Job Wizard Command Line Interface

Last update: 2025-09-05

JobWizard is a program (`job_wizard`) written in Go. It can be run as a service providing a web-based API, or locally as a command-line application.

Each time JobWizard is invoked on the command line, it executes one of the use cases described in the use case model document. The use case that is executed depends on the `-task` argument. The correspondence between tasks and use cases is as follows:

-task value	Use Case	Action or activity
register	Register	Create a new user in the database
create	Create Job	Create new job posting
search	Search Jobs	General search for jobs
detail	Show Job Details	See detailed information about selected job
offered	Search Offered Jobs	Search for jobs created by me
applied	Search Applied-for Jobs	Search for jobs that I have applied for
modify	Modify Job/Mark Job as Filled	Modify a job created by me
submit	Apply for Job	Submit an application for a job
candidates	View Applicants	Get applicants for a specific job

If you run JobWizard without any arguments, you will get an error.

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard
{ "error" : "Invalid task specified" }
```

To see a general help message, specify the `-help` argument:

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -help

General usage: ./job_wizard -task <taskname> [arguments...]
        Writes results to standard output in JSON format

Available tasks:
        register        Create a new user in the database
        create          Create a new job posting
        search          General search for jobs
        detail          See detailed information about a selected job
        offered         Search for jobs created by me
        applied         Search for jobs that I have applied for
        modify          Modify a job created by me
        submit          Submit an application for a job
        candidates      Get applicants for a specific job

For task-specific arguments, type ./job_wizard -help=true -task <task_name>
```

Once you have figured out what task you want to do, you can specify that task plus `-help` to get detailed information on the command-line arguments accepted/required.

The next pages explain the arguments for each of the tasks in some detail.

Register Task

For simplicity, JobWizard does not require any real authentication. There are no passwords. However, every command/task must have an associated user, identified by an email address. The `register` task creates a new user.

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task register -help

General usage: ./job_wizard -task <taskname> [arguments...]
        Writes results to standard output in JSON format

Register a new email as a JobWizard user
Arguments for register task:
        -email <NEW email address>
        -first <first name>
        -last <last name>
        -phone <10 digit Thai phone>
        -education <integer 0 to 4>
All arguments are required

Example: ./job_wizard -task register -email sally@gmail.com -first Sally -last Goldin -phone 0987651122 -education 4
```

Education is a code from 0 to 4, where 0 is unknown or not specified; 1 means grade school diploma; 2 means high school diploma; 3 means bachelors degree; 4 means graduate degree.

Here is an example of a successful registration.

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task register -email fred@gmail.com -first Frederick -last Roach -phone 0
851235422 -education 3
{ "success" : "Registered user fred@gmail.com"}
```

Note that the email must be unique. So if we try the same command again, we will get an error:

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task register -email fred@gmail.com -first Frederick -last Roach -phone 0
851235422 -education 3
{ "error" : "Email is not unique; user not created" }
```

Errors can also be produced if required values are missing or in the wrong format.

Create Task

Creating a new job is one of the more complicated operations since you must specify the job details, including a title and description. Since both these strings can be multiple words, they must be enclosed in quotation marks.

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task create -help

General usage: ./job_wizard -task <taskname> [arguments...]
       Writes results to standard output in JSON format

Create a new job posting in the JobWizard database
Arguments for create task:
        -creator <email of registered user>
        -title <job title in quotes>
        -description <job description in quotes, up to 1024 chars>
        -min_education <integer 0 to 4>
        -min_experience <integer 0 to 75>
        -salary <monthly salary in baht, 0 means unspecified>
Creator, title and description are required

Example: ./job_wizard -task create -creator sally@gmail.com -title "Front End Developer" -description "Build user interfaces for enterpris
e web applications" -min_education 2 -salary 35000
```

An example job creation action is show below:

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task create -creator sally@cmkl.ac.th -title "System Architect" -descript
ion "Take primary responsibility for design and implementation of edtech applications" -min_education 3 -salary 85000
{ "job_id" : "00007" }
```

If the job creation is successful, the id of the new job is returned. This job id can be used in the `detail` task to see specific information, or in the `submit` task to apply for the job.

Search Task

The `search` task allows you to view a list of jobs that satisfy certain criteria, or all jobs in the data base.

The only required argument is `-email` which is the email of the registered user making the search request. However, you can also request filtering by any combination of the data in a job record. You can use the `-keyword` argument to search for a string within the title or description (exact match only).

The arguments and several examples of searches are shown on the next page.

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task search -help

General usage: ./job_wizard -task <taskname> [arguments...]
        Writes results to standard output in JSON format

Search for jobs based on criteria, and print summaries
Arguments for search task:
        -email <email of registered user>
        -min_education <integer 1 to 4>
        -min_experience <integer >
        -salary <monthly salary in baht>
        -posted <date: YYYY-MM-DD>
        -keyword <keyword to search for in title>
Only email is required

Example: ./job_wizard -task search -email sally@gmail.com -salary 30000 -keyword Developer
```

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task search -email sally@cmkl.ac.th -keyword Develope
r -min_experience 2
[{"job_id":"00002","title":"Back End Developer","is_open":true,"date_posted":"2025-06-27"},{"job_id":"00001","title":"
Front End Developer","is_open":true,"date_posted":"2025-06-27"}]
```

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task search -email sally@cmkl.ac.th -salary 1000000
{ "warning" : "No matching jobs found"}
```

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task search -email sally@cmkl.ac.th
[{"job_id":"00007","title":"System Architect","is_open":true,"date_posted":"2025-09-05"},{"job_id":"00004","title":"Ex
ecutive Secretary","is_open":true,"date_posted":"2025-07-02"},{"job_id":"00005","title":"Student Relations Officer","i
s_open":true,"date_posted":"2025-07-02"},{"job_id":"00006","title":"Professor","is_open":true,"date_posted":"2025-07-0
2"},{"job_id":"00003","title":"HR Director","is_open":true,"date_posted":"2025-06-27"},{"job_id":"00002","title":"Back
 End Developer","is_open":true,"date_posted":"2025-06-27"},{"job_id":"00001","title":"Front End Developer","is_open":t
rue,"date_posted":"2025-06-27"}]
```

Detail Task

The information provided by the search task lists only a subset of the information stored about a job. For instance, it does not list salary information, or description. To see the full set of information about a job, execute the `detail` task.

The task requires an email identifying the user plus the job id.

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task detail -help

General usage: ./job_wizard -task <taskname> [arguments...]
        Writes results to standard output in JSON format

Return all detailed information for a specific job
Arguments for detail task:
        -email <email of registered user>
        -job_id <show detail for what job>

All arguments are required

Example: ./job_wizard -task detail -email sally@gmail.com -job_id 00003
```

Here is an example:

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task detail -email sally@cmkl.ac.th -job_id 00006
{"job_id":"00006","creator":"sally@cmkl.ac.th","title":"Professor","description":"Teaching and research to support the
 university ","min_education":4,"min_experience":5,"salary":95000,"is_open":true,"date_posted":"2025-07-02"}
```

Offered Task

This task displays a summary of all the jobs that have been created by the specified user.

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task offered -help

General usage: ./job_wizard -task <taskname> [arguments...]
        Writes results to standard output in JSON format

Return summaries for all jobs created/posted by a user
Arguments for offered task:
        -creator <email of registered job creator>
All arguments are required

Example: ./job_wizard -task offered -creator sally@gmail.com
```

For example:

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task offered -creator sally@cmkl.ac.th
[{"job_id":"00007","title":"System Architect","is_open":true,"date_posted":"2025-09-05"},{"job_id":"00006","title":"Pr
ofessor","is_open":true,"date_posted":"2025-07-02"},{"job_id":"00002","title":"Back End Developer","is_open":true,"dat
e_posted":"2025-06-27"},{"job_id":"00001","title":"Front End Developer","is_open":true,"date_posted":"2025-06-27"}]
```

The `detail` task can be used to get full information about any of these jobs.

Applied Task

You can use the `applied` task to find out which jobs a particular user has applied for.

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task applied -help

General usage: ./job_wizard -task <taskname> [arguments...]
        Writes results to standard output in JSON format

Return summaries for all jobs a user has applied for
Arguments for applied task:
        -email <email of registered user>
All arguments are required

Example: ./job_wizard -task applied -email sally@gmail.com
```

Like the `offered` task, it has a single argument, a user email. Note that the argument name is `-email` rather than `-creator`, though. Here is an example:

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task applied -email sally@cmkl.ac.th
[{"job_id":"00002","title":"Back End Developer","is_open":true,"date_posted":"2025-06-27"}]
```

Modify Task

The creator of a job can use the `modify` task to change any of its information (except the job_id, which is assigned by the system). In particular, the creator can set the status of a job to indicate that it has been filled. Once a job has been filled, any attempt by a use to apply for that job will fail. The arguments for this task are quite complex, since any attribute of the job could be modified. Most arguments are optional.

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task modify -help

General usage: ./job_wizard -task <taskname> [arguments...]
        Writes results to standard output in JSON format

Modify some attributes of a specific job
Arguments for modify task:
        -creator <email of registered user>
        -job_id <modify what job>
        -title <job title in quotes>
        -description <job description in quotes, up to 1024 chars>
        -min_education <integer 0 to 4>
        -min_experience <integer 0 to 75>
        -salary <monthly salary in baht, 0 means unspecified>
        -is_open=false
Creator and job_id are required, changes any other attributes specified

Example: ./job_wizard -task modify -creator sally@gmail.com -job_id 00002 -title "User Experience Developer" -salary 3
8000
```

Here is an example, setting a job to be filled.

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task modify -creator sally@cmkl.ac.th -job_id 00007 -
is_open=false
{ "modified_job_id" : "00007" }
```

Important note: Due to a peculiarity in Go's handling of boolean command line arguments, you must use an equal sign (as shown above) to set the value. If you forget this, you will get an obscure error message as shown below:

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task modify -creator sally@cmkl.ac.th -job_id 00007 -
is_open false
{ "error" : "near "WHERE": syntax error" }
```

Submit Task

Use the `submit` task to apply for a job. There are two arguments, the user applying and the job id.

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task submit -help

General usage: ./job_wizard -task <taskname> [arguments...]
        Writes results to standard output in JSON format

Apply for a particular job (submit application)
Arguments for submit task:
        -email <email of registered user>
        -job_id <apply for what job>

All arguments are required

Example: ./job_wizard -task submit -email sally@gmail.com -job_id 00014
```

Here are two examples:

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task submit -email joe@cmkl.ac.th -job_id 00005
{ "error" : "Creator cannot submit an application for their own job" }

(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task submit -email sally@cmkl.ac.th -job_id 00005
{ "applied_job_id" : "00005" }
```

As noted above, an attempt to apply to a filled job will produce an error message:

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task submit -email joe@cmkl.ac.th -job_id 00007
{ "error" : "Job has already been filled" }
```

Candidates Task

The `candidates` task allows a job creator to see which other users have applied to a job that user created.

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task candidates -help

General usage: ./job_wizard -task <taskname> [arguments...]
        Writes results to standard output in JSON format

Return candidates for a specific job
Arguments for candidates task:
        -creator <email of job creator>
        -job_id <show candidates for what job>

All arguments are required

Example: ./job_wizard -task candidates -creator sally@gmail.com -job_id 00003
```

Here is an example:

```
(base) goldin@robusto:/devel/JobWizard/job_wizard> ./job_wizard -task candidates -creator joe@cmkl.ac.th -job_id 00005
[{"email":"sally@cmkl.ac.th","name":"Sally Goldin","phone":"“0879990088”","applied_date":"2025-09-05"}]
```