

- Project Direction

Evolve AB3 into a more powerful / optimized contact management app for job recruiters.

Product Name

CANdicates

Target User Profile

Tim is a job recruiter. He needs to store the contact of potential candidates with their relevant details.

The app also tracks candidates according to which job listing they are interested in. It could include features to rate and filter candidates using various criteria such as years of experience and make notes about candidates. [\[See below\]](#)

User Stories

General

1. As a new user exploring the application, I can see sample data stored in the app to show me how it would work when it is in use.
2. As a user that is ready to start using the app, I would like an easy way to clear all the sample data to fill in my data.
3. As a technology inept person, I would like an app that is intuitive to use.
4. As a long term user of the app, I would like a way to archive some of the unused data to prevent cluttering.
5. As an aesthetic individual, I want the app to be clean and aesthetically pleasing.
- 6. As a single person, I would like a way to store the contacts of cute job applicants.**
7. As a veteran job recruiter, I would like a way to store as many contacts as possible without lagging the software.
8. As an efficient person, I would like a way to perform mass operations like deleting multiple entries at once.

List of Candidates

9. **As a job recruiter, I would like a platform to consolidate all information of candidates so that I do not have to receive applications from the same people repeatedly.**
10. As a job recruiter, I would like a way to filter job applicants by their gender.
11. As a job recruiter, I would like a way to filter candidates by the date they apply for the job.
12. **As a job recruiter, I would like a way to filter candidates by the type of job they are applying for.**
13. As a job recruiter, I would like a way to filter job applicants by years of experience.
14. As a job recruiter, I would like a way to blacklist candidates that do not respond or have poor attitudes so that I can take note to not consider them again in the future.
15. As a job recruiter, I would like a way to organise my contact information easily.
16. **As a job recruiter, I would like a way to be able to delete candidates when they are successfully hired so that I can optimise my search.**
17. As a job recruiter, I would like a way to sort candidates based on their expected salary.
18. **As a job recruiter, I would like a way to keep track of duplicate candidates so I will not add multiple times.**
19. As an efficient job recruiter, I would like to have a way to effectively search for candidates whenever there is a new job opening so that I can answer to hiring companies quickly.
20. **As a careless job recruiter, I would like a way to be able to edit the contacts in case there are wrong entries.**
21. **As a job recruiter, I would like to keep the links to candidates' resumes (github, linkedin, etc.) so that I can review them conveniently.**
22. As a job recruiter, I would like a way to be brought directly to the relevant pages to view the candidates' further information.
23. As a job recruiter, I would like a way to be able to view the ranking of the candidates based on various criteria(s) so that I can provide a list of candidates that best fulfil the recruitment criteria(s) to the companies.

List of Job Openings

24. As a job recruiter, I would like a way to be notified of duplicate job listings when adding a new entry to the list.
25. **As a job recruiter, I would like a way to delete job listings when they are filled up.**
26. **As a job recruiter, I would like a way to filter jobs by their vacancies.**
27. **As a job recruiter, I would like a way to tag jobs with different priorities.**
28. **As a job recruiter, I would like a way to filter jobs by their priorities.**
29. As a job recruiter, I would like a way to keep track of interview schedules

Value Proposition

Main problem solved:

Relevant contacts of all applicants are centralised and categorised, reducing the hassle of having to look for applicants repeatedly on different platforms for each new job posting.

Preliminary Features:

- Get the contacts of all job applicants at one glance.
- Manage applicants so that he can easily get back to them.
- Sort applicants by type (such as prospective applicants, rejected applicants, successful etc.)
- Key in job openings.
- Easily find suitable candidates when there is a job opening -- saves time, efficient, larger pool to choose from.
- Sort job applicants according to job listings which they expressed interest for, so that the job recruiter can find information more easily.
- Sort by date of application, so that the recruiter can attend to applicants on a first come first served basis.
- Search by name of applicants so that the recruiter can get access to all contact information that the company requires.
- Delete applicants when they have successfully attained a job.
- Blacklist candidates

Feature list for v1.2:

- ~~1. Delete candidate~~
- ~~2. Add candidate~~
- ~~3. View candidate list~~
- ~~4. Detect duplicate candidate on addition~~
5. Create job listing list - 1 person
6. Delete job listing -
7. Add job listing -
8. View job listings -
9. Tag jobs with priority -
- ~~10. Store candidate info~~
11. Cute candidates -

v1.2 Screenshots:

