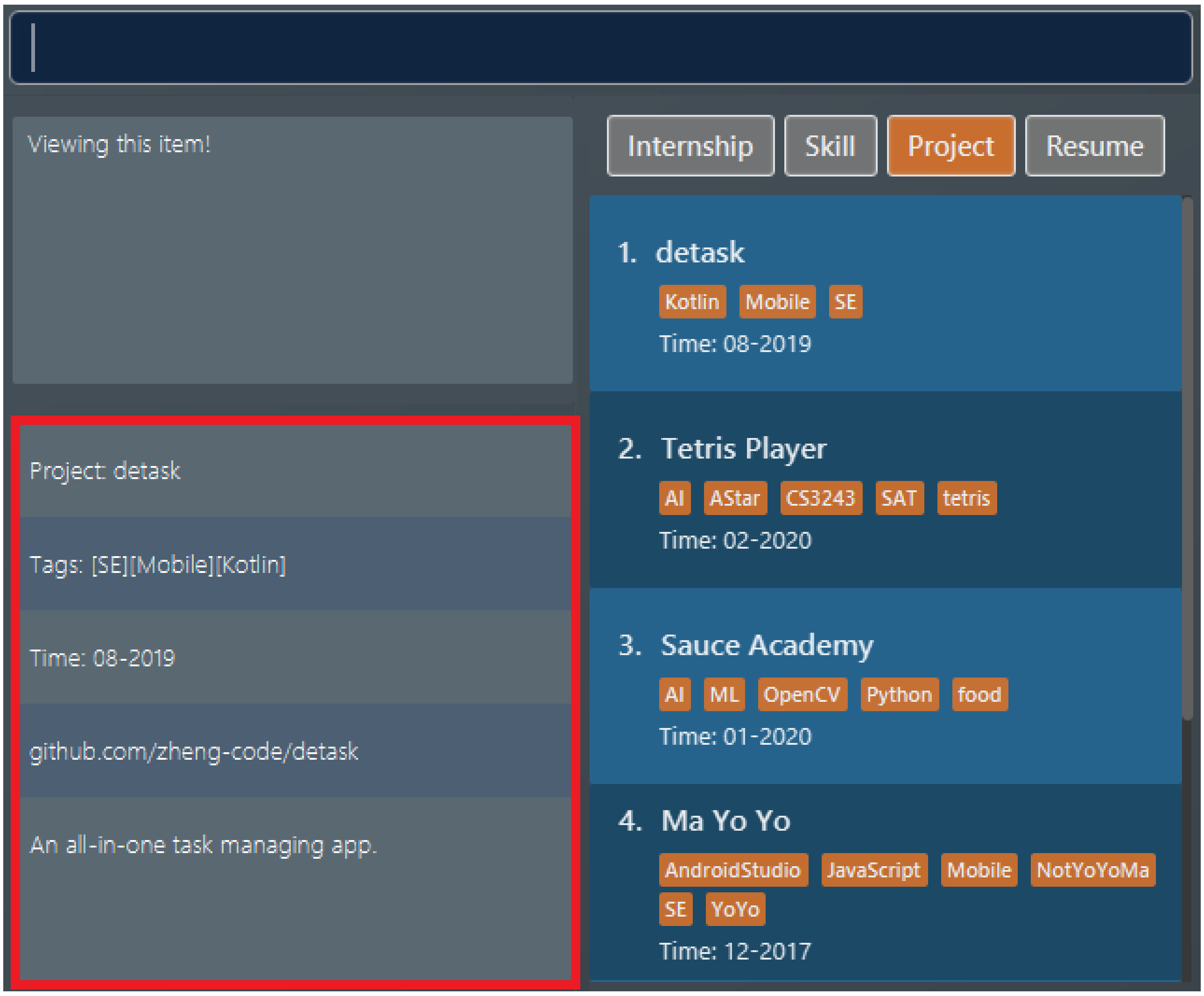
2. Tetris Player
AI AStar CS3243 SAT tetris
Time: 02-2020

3. Sauce Academy
AI ML OpenCV Python food
Time: 01-2020

4. Ma Yo Yo
AndroidStudio JavaScript Mobile NotYoYoMa SE YoYo
Time: 12-2017

Figure 24. List of all project items

2. The second command shows full details of the `project` at index 1, displayed in the view box. These includes details that were not shown in the list box, such as its `website` (*"github.com/zheng-code/detask"*) and its `description` (*"An all-in-one task managing app."*).



Project: detask

Tags: [SE][Mobile][Kotlin]

Time: 08-2019

github.com/zheng-code/detask

An all-in-one task managing app.

1. detask
Kotlin Mobile SE
Time: 08-2019

2. Tetris Player
AI AStar CS3243 SAT tetris
Time: 02-2020

3. Sauce Academy
AI ML OpenCV Python food
Time: 01-2020

4. Ma Yo Yo
AndroidStudio JavaScript Mobile NotYoYoMa SE YoYo
Time: 12-2017

Figure 25. Updated View Box with the details of the project item at index 1

{ end of `view` section written by: Christian James Welly }

{ start of `redit` section written by: Christian James Welly }

4.10. Modifying the content a resume: `redit`

Modifies the content of a `resume` to contain the items specified in the command.

`redit` is a different command from `edit`. Please visit FAQ for more information.

Format: `redit RESUME_INDEX TYPE/ [ITEM_INDEX…] [MORE_TYPE/ [ITEM_INDEX…]]…`

If you find it a hassle to manually enter items into the `resume`, you can consider using tagpull!

- For each `TYPE`, existing items will be updated to the input items.
- You can add multiple items of a certain type to a `resume` by chaining `ITEM_INDEX` after `TYPE/`. For example `proj/ 3 6`, will add `project` item of indices 3 and 6 to the `resume`.

- You can remove all items of type `TYPE` by typing `TYPE/` without specifying any `ITEM_INDEX` after it.

Example 1: Adding items into a new resume

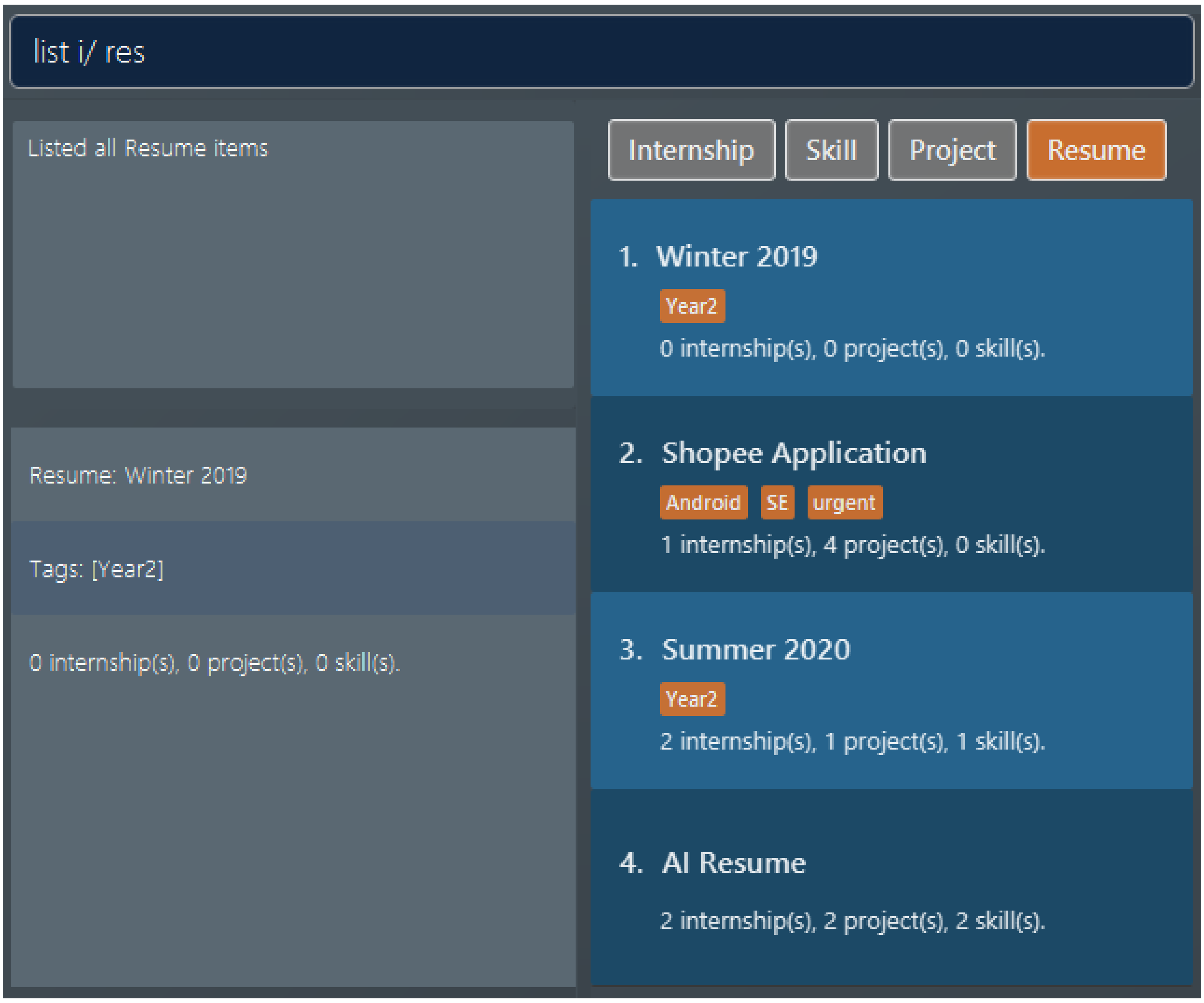
Before you try this example, please ensure you have added at least one `internship`, one `project`, and two `skill` items. You may verify for their presence by using the `list` command.

Try typing in these commands:

1. `list i/ res`
2. `redit 1 int/ 1 proj/ 1 ski/ 1 2`

Outcome:

1. The first command lists out all `resume` items. Assuming you want to modify the contents of the first `resume` in the list box.



```
list i/ res
```

Listed all Resume items

Resume: Winter 2019

Tags: [Year2]

0 internship(s), 0 project(s), 0 skill(s).

1. Winter 2019
Year2
0 internship(s), 0 project(s), 0 skill(s).

2. Shopee Application
Android SE urgent
1 internship(s), 4 project(s), 0 skill(s).

3. Summer 2020
Year2
2 internship(s), 1 project(s), 1 skill(s).

4. AI Resume
2 internship(s), 2 project(s), 2 skill(s).

Figure 26. List of all resume items

2. The second command modifies the `resume` at index 1, named "Winter 2019". The `resume` now contains the `internship` item at index 1, `project` item at index 1, and `skill` items at indices 1 and 2. The following screenshot illustrates what happens after this command:

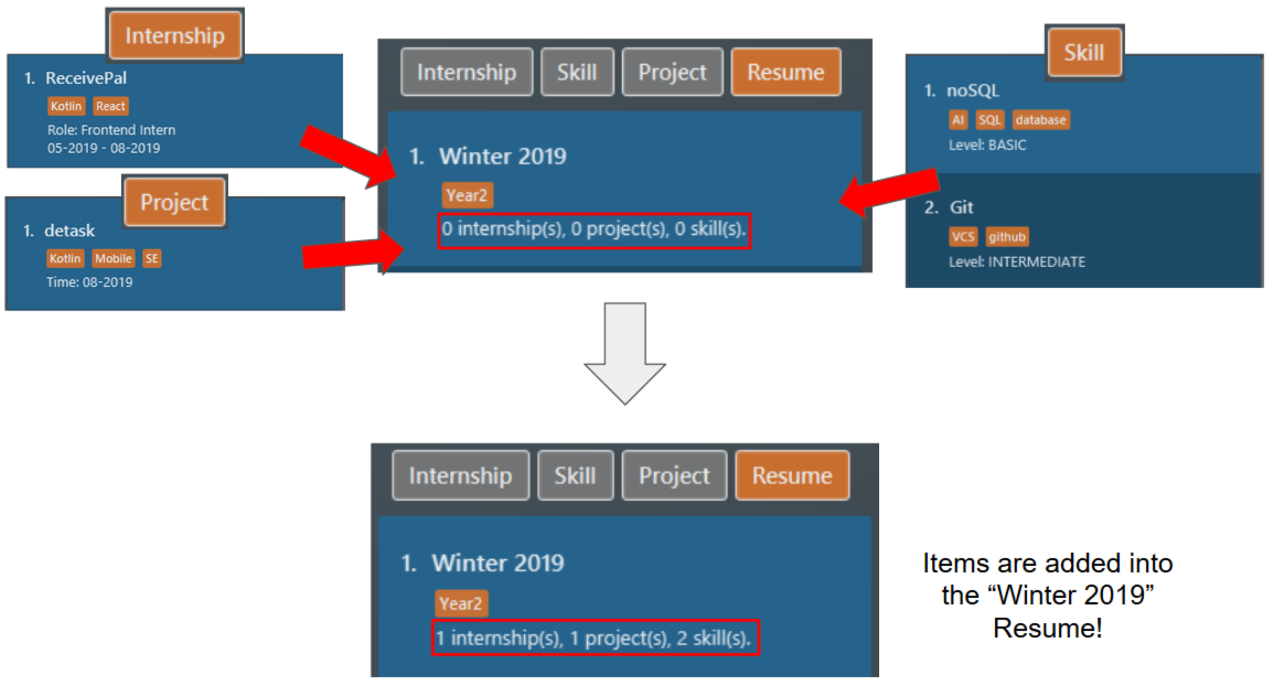


Figure 27. Resume content after adding new items

Example 2: Modifying the contents of the resume

Following from example 1, you realise that you actually wanted to use a different set of items for your "Winter 2019" `resume`. Here is how you can rectify this!