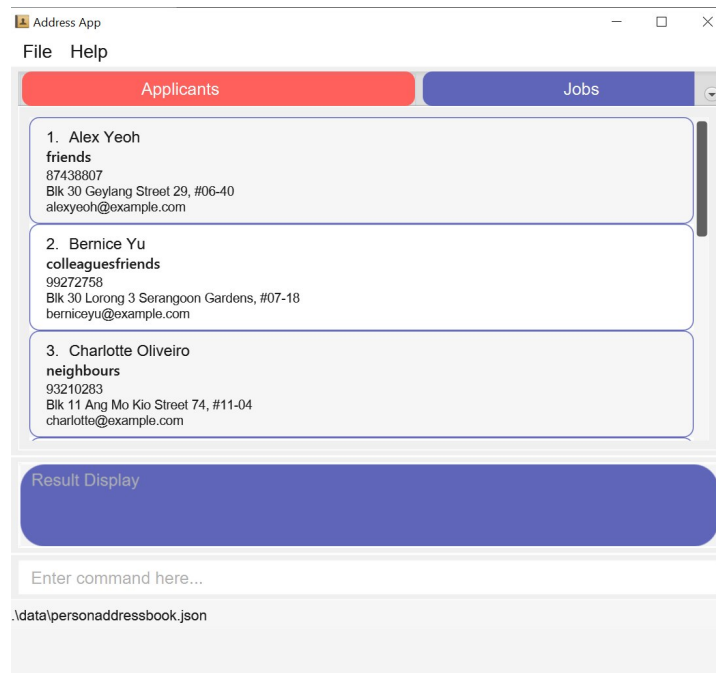


Figure 1: Applicants Address Book

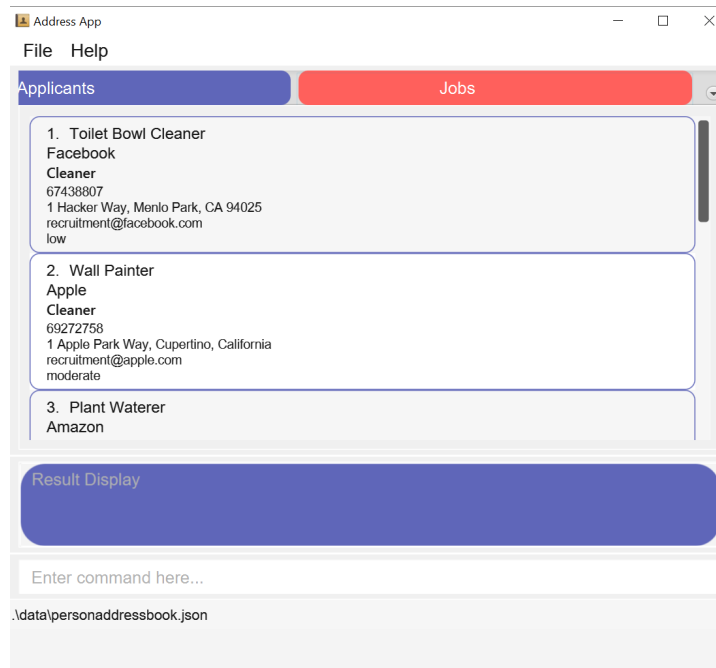


Figure 2: Jobs Address Book

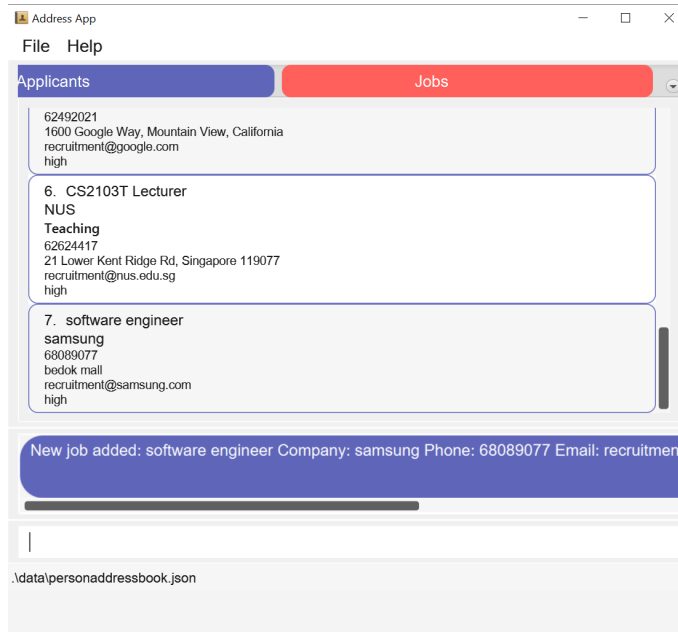


Figure 3: Add new job with all fields

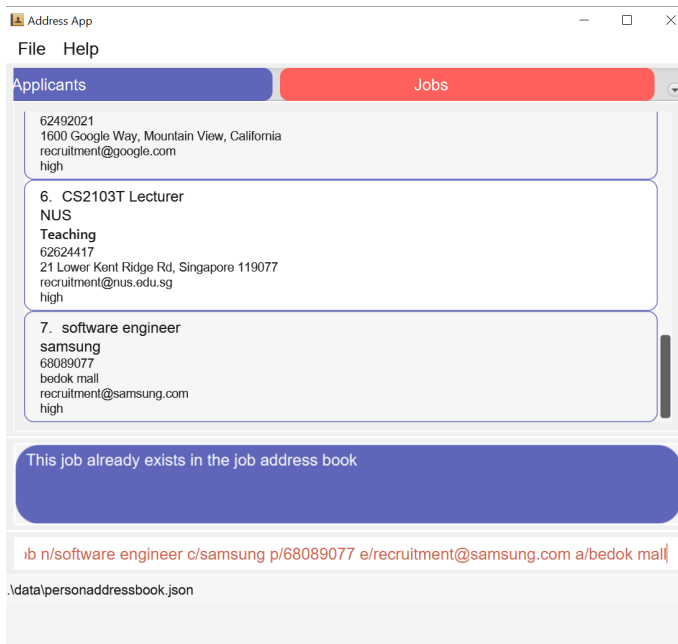


Figure 4: Duplicate job listings

Feature list for v1.3:

1. Person fields: jj

- a. Blacklist: [bl/true] optional field
- b. Years of experience: exp/5 (in years, >=0) compulsory
- c. Date of application: doa/021020 (date format?) compulsory
- d. Expected salary optional -- range?? Value?? Tbc sal/
- e. Links to any profile they like (how to make it clickable ?! @nigel) optional link/[LINK]

2. Job fields: germ

- a. Vacancies v/1

3. Sort person by: nigel

- a. Years of experience: sort can exp/desc
- b. Blacklisted: sort can bl/desc
- c. Date of application sort can d/asc
- d. Expected salary: sort can s/desc

4. Sort job by: germ

- a. Priority: sort job pr/asc
- b. Vacancies: sort job v/asc

5. Search person by:

- a. Any prefix, can chain
find can KEYWORD [MORE_KEYWORDS]...
E.g. Find can n/john exp/5 doa/020202 WHO CHAIN TGT ?!
Find can exp/ : amelia
Find can doa/ : ray
Find can sal/
Find can bl/
Find can n/ -- must edit so only specific things come out lol john doe only. //name
Find can t/ //tags
Find can a/ //address
Find can p/
Find can e/
Find can link/

Returns all results with all fields containing ALL the specified keywords

6. Search job by:

- Find job n/ //jobTitle
- Find job t/ //tags
- Find job a/ //address
- Find job p/

Find job e/

Find job c/

Find job pr/

Find job v/

a. Any prefix, can chain

stuff to edit in dg:

Nigel - pink
Green - ray
Orange - amelia
Purple - germs
Blue - jj

- 2.3 logic component step 3 adding a ~~person~~-candidate **DONE**
- 2.3 logic component execute("delete can 1") **DONE**
- 2.3 logic component note DeletePersonCommandParser **DONE**
- 3.1 step 2 ... to add a candidate with Name John -> John Doe **DONE**
- 3.1 step 4 ... the equals method of ~~Persons~~-Person **DONE**
- 3.1 add command sequence diagram should show all the fields u put in the example command **DONE**
- 3.1 Remove storage component in sequence diagram **DONE**
- 3.1, 3.3- 3.5 at start follow 3.2 "The ... feature has two variants, .." **DONE**
- 3.3 list can instead of list job **DONE**
- ~~- 3.4 step 2 If the type of comparator field e.g. i.e. exp or the order e.g. i.e. asc is missing,~~
- ~~- 3.4 update sort seq diagram ,its outdated~~
- ~~- 3.5 change find seq diagram to be clearer~~
- ~~- 3.5 find seq diagram remove the extra findcommandparser outside the diagram~~
- 3.6 change view seq diagram (setPerson()....)
- ~~- 3.2 - 3.6 after sequence diagram should follow 3.1's format of Note: blah blah analogous to blah blah~~
- ~~- 4.1.1 Sequence diagram should be "undo can", similar for the introduction sentence to the sequence diagram should be undo can instead of just undo~~
- 4.1.1 UML diagram shld be "After command clear can ", addressbook should be personaddressbook **(IDGI) bobian need wipe nigel's ass agn** **DONE**
- 4.1.1 Activity diagram should be "purge redundant states" instead of "redunant" **(IDGI)** **DONE**
- ~~- 4.1.2 remove colon in the header~~
- ~~- 4.1.2.1 alt 2 pros should be `delete can`~~
- ~~- 6.1 value proposition 4th point: job candidates~~
- 6.3 change use case numbers and make sure its references r correct (similar to uc___) **DONE**
- 6.3.8 indent the "use case ends" for 2a extension **DONE**
- 6.3.10 Similar to UC02, except user will request to list jobs and CANDidates will show a list of jobs instead. **DONE**
- 6.3.14 Similar to UC06, except user will request to find jobs with keywords and CANDidates will show a list of jobs instead. **DONE**
- 7.1 "Exit" should be "exit" **DONE**
- 7.1 step 1: Shows the GUI with a set of sample contacts candidates and jobs list. **DONE**
- 7.1 step 2: Shows the GUI with contacts candidates and jobs loaded from the json data files. **DONE**

- 7.1 add breakline between 2. and 3.
- 7.4 & 7.5 Note box very large DONE
- 7.7 add space between "Adding a job" and "to add..." DONE
- 7.12 change "with a 5 candidates" to "with 5 candidates". Add breakline between 1. and 2 DONE
- 7.13 Change "Insert glibberish" to "Insert gibberish" DONE

stuff to edit in ug

- update understanding GUI picture DONE
- change SS for sort
- 2.1 notes and tips change feature to section DONE
- 3 insert line between the warning and tip in the blue box DONE
- 4.1 remove 2nd and 3rd point in blue box DONE
- 4.3 add fullstop after first sentence "you can use the command below" DONE
- 4.4 change find can bl/false and sort can type/sal order/asc to be grey highlight DONE
- 4.3 step 2, 4.6 step 2, 4.8 step 2, 5.2 (note section), 5.3 step 2, 5.4 step 2, 5.5 step 2, 5.6 step 2, 5.8 step 2: remove automatically switch tab thingy bc alr got a subsection on it DONE
- 4.4 step 2 modify to: {Number of candidates found} candidates listed! DONE
- ~~- 4.7 change examples from view candidate to view can~~
- 4.7 in the intro of feature, italics for display panel, similar for 5.7 DONE
- 5.1 remove 2nd and 3rd point in blue box DONE
- 5.4 & 6.4 step 2 modify to: {Number of jobs found} jobs listed! DONE
- 5.4 ~~Similar to the feature above,~~ You can also find job listings by inserting keywords for the fields you are looking for! DONE
- many places: {} put in symbol table?? DONE
- 6.1 help command table shows address as compulsory field DONE
- 6.1 delete rays stupid ss DONE
- 8 add can shows address as compulsory DONE
- 4.3, 4.4, 5.3, 5.4 and 8 should be p/PHONE_NUMBER instead of p/PHONE DONE
- 4.4 & command summary find can should be doa/DATE_OF_APPLICATION instead of doa/APPLICATION_DATE DONE
- 8 find can should be exp/YEARS_OF_EXPERIENCE instead of exp/EXPERIENCE DONE
- 4.3, 8 edit can should have [exp/YEARS_OF_EXPERIENCE] DONE

User Guide

The UG can be found [here](#).

Features

 Notes about the command format:

- Words in `UPPER_CASE` are the parameters to be supplied by the user.
e.g. in `add n/NAME`, `NAME` is a parameter which can be used as `add n/John Doe`.
- Items in square brackets are optional.
e.g. `n/NAME [t/TAG]` can be used as `n/John Doe t/friend` or as `n/John Doe`.
- Items with `...` after them can be used multiple times including zero times.
e.g. `[t/TAG]...` can be used as (i.e. 0 times), `t/friend`, `t/friend t/family` etc.
- Parameters can be in any order.
e.g. if the command specifies `n/NAME p/PHONE_NUMBER`, `p/PHONE_NUMBER n/NAME` is also acceptable

Adding a candidate: `add can`

Adds a candidate to the candidate list.