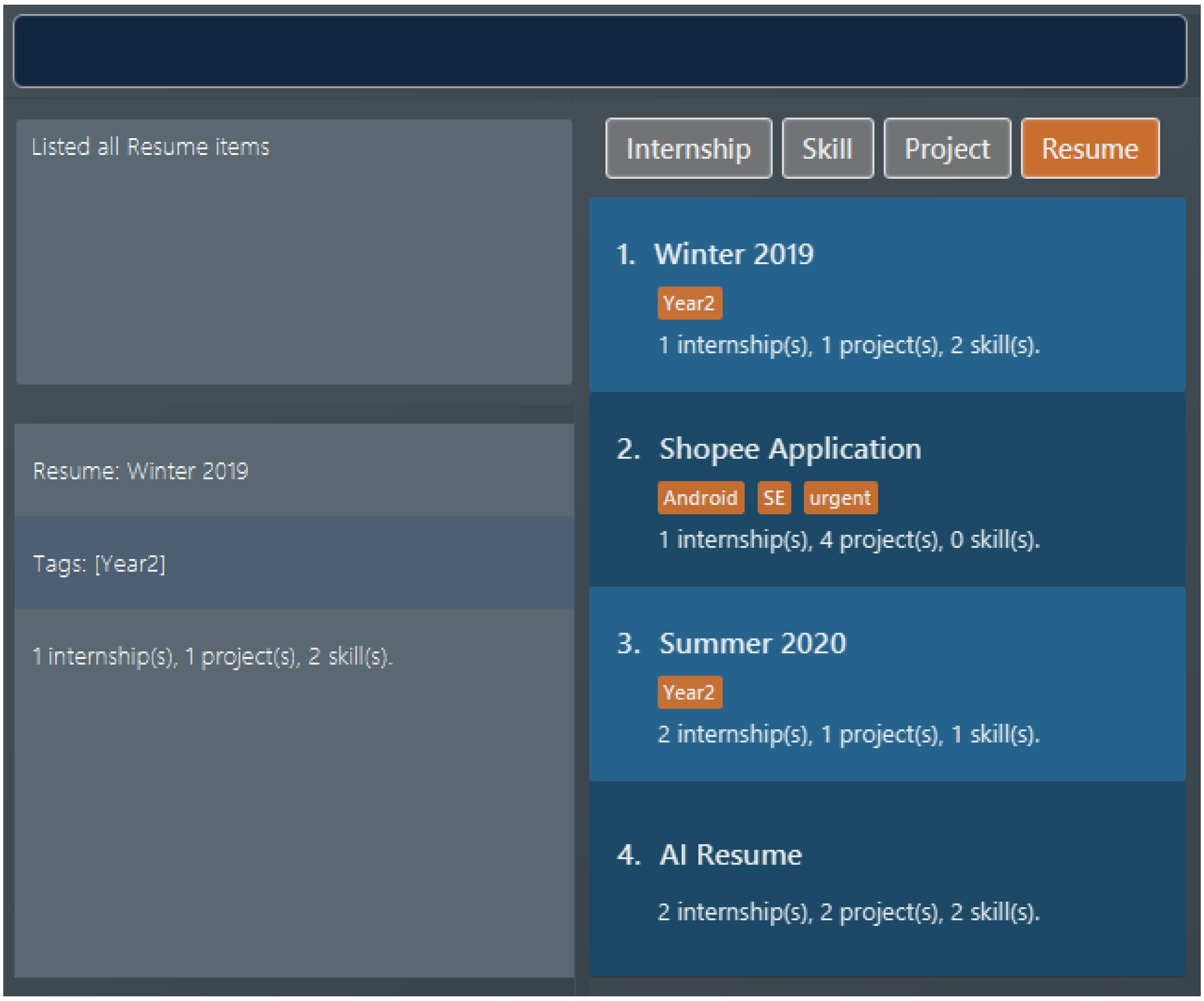
Before you try this example, please ensure you have added at least one `internship` and three `skill` items. You may verify for their presence by using the `list` command.

Try typing in these commands:

1. `list i/ res`
2. `redit 1 int/ 1 proj/ ski/ 3`

Outcome:

1. The first command lists out all `resume` items. Assuming you want to modify the contents of the first `resume` in the list box. Since we are following from example 1, the "Winter 2019" `resume` would already have some items in it.


Listed all Resume items

Resume: Winter 2019

Tags: [Year2]

1 internship(s), 1 project(s), 2 skill(s).

1. Winter 2019
   Year2
   1 internship(s), 1 project(s), 2 skill(s).
2. Shopee Application
   Android SE urgent
   1 internship(s), 4 project(s), 0 skill(s).
3. Summer 2020
   Year2
   2 internship(s), 1 project(s), 1 skill(s).
4. AI Resume
   2 internship(s), 2 project(s), 2 skill(s).

Figure 28. List of all resume items

2. Before the second command, the `resume` contains the same set of items as the one in example 1. The command modifies the `resume` at index 1, named "Winter 2019". The `resume` now still contains the `internship` item at index 1, no `project` items, and `skill` item at index 3. The following screenshot illustrates what happens after this command:

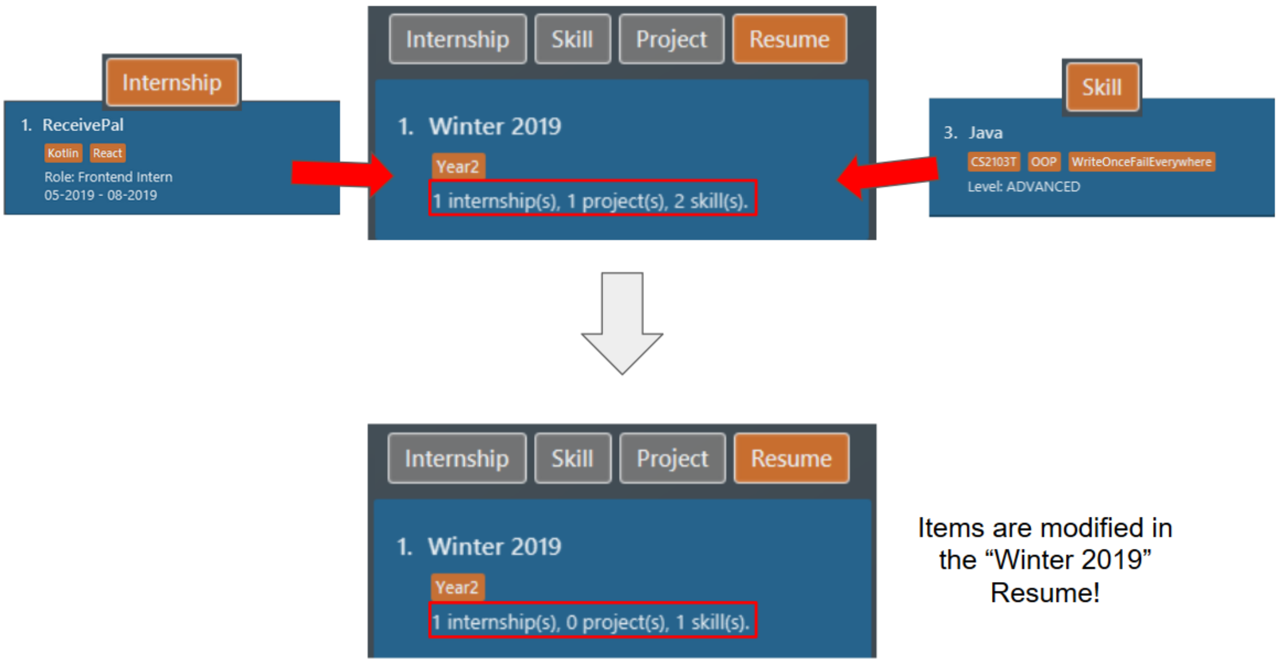


Figure 29. Resume content after modification

You can think of this command as using only the items that has been specified in the command. In this example, we no longer want the items we have previously added, and we would like the "Winter 2019" `resume` to contain only the first `internship` item and the third `skill` item.

Example 3: No modifications to `internship` and `skill` items

Following from example 2, you are now satisfied with the `internship` and `skill` items, but you would like to add a `project` item without changing the `internship` and `skill` items. Here is how you can do it!

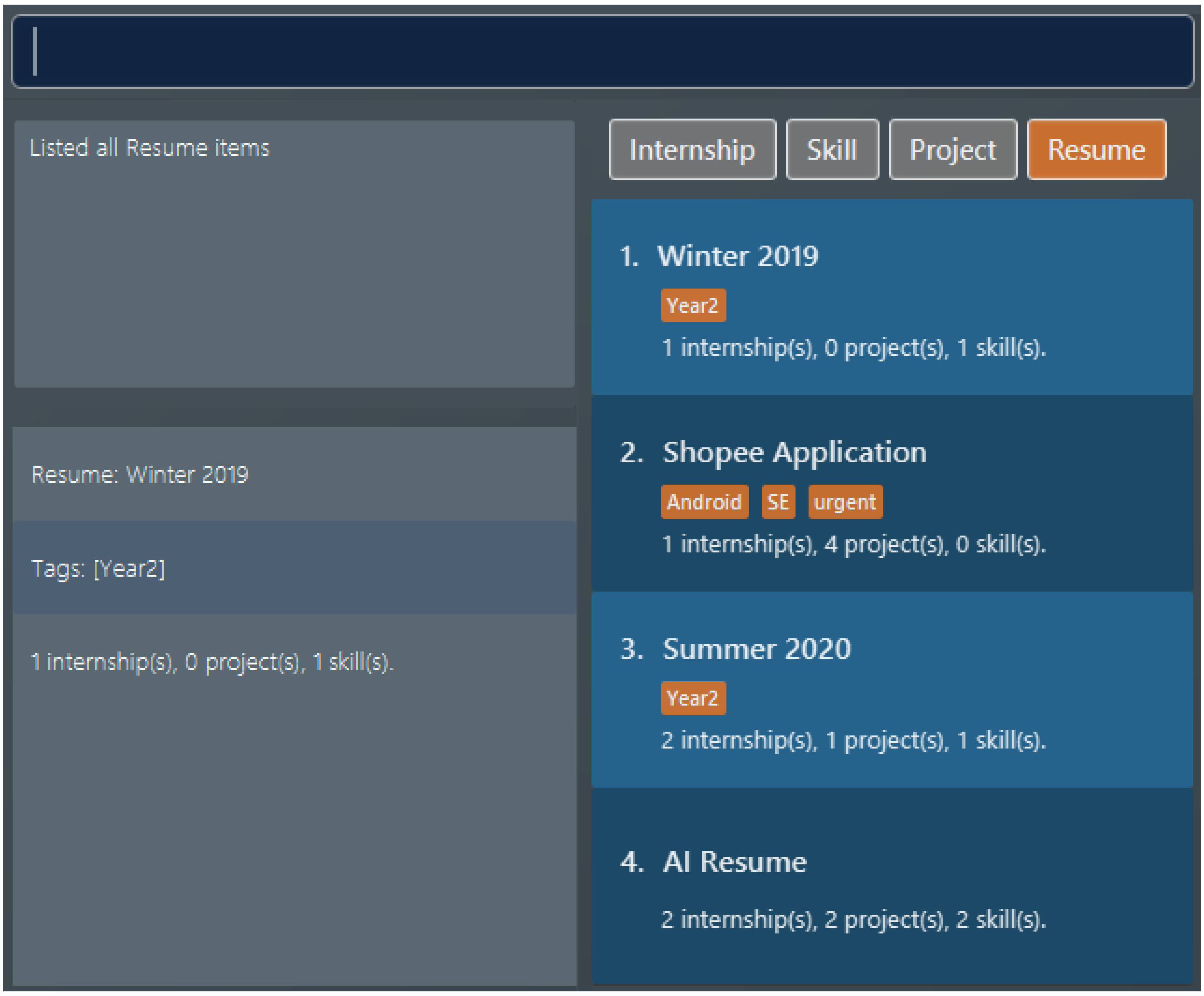
Before you try out this example, please ensure you have at least two `project` items.

Try typing in these commands:

1. `list i/ res`
2. `redit 1 proj/ 2`

Outcome:

1. The first command lists out all `resume` items. Assuming you want to modify the contents of the first `resume` in the list box. Since we are following from example 2, the "Winter 2019" `resume` would already have some items in it.



Listed all Resume items

Resume: Winter 2019

Tags: [Year2]

1 internship(s), 0 project(s), 1 skill(s).

1. Winter 2019
   Year2
   1 internship(s), 0 project(s), 1 skill(s).
2. Shopee Application
   Android SE urgent
   1 internship(s), 4 project(s), 0 skill(s).
3. Summer 2020
   Year2
   2 internship(s), 1 project(s), 1 skill(s).
4. AI Resume
   2 internship(s), 2 project(s), 2 skill(s).

Figure 30. List of all resume items

2. Before the second command, the `resume` contains the same set of items as the one in example 2. The command modifies the `resume` at index 1, named "Winter 2019". The `resume` will now contain the the `project` item at index 2. The `internship` and `skill` items remain unchanged. The following screenshot illustrates what happens after this command:

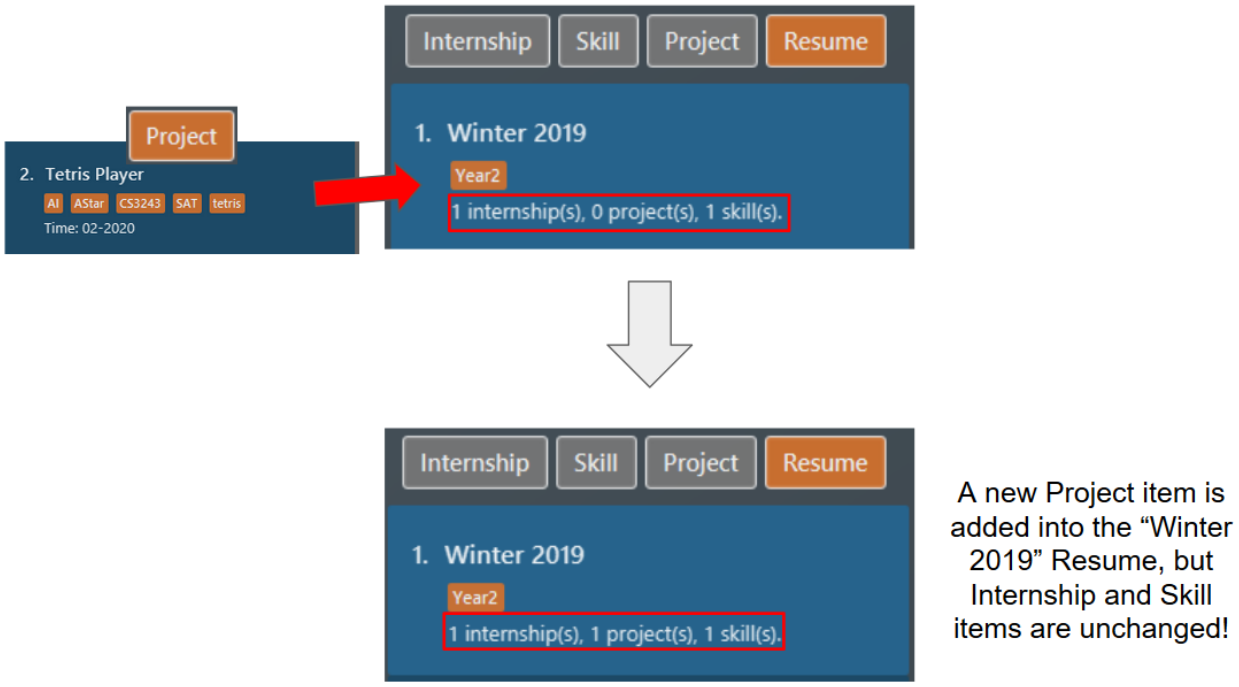

Project
2. Tetris Player
AI AStar CS3243 SAT tetris
Time: 02-2020
Internship Skill Project Resume
1. Winter 2019
Year2
1 internship(s), 0 project(s), 1 skill(s).
Internship Skill Project Resume
1. Winter 2019
Year2
1 internship(s), 1 project(s), 1 skill(s).
A new Project item is added into the "Winter 2019" Resume, but Internship and Skill items are unchanged!

Figure 31. Resume content after modification

So by leaving out one of the prefixes (`TYPE/ ITEM_INDEX…`) from the command, you can keep some of the items in your `resume`!

{ end of `redit` section written by: Christian James Welly }

{ start of `tagpull` section written by: Christian James Welly }

4.11. Pulling tagged items into a resume: `tagpull`

Pulls all the `internship`, `project` and `skill` items with the specified tag(s) into a `resume`.

Format: `tagpull RESUME_INDEX [#/ TAG]…`

This command does not remove items from the `resume` and will add items on top of the existing items.

Example 1: Pulling items using one tag

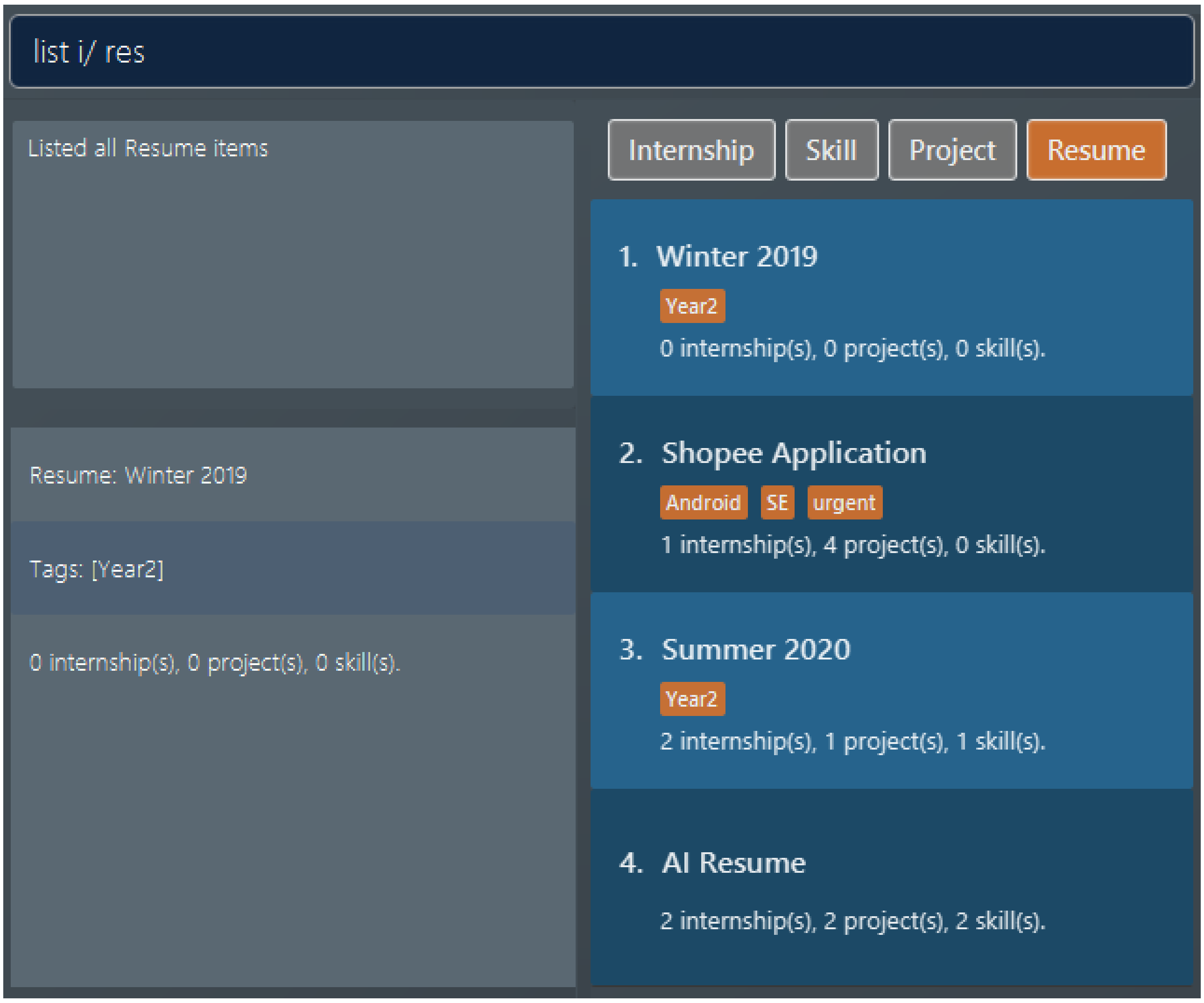
Before you try this example, please ensure you have added some items with the tag "SE". You may verify for their presence by using the `list` command.

Try typing in these commands:

1. `list i/ res`
2. `tagpull 1 #/ SE`

Outcome:

1. The first command lists out all `resumes`. Assuming you want to modify the contents of the first `resume` in the list box.



```
list i/ res
```

Listed all Resume items

Resume: Winter 2019

Tags: [Year2]

0 internship(s), 0 project(s), 0 skill(s).

1. Winter 2019
Year2
0 internship(s), 0 project(s), 0 skill(s).

2. Shopee Application
Android SE urgent
1 internship(s), 4 project(s), 0 skill(s).

3. Summer 2020
Year2
2 internship(s), 1 project(s), 1 skill(s).

4. AI Resume
2 internship(s), 2 project(s), 2 skill(s).

Figure 32. List of all resume items

2. The second command pulls all items which has "SE" as the tag into the `resume` at index 1, named "Winter 2019". The following screenshot illustrates the outcome of the command if we have 1 `internship` item and 2 `project` items tagged with "SE".

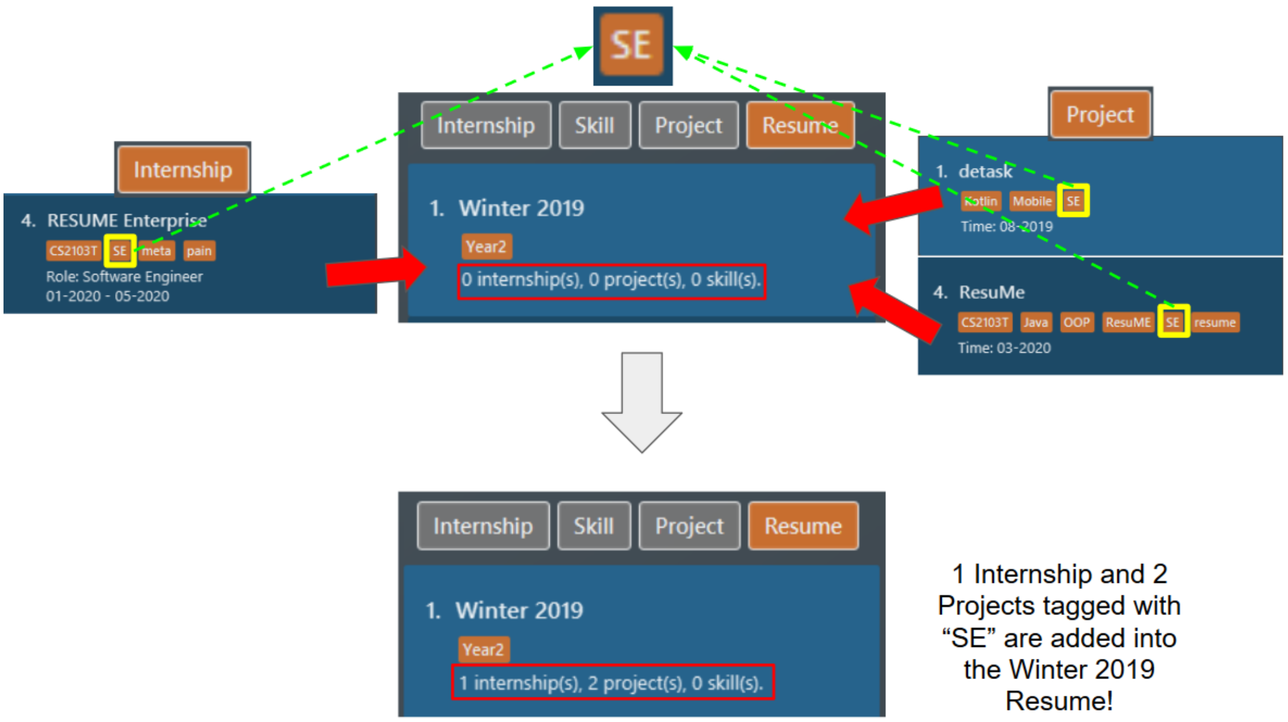


Figure 33. Resume content after `tagpull`

Example 2: Pulling items using multiple tags

Now, maybe you have categorised your items by using multiple tags. So, you think just pulling items using one tags is not sufficient. Our `tagpull` supports that too!