Format: `add can n/NAME p/PHONE_NUMBER e/EMAIL a/ADDRESS [t/JOB_TYPE]...`



****Tip:**** A person can have any number of tags (including 0).

Examples:

- `add can n/John Doe p/98765432 e/johnd@example.com a/John street, block 123, #01-01 t/Dog Groomer`
- `add can n/Betsy Crowe e/betsycrowe@example.com a/NUS Temasek Hall p/92920033 t/HR t/Office Lady`

Adding a job listing: `add job`

Adds a job to the job listings.

Format: `add job n/COMPANY_NAME p/PHONE_NUMBER e/EMAIL a/ADDRESS [t/JOB_TYPE]
[pr/PRIORITY]`

- Default priority for all jobs is moderate unless tagged otherwise.



****Tip:**** A job listing can have any number of job type tags (including 0)

Examples:

- `add job n/Amelia's Eating House p/98765432 e/ameliatjy@example.com
a/Amelia Street, Block 123, #01-01 t/chef t/waiter pr/low`
- `add job n/FedEx e/fedex@example.com a/Joo Koon p/93333222 t/Delivery Man
pr/high`

Listing all persons : `list can`

Shows a list of all candidates in the address book.

Format: `list can`

Listing all persons : `list job`

Shows a list of all jobs in the job listings.

Format: `list job`

Editing a person : `edit can`

Edits an existing candidate in the address book.

Format: `edit can INDEX [n/NAME] [p/PHONE] [e/EMAIL] [a/ADDRESS] [t/JOB_TYPE]...`

- Edits the candidate at the specified `INDEX`. The index refers to the index number shown in the displayed person list. The index must be a positive integer 1, 2, 3, ...
- At least one of the optional fields must be provided.
- Existing values will be updated to the input values.
- When editing tags, the existing tags of the person will be removed i.e adding of tags is not cumulative.

- You can remove all the person's tags by typing `t/` without specifying any tags after it.

Examples:

- `edit can 1 p/91234567 e/johndoe@example.com` Edits the phone number and email address of the 1st person to be 91234567 and johndoe@example.com respectively.
- `edit can 2 n/Betsy Crower t/` Edits the name of the 2nd person to be Betsy Crower and clears all existing tags.

Editing a job : `edit job`

Edits a job in the address book.

Format: `edit job INDEX [n/COMPANY_NAME] [p/PHONE] [e/EMAIL] [a/ADDRESS] [t/JOB_TYPE]... [pr/PRIORITY]`

- Edits the person at the specified `INDEX`. The index refers to the index number shown in the displayed person list. The index must be a positive integer 1, 2, 3, ...
- At least one of the optional fields must be provided.
- Existing values will be updated to the input values.
- When editing tags, the existing tags of the person will be removed i.e adding of tags is not cumulative.

Examples:

- `edit job 2 n/Dog Cafe p/82827731 a/Dog Street pr/high:` Edits the job name, phone number, address and priority of the 2nd job to be Dog Cafe, 82827731, Dog Street and high.

Deleting a person : `delete can`

Deletes the specified candidate from the address book.

Format: `delete can INDEX`

- Deletes the person at the specified `INDEX`.
- The index refers to the index number shown in the displayed person list.

- The index must be a positive integer 1, 2, 3, ...

Examples:

- `list` followed by `delete can 2` deletes the 2nd person in the address book.
- `find Betsy` followed by `delete can 1` deletes the 1st person in the results of the `find` command.

Deleting a job listing: `delete job`

Deletes the specified job from the job listing.

Format: `delete job INDEX`

- Deletes the person at the specified `INDEX`.
- The index refers to the index number shown in the displayed person list.
- The index must be a positive integer 1, 2, 3, ...

Examples:

- `list` followed by `delete job 2` deletes the 2nd person in the address book.
- `find Betsy` followed by `delete job 1` deletes the 1st person in the results of the `find` command.

Clearing all entries : `clear can`

Clears all entries from the candidate list.

Format: `clear can`

Clearing all entries : `clear job`

Clears all entries from the job listing list.

Format: `clear job`

Exiting the program : `exit`

Exits the program.