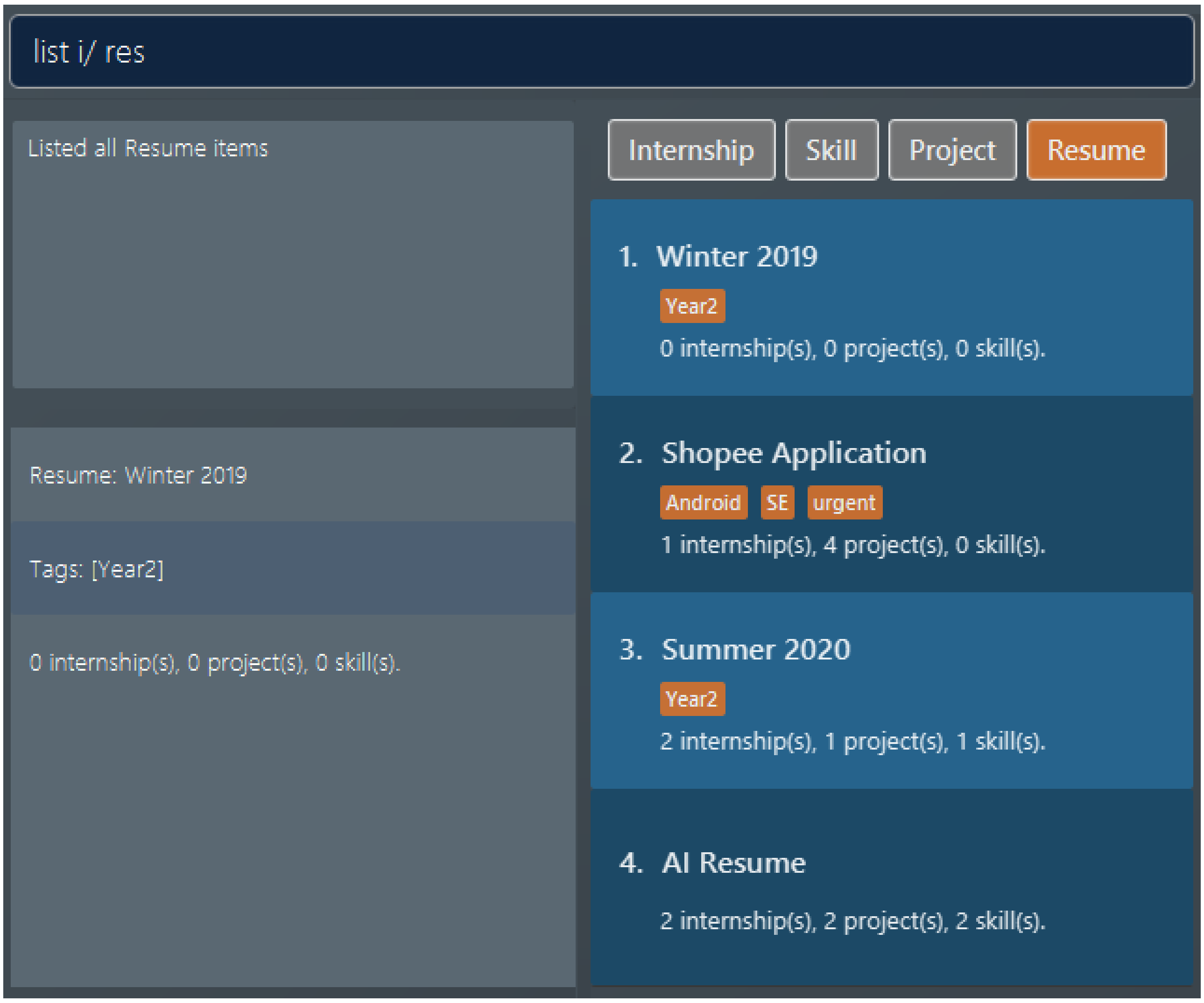
Before you try this example, please ensure you have added some items with the tag "SE" and/or "github". You may verify for their presence by using the `list` command.

Try typing in these commands:

1. `list i/ res`
2. `tagpull 1 #/ SE #/ github`

Outcome:

1. The first command lists out all `resumes`. Assuming you want to modify the contents of the first `resume` in the list box.



```
list i/ res
```

Listed all Resume items

Resume: Winter 2019

Tags: [Year2]

0 internship(s), 0 project(s), 0 skill(s).

1. Winter 2019
   Year2
   0 internship(s), 0 project(s), 0 skill(s).
2. Shopee Application
   Android SE urgent
   1 internship(s), 4 project(s), 0 skill(s).
3. Summer 2020
   Year2
   2 internship(s), 1 project(s), 1 skill(s).
4. AI Resume
   2 internship(s), 2 project(s), 2 skill(s).

Figure 34. List of all resume items

2. The second command pulls all items which has "SE" *or* "github" as the tag into the `resume` at index 1, named "Winter 2019". The following screenshot illustrates the outcome of the command if we have 1 `internship` item and 2 `project` items tagged with "SE", and 1 `skill` item tagged with "github".

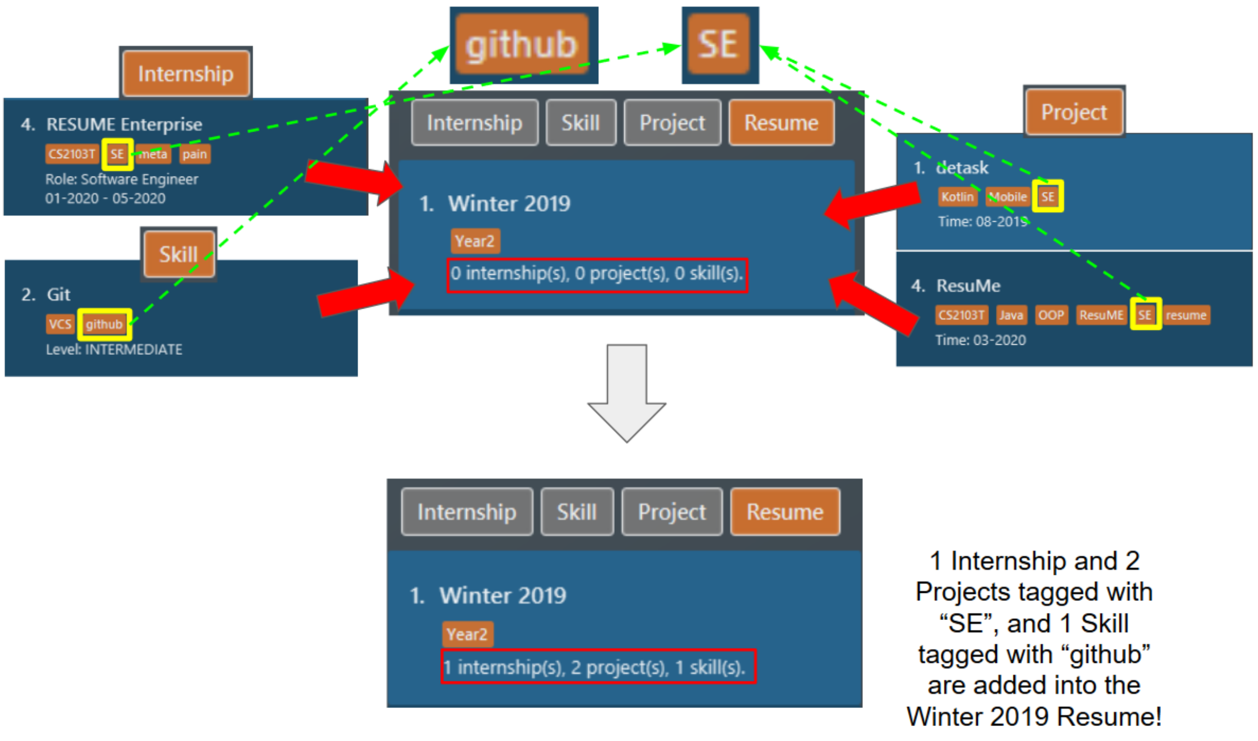

github
SE
Internship
4. RESUME Enterprise
CS2103T SE meta pain
Role: Software Engineer
01-2020 - 05-2020
Skill
2. Git
VCS github
Level: INTERMEDIATE
Internship Skill Project Resume
1. Winter 2019
Year2
0 internship(s), 0 project(s), 0 skill(s).
Project
1. detask
Kotlin Mobile SE
Time: 08-2019
4. ResuMe
CS2103T Java OOP ResuME SE resume
Time: 03-2020
Internship Skill Project Resume
1. Winter 2019
Year2
1 internship(s), 2 project(s), 1 skill(s).
1 Internship and 2 Projects tagged with "SE", and 1 Skill tagged with "github" are added into the Winter 2019 Resume!

Figure 35. Resume content after tagpull

{ end of `tagpull` section written by: Christian James Welly }

{ start of `undo` section written by: Nguyen Minh Hoang}

4.12. Undoing a command: `undo`

Undoes the previous command and restores the state of the application to before that command is performed.

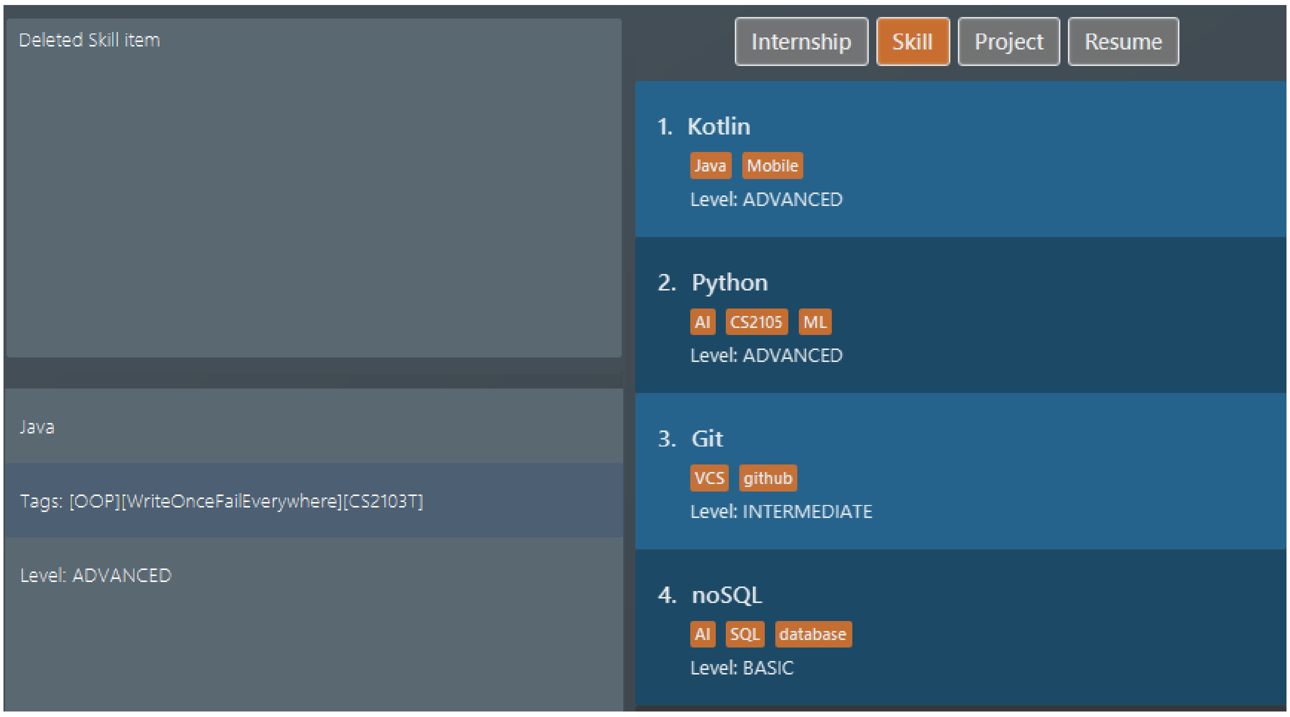
Format: `undo`

- An `undo` is done per command and not per change. If you make multiple changes to your user profile, for instance, using a single `me` command, then `undo` will revert all the changes.
- Commands that can be undone are: `add`, `delete`, `edit`, `sort`, `redit`, `tagpull`, `done`, `clear` and `me`.
- Commands that make no change to the application state, namely `list`, `view`, `find`, `rpreview` and `rgen`, cannot be undone.
- Successive `undo` commands will bring the application further back, until there is no more change to `undo`.
- You cannot `undo` if there is no previous state to return to.

Example:

1. Suppose we hate Java and therefore want to delete it from the list of `skill` items using a `delete`

`1 i/ ski` command. We also do a `list i/ proj` to switch to viewing projects because that is how staging for an example works.



Deleted Skill item

Java

Tags: [OOP][WriteOnceFailEverywhere][CS2103T]

Level: ADVANCED

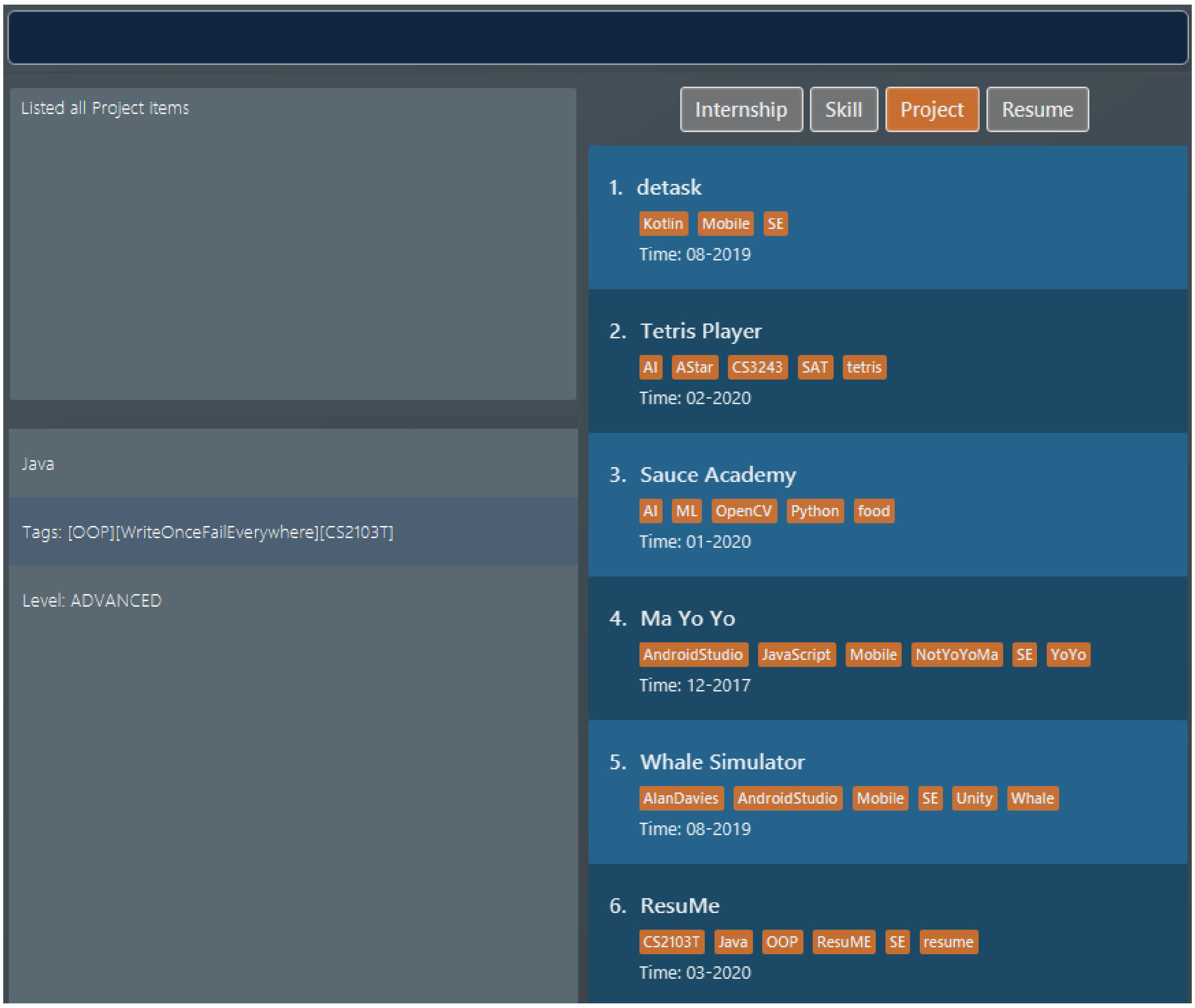
1. Kotlin
Java Mobile
Level: ADVANCED

2. Python
AI CS2105 ML
Level: ADVANCED

3. Git
VCS github
Level: INTERMEDIATE

4. noSQL
AI SQL database
Level: BASIC

Figure 36. List of all skill items after deleting "Java"



Listed all Project items

Java

Tags: [OOP][WriteOnceFailEverywhere][CS2103T]

Level: ADVANCED

1. detask
Kotlin Mobile SE
Time: 08-2019

2. Tetris Player
AI AStar CS3243 SAT tetris
Time: 02-2020

3. Sauce Academy
AI ML OpenCV Python food
Time: 01-2020

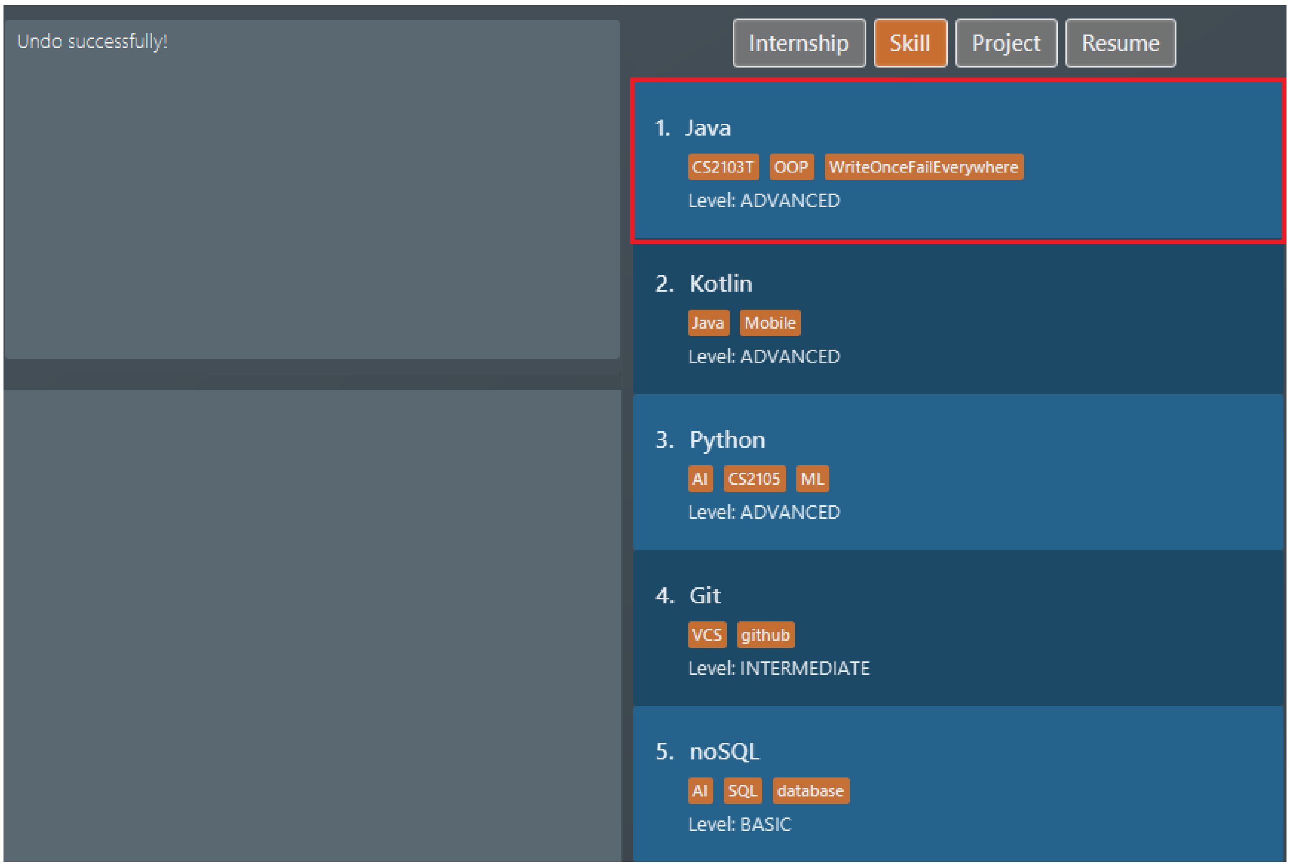
4. Ma Yo Yo
AndroidStudio JavaScript Mobile NotYoYoMa SE YoYo
Time: 12-2017

5. Whale Simulator
AlanDavies AndroidStudio Mobile SE Unity Whale
Time: 08-2019

6. ResuMe
CS2103T Java OOP ResuME SE resume
Time: 03-2020

Figure 37. List of all project items

2. But Stockholm's syndrome kicks in and we decide that Java has its merits, so we revert our action with `undo`. Voila, Java is back, and our list box now displays `skill` items.


Undo successfully!

1. Java
CS2103T OOP WriteOnceFailEverywhere
Level: ADVANCED

2. Kotlin
Java Mobile
Level: ADVANCED

3. Python
AI CS2105 ML
Level: ADVANCED

4. Git
VCS github
Level: INTERMEDIATE

5. noSQL
AI SQL database
Level: BASIC

Figure 38. List of all skill items after undo

{ end of `undo` section written by: Nguyen Minh Hoang}

{ start of `redo` section written by: Nguyen Minh Hoang}

4.13. Redoing a command: `redo`

Redoes the most recent undone command and brings the application state to after the (re)execution of that command.

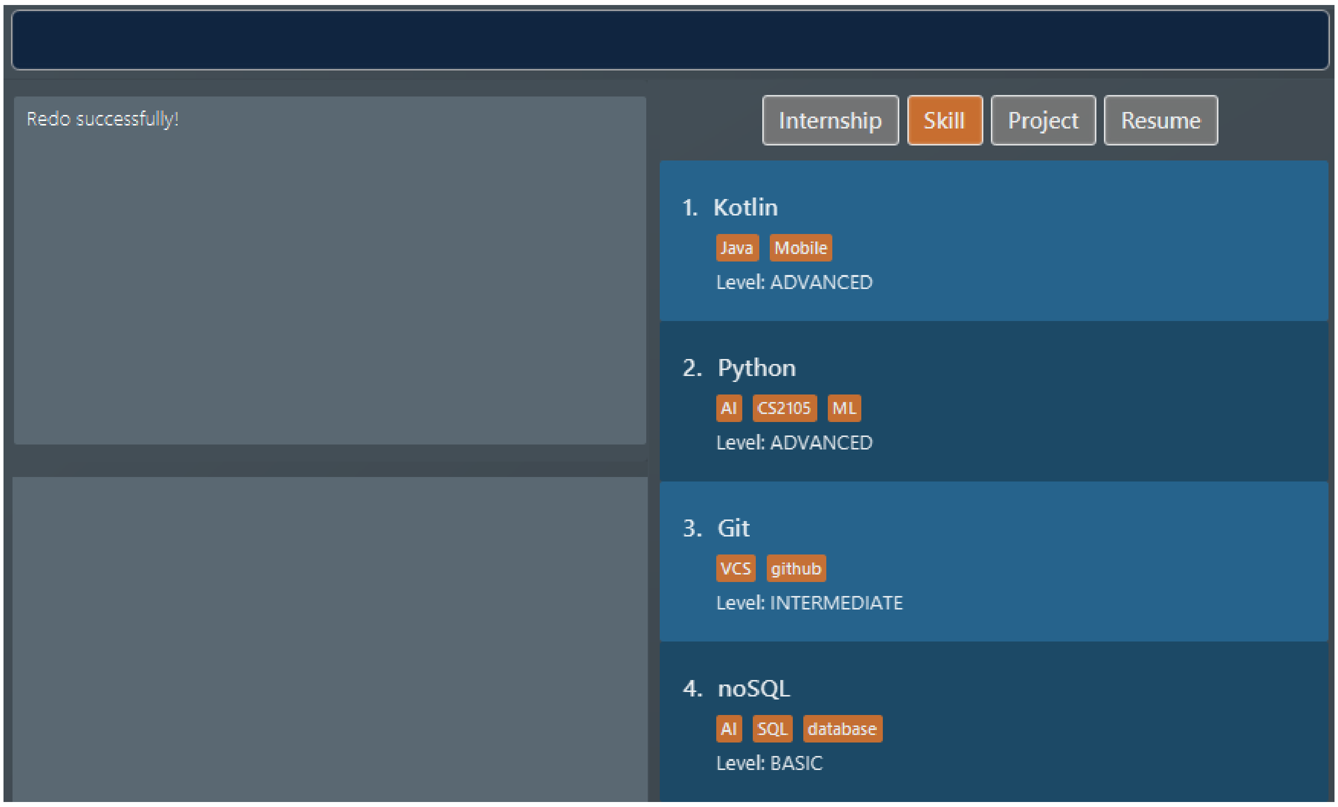
Format: `redo`

- A `redo` is done per command and not per change. If you undo a command that can make multiple changes like `redit`, then redoing it will resurrect all the changes. Spooky!
- If a new `add`, `delete`, `edit`, `sort`, `redit`, `tagpull`, `done`, `clear`, or `me` command is performed after an `undo`, then all states currently available for `redo` will be erased and you will not be able to go to these states anymore.
- You cannot `redo` if there is no forward state to go to.