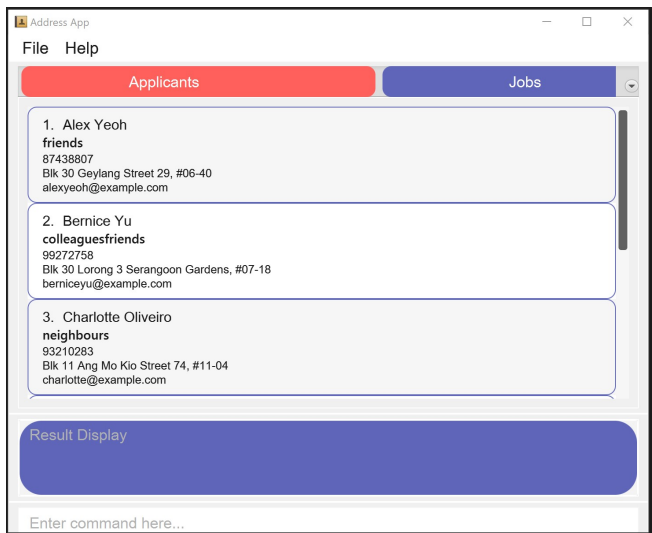
Format: `exit`

Command summary

Action	Candidate Format	Job Listing Format
Add	add can n/NAME p/PHONE_NUMBER e/EMAIL a/ADDRESS [t/JOB_TYPE]...	add job n/COMPANY_NAME p/PHONE_NUMBER e/EMAIL a/ADDRESS [t/JOB_TYPE]... [pr/PRIORITY]
Clear	clear can	clear job
Delete	delete can INDEX	delete job INDEX
Edit	edit can INDEX [n/NAME] [p/PHONE] [e/EMAIL] [a/ADDRESS] [t/JOB_TYPE]...	edit job INDEX [n/COMPANY_NAME] [p/PHONE] [e/EMAIL] [a/ADDRESS] [t/JOB_TYPE]... [pr/PRIORITY]
List	list can	list job
Exit	exit	

V1.2 Demo

V1.2 Demo



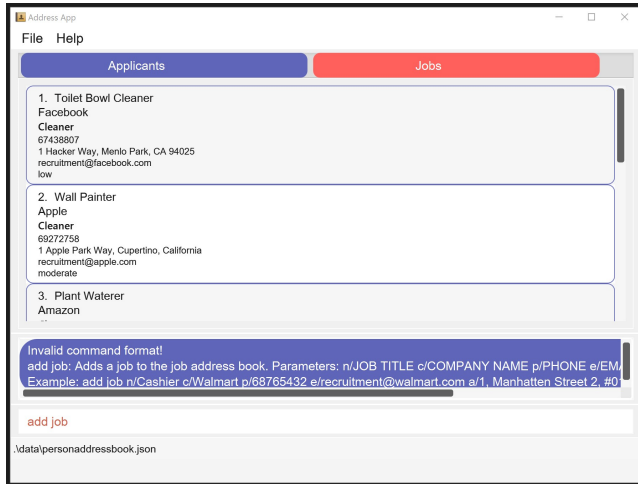
On startup, the user will be greeted by a friendly UI, with sample applicants to guide them in the journey



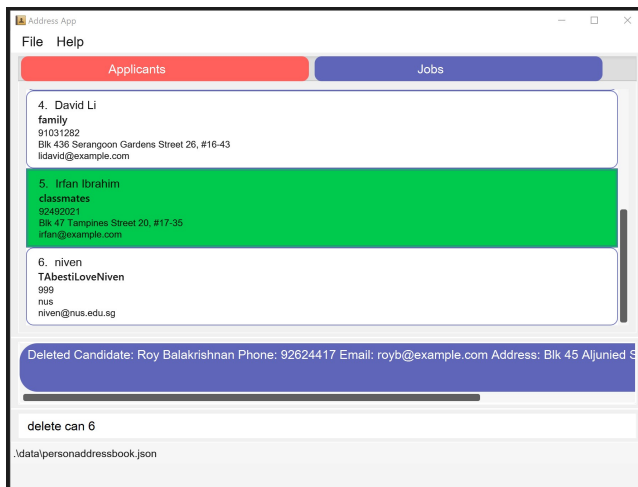
Similarly for the “Jobs” tab



To add an applicant, user can enter the following command:
Add can n/{name} p/{phone} e/{email}
a/{address} t/{optional tag}