Example:

After the Java deletion incidence in the previous example section (for `undo`), we decide that the merits of Java do not warrant the pain it wreaks so we want to `redo` our original decision to delete

Java. Once again, Java is removed from our life.



Redo successfully!

1. Kotlin
Java Mobile
Level: ADVANCED

2. Python
AI CS2105 ML
Level: ADVANCED

3. Git
VCS github
Level: INTERMEDIATE

4. noSQL
AI SQL database
Level: BASIC

Figure 39. List of all skill items after redo

{ end of `redo` section written by: Nguyen Minh Hoang}

{ start of `rpreview` section written by: Nguyen Chi Hai }

4.14. Previewing a resume: `rpreview`

Previews a `resume` in text format in a pop-up window.

Format: `rpreview RESUME_INDEX`

- `RESUME_INDEX` is the index of the `resume` seen when `list i/ res` is called.

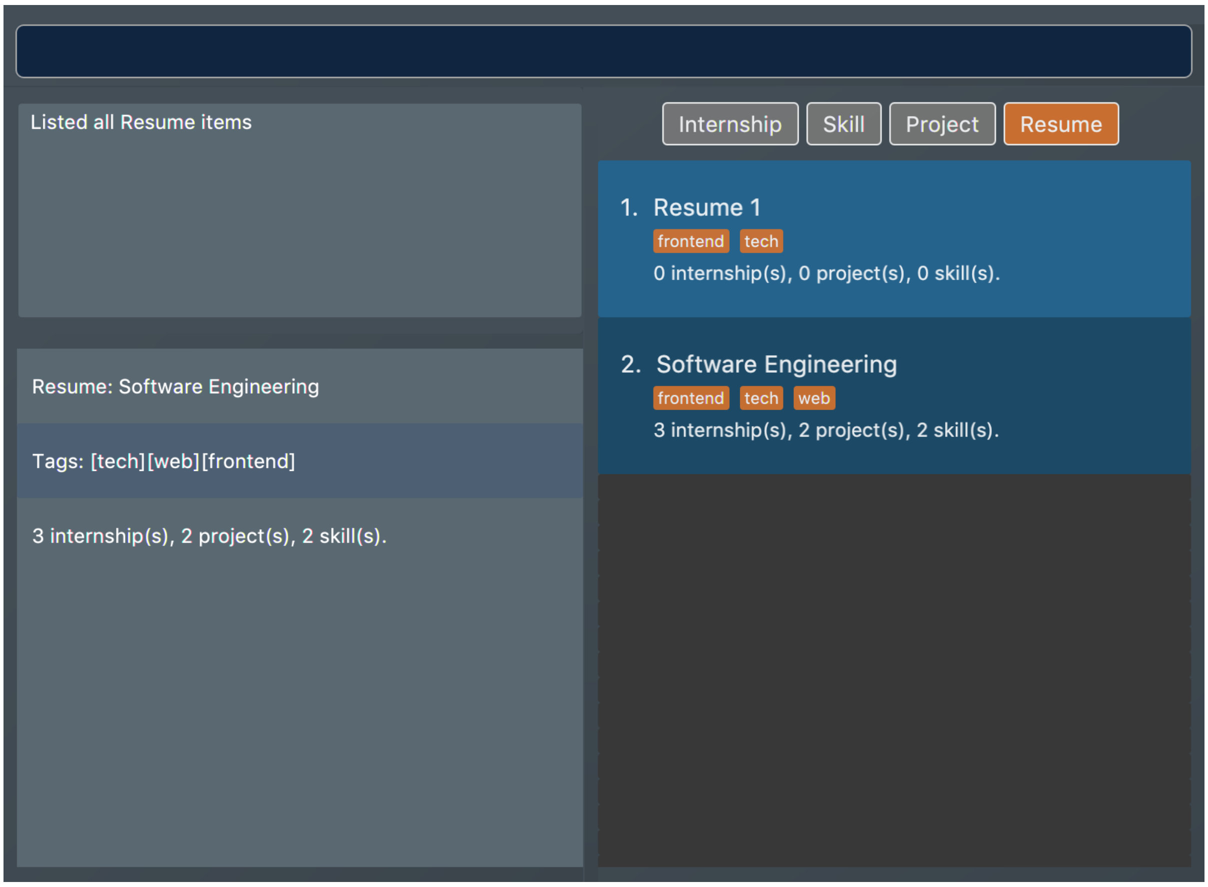
- `description` of `internship` and `project` is automatically separated into bullet points when the program detects full sentences marked by a full stop.

Example: Let's try out the following commands!

1. `list i/ res`
2. `rpreview 2`

Outcome:

1. The first command lists out all `resume` items. Assuming that you want to preview the second `resume` in the list box.


Listed all Resume items

Resume: Software Engineering

Tags: [tech][web][frontend]

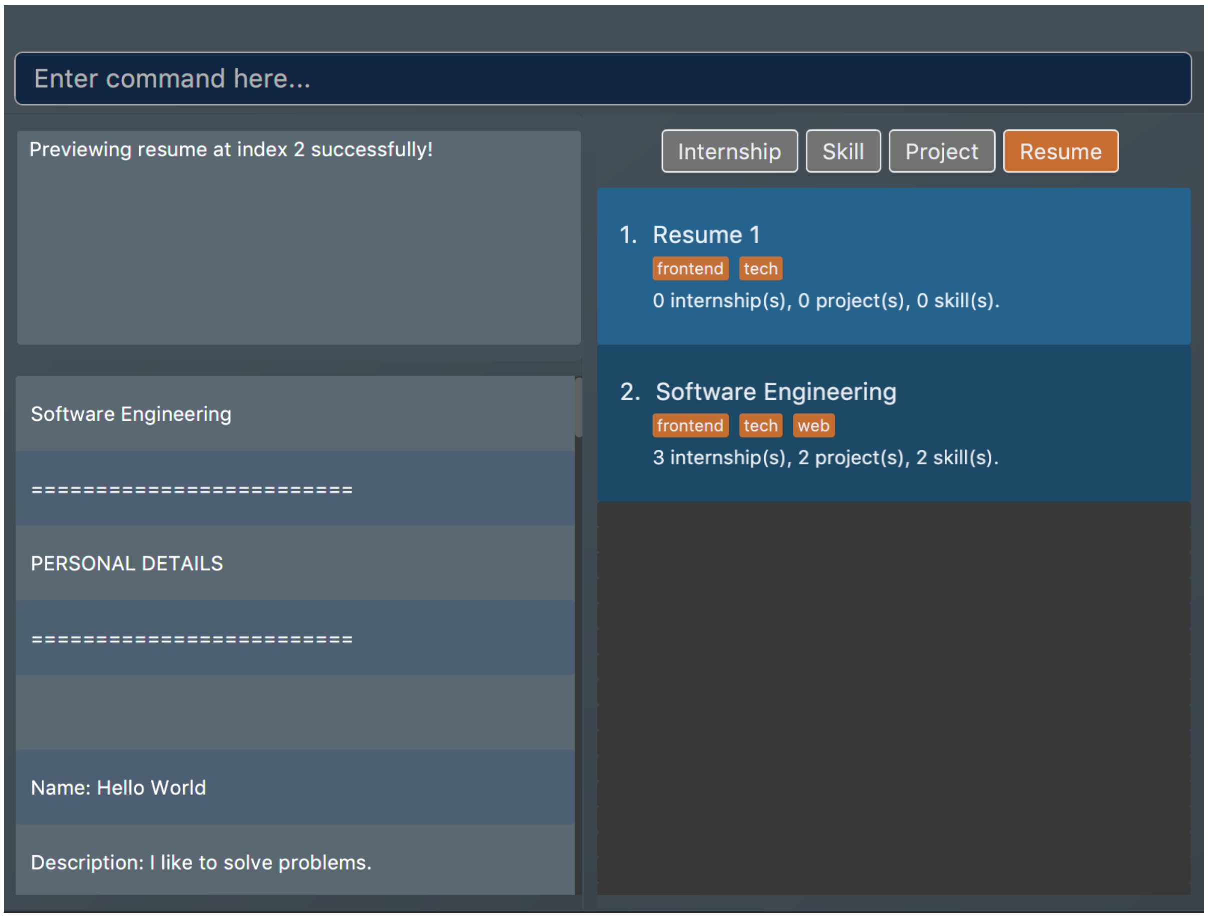
3 internship(s), 2 project(s), 2 skill(s).

1. Resume 1
frontend tech
0 internship(s), 0 project(s), 0 skill(s).

2. Software Engineering
frontend tech web
3 internship(s), 2 project(s), 2 skill(s).

Figure 40. List of all resume items

2. The second command will preview the `resume` at index 2 named "Software Engineering". A pop-up window will be opened, featuring a text-based view of the content of "Software Engineering". The screenshots of the results are as shown below:



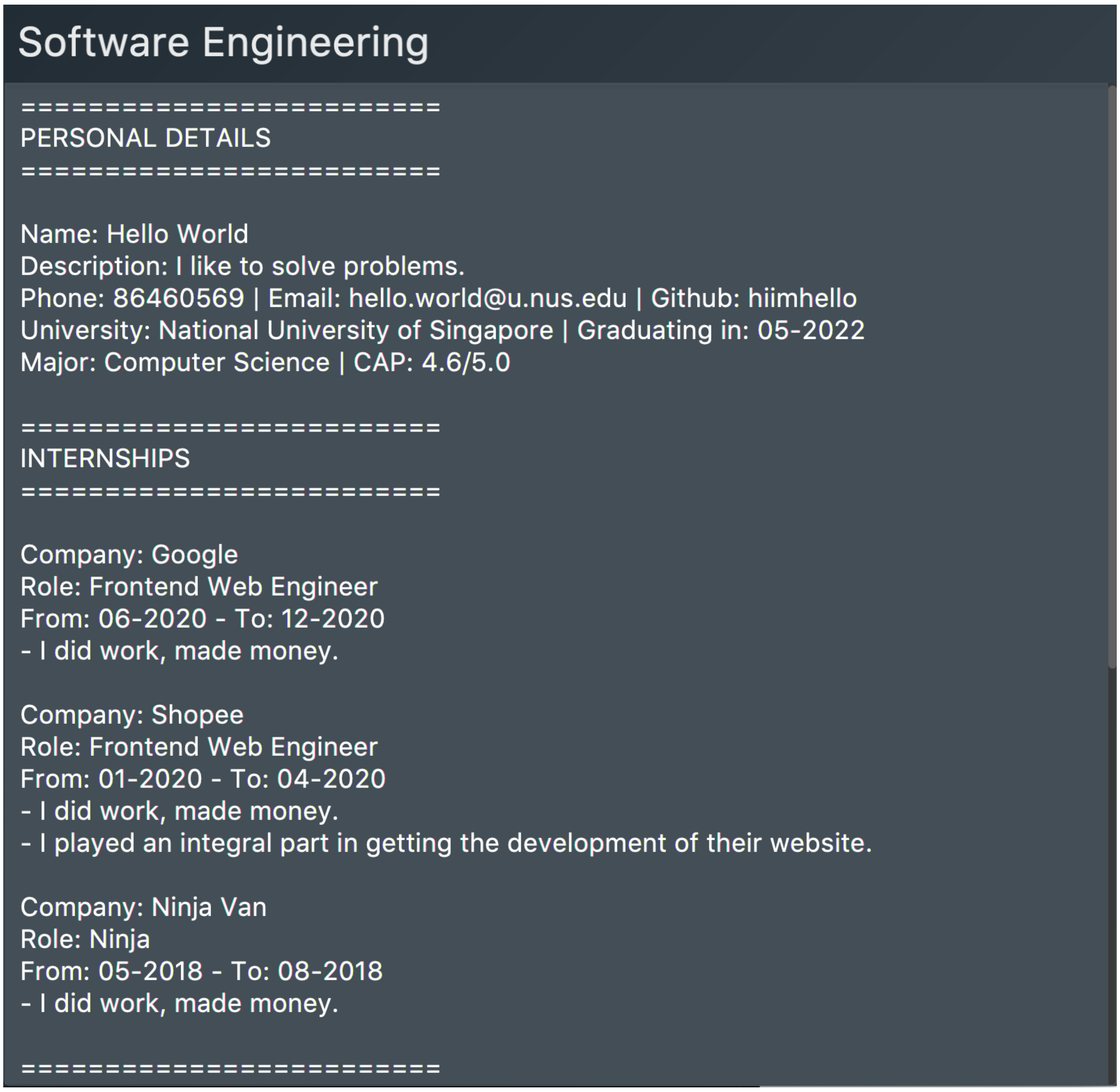
Previewing resume at index 2 successfully!

```
Software Engineering
========================
PERSONAL DETAILS
========================

Name: Hello World
Description: I like to solve problems.
```