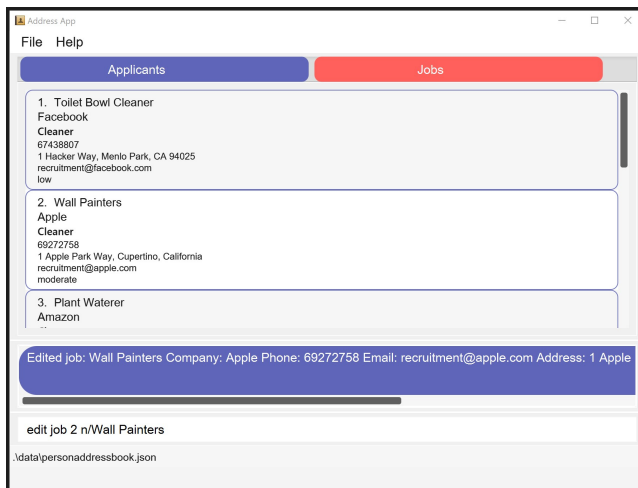
Similar for adding jobs



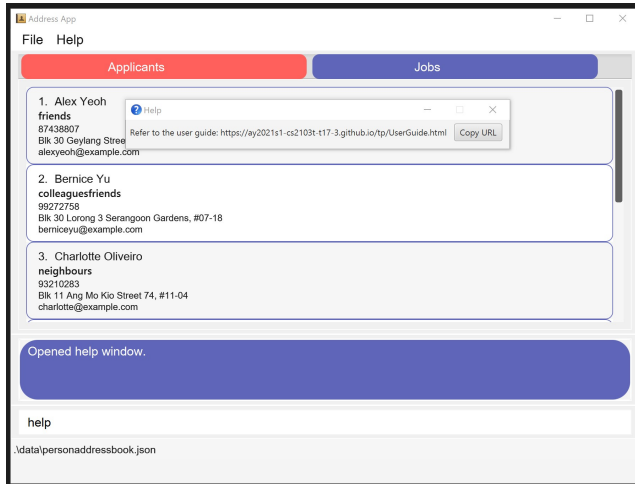
To delete an entry, key in

delete can/job {index}

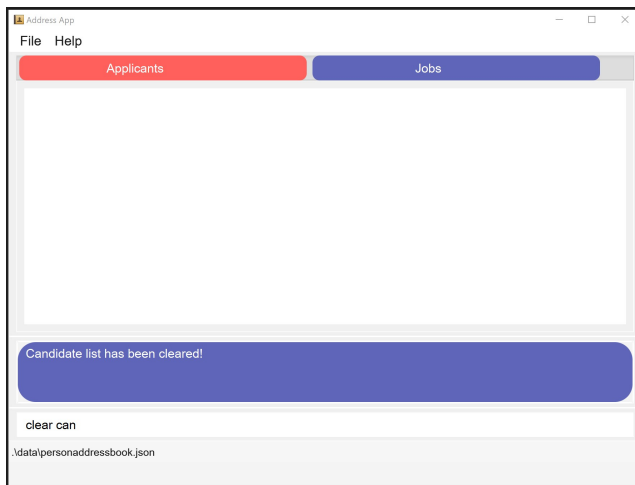
Over here, we deleted Roy Balakrishnan



Entries can also be edited by using
edit job {index} <fields to change>



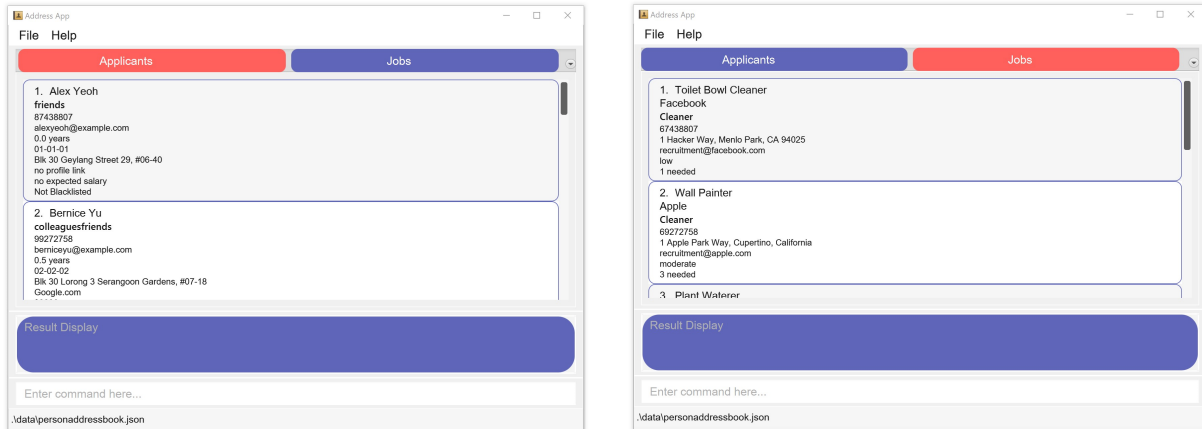
Type help and enter to bring up the help window



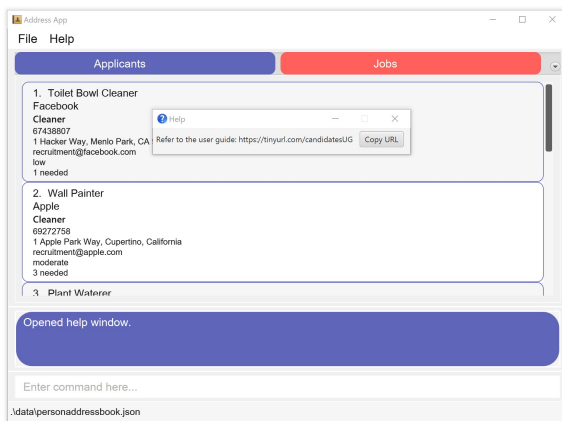
If the user is familiar and ready to start using the application, he can clear the address book by using clear can/job

V1.3 Features Demo

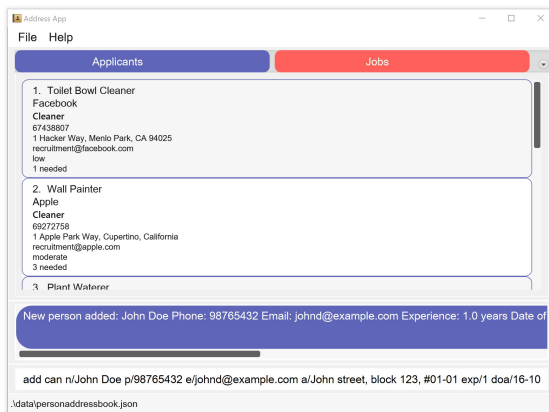
The landing page contains a sample of both candidates and jobs



Typing `help` then enter brings up the help page



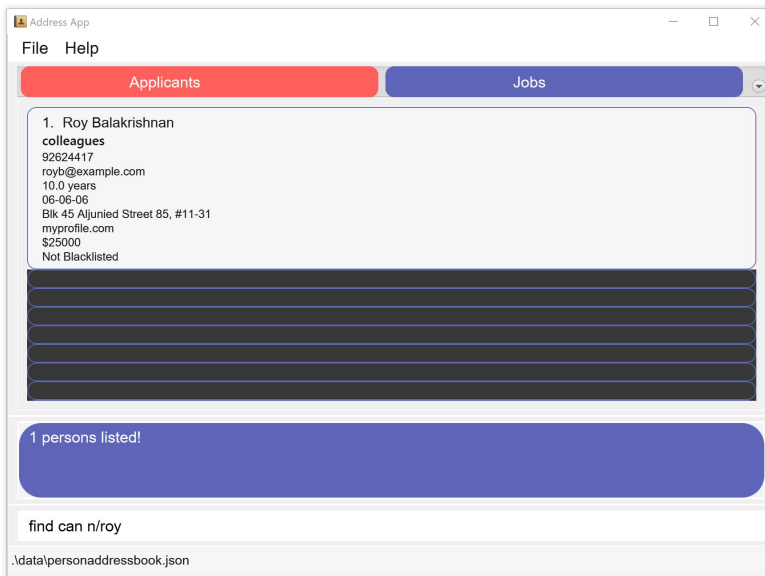
The `add can` or `add job` command adds a candidate or job to the list



The `delete can` or `delete job` command deletes the candidate or job to the list

The `edit can` or `edit job` command deletes the candidate or job to the list

The `find can` or `find job` command deletes the candidate or job to the list, for example, if we look for a person named Roy



The `sort can` or `sort job` command followed by `list can` or `list job` sorts the candidate or job by ascending/descending order of the field specified, for example, if we sort by years of experience in ascending order