1. Resume 1
   frontend tech
   0 internship(s), 0 project(s), 0 skill(s).
2. Software Engineering
   frontend tech web
   3 internship(s), 2 project(s), 2 skill(s).

Figure 41. Application view after rpreview is executed


Software Engineering

```
=========================
PERSONAL DETAILS
=========================

Name: Hello World
Description: I like to solve problems.
Phone: 86460569 | Email: hello.world@u.nus.edu | Github: hiimhello
University: National University of Singapore | Graduating in: 05-2022
Major: Computer Science | CAP: 4.6/5.0

=========================
INTERNSHIPS
=========================

Company: Google
Role: Frontend Web Engineer
From: 06-2020 - To: 12-2020
- I did work, made money.

Company: Shopee
Role: Frontend Web Engineer
From: 01-2020 - To: 04-2020
- I did work, made money.
- I played an integral part in getting the development of their website.

Company: Ninja Van
Role: Ninja
From: 05-2018 - To: 08-2018
- I did work, made money.

=========================
```

Figure 42. Preview pop-up window

{ end of `rpreview` section written by: Nguyen Chi Hai }

{ start of `rgen` section written by: Pham Thuy Duong }

4.15. Generating a resume: `rgen`

Generates a `.pdf` file from an existing `resume` stored inside the application at the specified index.

Format: `rgen RESUME_INDEX [n/ FILE_NAME]`

- A valid `RESUME_INDEX` is a positive integer that identifies an existing `resume`.
- The exported `.pdf` file will be located in the same folder as the application `.jar` file.

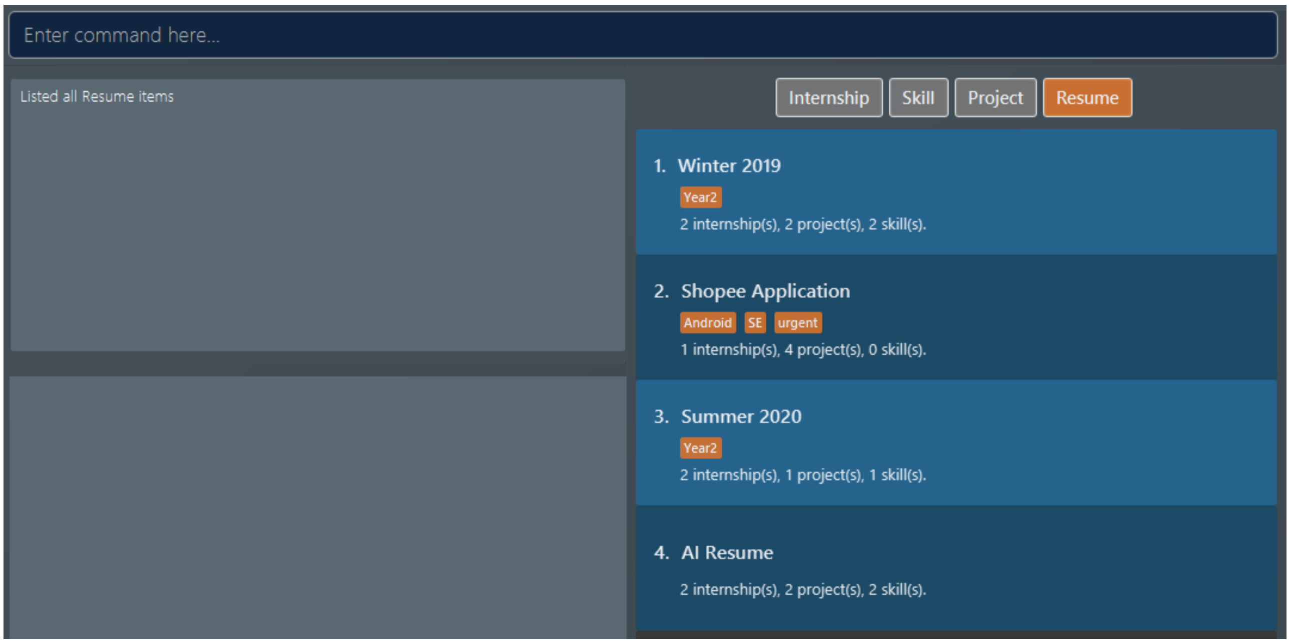
It is optional to specify a `FILE_NAME` for the .pdf file. If no name is specified, the filename will be set, by default, to the name of the generated `resume`.

Example: Let's try out the following commands!

1. `list i/ res`
2. `rgen 1 n/ My Resume`

Outcome:

1. The first command lists out all `resume` items. Assuming that you want to generate the first `resume` in the list box.



Listed all Resume items

1. Winter 2019
Year2
2 internship(s), 2 project(s), 2 skill(s).

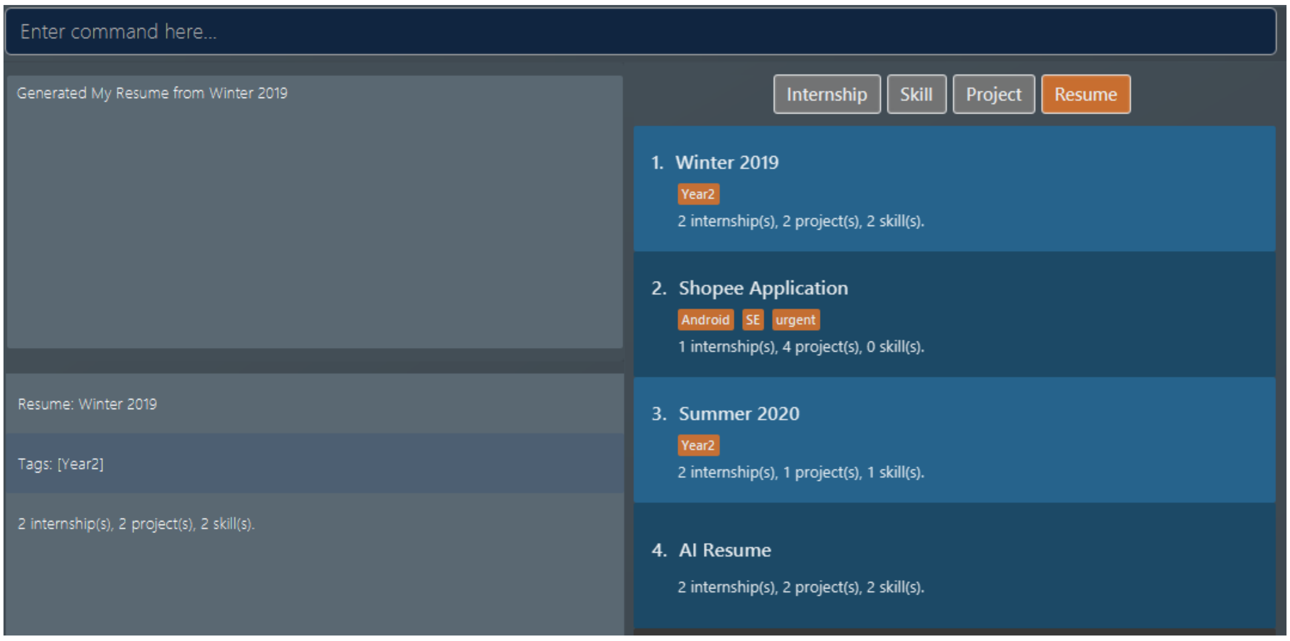
2. Shopee Application
Android SE urgent
1 internship(s), 4 project(s), 0 skill(s).

3. Summer 2020
Year2
2 internship(s), 1 project(s), 1 skill(s).

4. AI Resume
2 internship(s), 2 project(s), 2 skill(s).

Figure 43. List of all resume items

2. The second command generates a "My Resume.pdf" file from the specified resume. The screenshots of the application view and of the generated file are as below:



Generated My Resume from Winter 2019

Resume: Winter 2019

Tags: [Year2]

2 internship(s), 2 project(s), 2 skill(s).

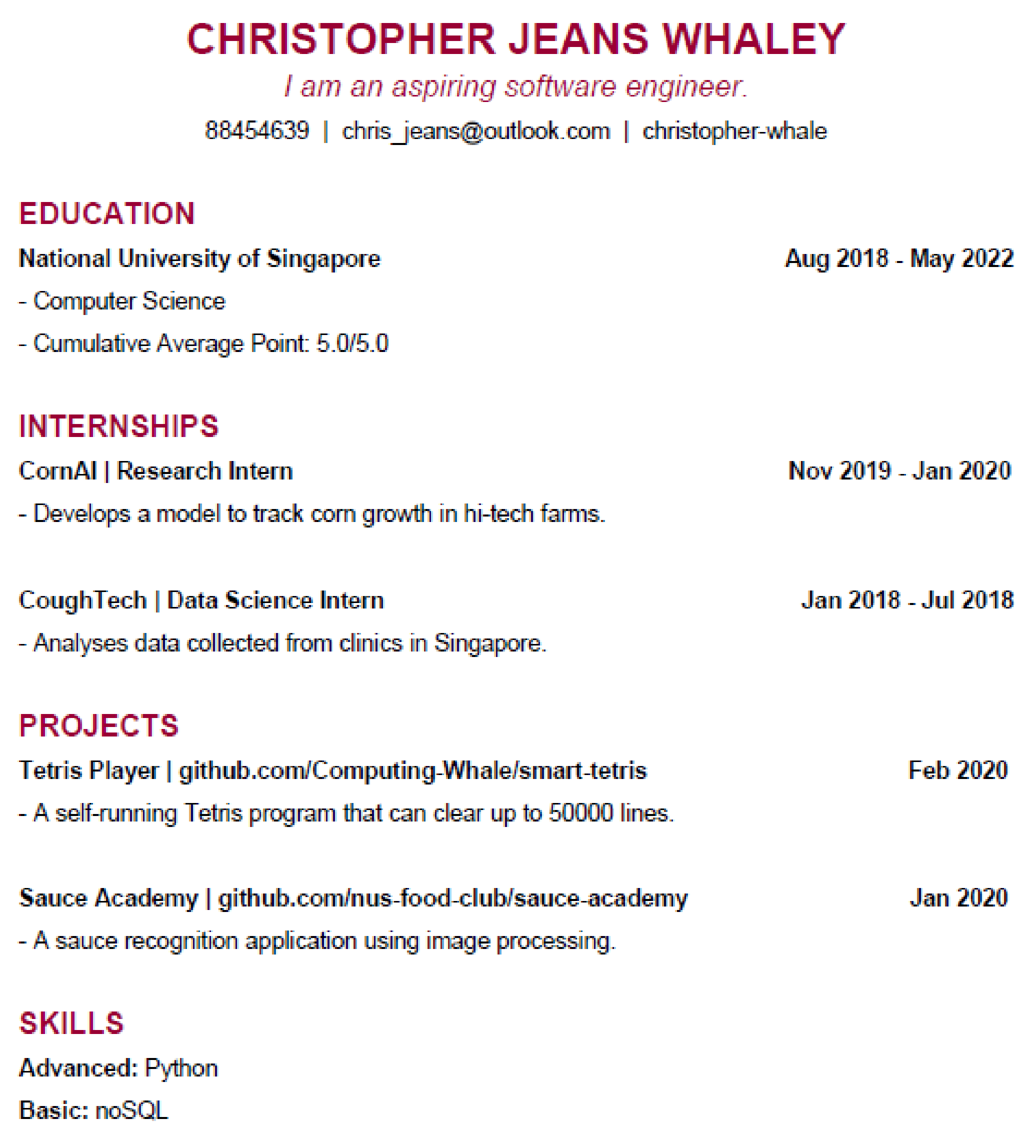
1. Winter 2019
Year2
2 internship(s), 2 project(s), 2 skill(s).

2. Shopee Application
Android SE urgent
1 internship(s), 4 project(s), 0 skill(s).

3. Summer 2020
Year2
2 internship(s), 1 project(s), 1 skill(s).

4. AI Resume
2 internship(s), 2 project(s), 2 skill(s).

Figure 44. Application view after rgen is executed


CHRISTOPHER JEANS WHALEY

I am an aspiring software engineer.

88454639 | chris_jeans@outlook.com | christopher-whale

EDUCATION

National University of Singapore **Aug 2018 - May 2022**

- Computer Science
- Cumulative Average Point: 5.0/5.0

INTERNSHIPS

CornAI | **Research Intern** **Nov 2019 - Jan 2020**

- Develops a model to track corn growth in hi-tech farms.

CoughTech | **Data Science Intern** **Jan 2018 - Jul 2018**

- Analyses data collected from clinics in Singapore.

PROJECTS

Tetris Player | github.com/Computing-Whale/smart-tetris **Feb 2020**

- A self-running Tetris program that can clear up to 50000 lines.

Sauce Academy | github.com/nus-food-club/sauce-academy **Jan 2020**

- A sauce recognition application using image processing.

SKILLS

Advanced: Python

Basic: noSQL

Figure 45. Generated .pdf file

{ end of `rgen` section written by: Pham Thuy Duong }

{ start of `done` section written by: Nham Quoc Hung }

4.16. Mark a note as done: `done`

Mark a `Note` at a specific index from the current note list as `done`.

Format: `done INDEX`

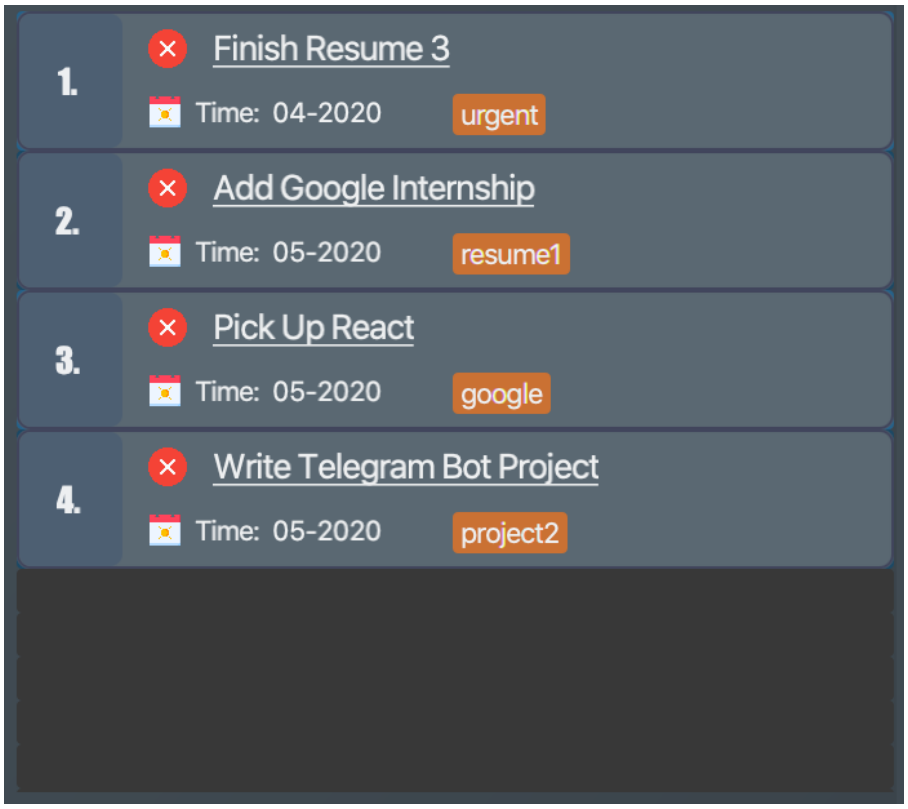
- A valid `INDEX` is a positive integer that identifies an existing `note`.

Example: Let’s try out the following commands!

1. `list i/ note`
2. `done 1`

Outcome:

1. The first command lists out all `notes`. Assuming that you want to mark the first `note` in the list as `done`.



1. ✗ Finish Resume 3
Time: 04-2020 urgent

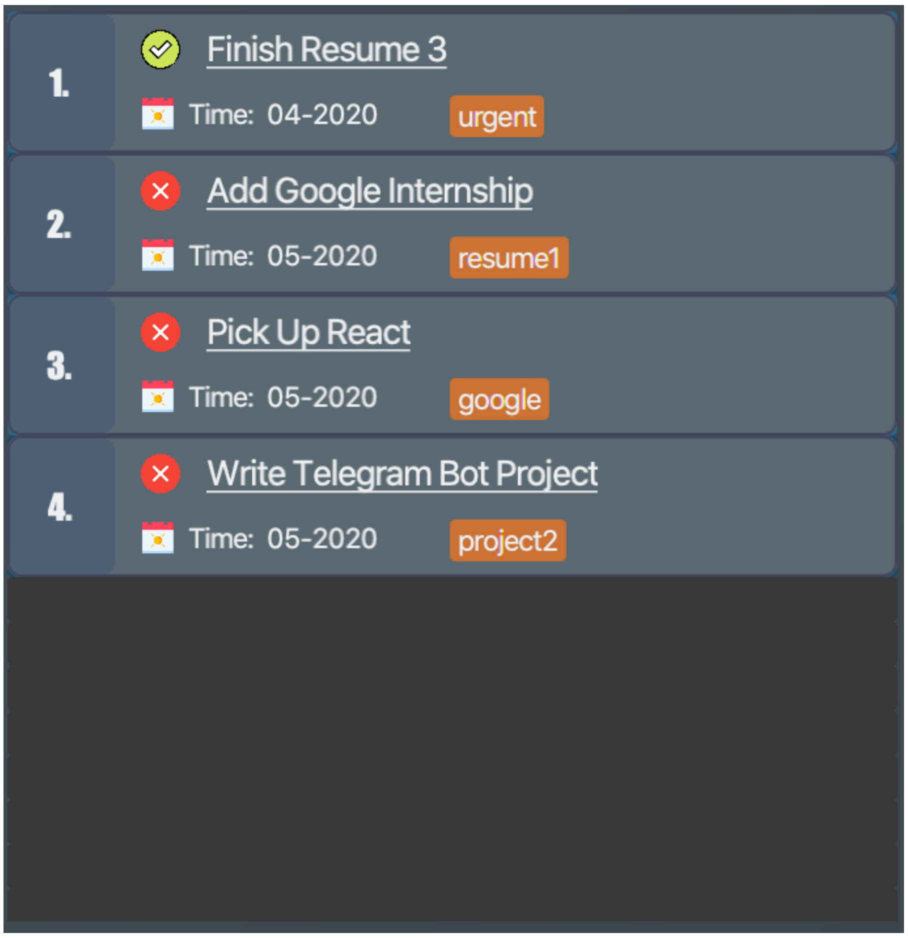
2. ✗ Add Google Internship
Time: 05-2020 resume1

3. ✗ Pick Up React
Time: 05-2020 google

4. ✗ Write Telegram Bot Project
Time: 05-2020 project2

Figure 46. List of all note items

2. The second command marks this `note` as done, by updating the tick box.



1.
✓ Finish Resume 3
Time: 04-2020 urgent

2.
✕ Add Google Internship
Time: 05-2020 resume1

3.
✕ Pick Up React
Time: 05-2020 google

4.
✕ Write Telegram Bot Project
Time: 05-2020 project2

Figure 47. List of all note items after `done` *command is executed*

{ end of `done` section written by: Nham Quoc Hung }

4.17. Clearing all data: `clear`

Clears all data from **ResuMe**. Empties all data in the resume book, user information is replaced with default user information.

Format: `clear`

This command can be undone.

{ start of `exit` section written by: Nham Quoc Hung }

4.18. Exiting the program : `exit`

Exits from **ResuMe**.

Format: `exit`

{ end of `exit` section written by: Nham Quoc Hung }

{ start of `data-encryption` section written by: Nguyen Minh Hoang}

4.19. Saving the data

ResuMe data is saved in the hard disk automatically after any command that changes the data. There is no need to save manually.

4.20. Encrypting data files `[coming in v2.0]`

We are currently working on a login feature that will let you set password for ResuMe. You will also be able to have your locally stored data encrypted, so that no one (even us) can gain access to your precious information.

We value your privacy and we want you to feel safe. Stay tuned!

{ end of `data-encryption` section written by: Nguyen Minh Hoang}

5. FAQ

> **Q:** What is the difference between `redit` and `edit i/ res`?
>
> **A:** `redit` is used when you want to edit the content of a `resume`, in other words, add or remove certain items from that `resume`. Meanwhile, `edit i/ res` is used when you want to edit attributes of the `resume`, like its `name` or `tags`.

> **Q:** How do I transfer my data to another Computer?
>
> **A:** Install the app in the other Computer and overwrite the empty data file created with the data file of your **ResuMe** folder in the current Computer.

{ start of `summary` section written by: Nguyen Chi Hai }

6. Command Summary

This is a summary of all available commands for your reference.

6.1. General commands

These are commands that have consistent format regardless of item type.

Command	Format
Clear	`clear`
Delete	`delete INDEX i/ TYPE`

Command	Format
Done	`done INDEX`
Exit	`exit`
Find	`find KEYWORD [MORE_KEYWORDS]… i/ TYPE`
Help	`help OPTION`
List	`list i/ TYPE`
Me	`me [dp/ FILE_PATH] [n/ NAME] [d/ DESCRIPTION] [p/ PHONE_NUMBER] [e/ EMAIL] [g/ GITHUB] [u/ UNIVERSITY] [m/ MAJOR] [f/ FROM] [t/ TO] [c/ CURRENT_CAP MAX_CAP]`
Redo	`redo`
Sort	`sort i/ TYPE order/ SORT_WORD [reverse/ TRUE_OR_FALSE]`
Undo	`undo`
View	`view INDEX i/ TYPE`

6.2. Item-specific commands

These are commands whose format varies depending on item type.

Command	Type	Format
Add	Internship	`add i/ int n/ COMPANY_NAME r/ ROLE f/ FROM t/ TO d/ DESCRIPTION [#/ TAG]…`
	Project	`add i/ proj n/ PROJECT_NAME t/ TIME w/ WEBSITE d/ DESCRIPTION [#/ TAG]…`
	Skill	`add i/ ski n/ SKILL_NAME l/ LEVEL [#/ TAG]…`
	Resume	`add i/ res n/ RESUME_NAME [#/ TAG]…`
	Note	`add i/ note n/ NOTE_NAME t/ TIME [#/ TAG]…`
Edit	Internship	`edit i/ int [n/ COMPANY_NAME] [r/ ROLE] [f/ FROM] [t/ TO] [d/ DESCRIPTION] [#/ TAG]…`
	Project	`edit i/ proj [n/ PROJECT_NAME] [t/ TIME] [w/ WEBSITE] [d/ DESCRIPTION] [#/ TAG]…`
	Skill	`edit i/ ski [n/ SKILL_NAME] [l/ LEVEL] [#/ TAG]…`
	Resume	`edit i/ res [n/ RESUME_NAME] [#/ TAG]…`
	Note	`add i/ note [n/ NOTE_NAME] [t/ TIME] [#/ TAG]…`

6.3. Resume commands

These are commands specific to `resume` items.

Command	Format
Edit Resume	`redit RESUME_INDEX TYPE/ [ITEM_ID…] [MORE_TYPE/ [ITEM_ID…]]…`

Command	Format
Generate Resume	`rgen RESUME_INDEX n/ FILE_NAME`
Preview Resume	`rpreview RESUME_INDEX`
Tag Pull	`tagpull RESUME_INDEX [#/ TAG]`

{ end of `summary` section written by: Nguyen Chi Hai }