The general newman code will look like as follows:

C:\Users\Pratik\Desktop>newman run collection.json -e environment.json -n 2 -r html,cli,json,junit

Description:

1. C:\Users\Pratik\Desktop> : Specifies the directory in why the exported collection, environment and data file is saved in the system. Also the location where the newman reports are saved.
2. Collection.json : name of the exported collection file. (Note: Do not name your collection with a space in between e.g.: abc collection.json because the space is not accepted in commend line while running the code.)
3. –e environmebt.json : name of the exported environment file. (Note: Do not name your environment with a space in between e.g.: abc environment.json because the space is not accepted in commend line while running the code.)
4. –n 2 : Specifies the number of times the collection has to be run when used in conjunction with iteration data file.
5. -r html,cli,json,junit : Specifies the type of reports to be generated.

**Some additional commands are listed below (reference :** [**https://github.com/postmanlabs/newman#newman-run-collection-file-source-options**](https://github.com/postmanlabs/newman#newman-run-collection-file-source-options) **)**

Command line options:

newman run <collection-file-source> [options]

-e <source>, --environment <source>

Specify an environment file path or URL. Environments provide a set of variables that one can use within collections. Read More

-g <source>, --globals <source>

Specify file path or URL for global variables. Global variables are similar to environment variables but has a lower precedence and can be overridden by environment variables having same name.

-d <source>, --iteration-data <source>

Specify a data source file (CSV) to be used for iteration as a path to a file or as a URL. Read More

-n <number>, --iteration-count <number>

Specifies the number of times the collection has to be run when used in conjunction with iteration data file.

--folder <name>

Run requests within a particular folder in a collection.

--export-environment <path>

The path to the file where Newman will output the final environment variables file before completing a run.

--export-globals <path>

The path to the file where Newman will output the final global variables file before completing a run.

--export-collection <path>

The path to the file where Newman will output the final collection file before completing a run.

--timeout <ms>

Specify the time (in milliseconds) to wait for the entire collection run to complete execution.

--timeout-request <ms>

Specify the time (in milliseconds) to wait for requests to return a response.

--timeout-script <ms>

Specify the time (in milliseconds) to wait for scripts to complete execution.

-k, --insecure

Disables SSL verification checks and allows self-signed SSL certificates.

--ignore-redirects

Prevents newman from automatically following 3XX redirect responses.

--delay-request

Specify the extent of delay between requests (milliseconds).

--bail [optional modifiers]

Specify whether or not to stop a collection run on encountering the first test script error.

Can optionally accept modifiers, currently include folder and failure.

folder allows you to skip the entire collection run in case an invalid folder was specified using the --folder option or an error was encountered in general.

On the failure of a test, failure would gracefully stop a collection run after completing the current test script.

-x, --suppress-exit-code

Specify whether or not to override the default exit code for the current run.

--color

Use this option to force colored CLI output (for use in CLI for CI / non TTY environments).

--no-color

Newman attempts to automatically turn off color output to terminals when it detects the lack of color support. With this property, one can forcibly turn off the usage of color in terminal output for reporters and other parts of Newman that output to console.

--disable-unicode

Specify whether or not to force the unicode disable option. When supplied, all symbols in the output will be replaced by their plain text equivalents.

--global-var "<global-variable-name>=<global-variable-value>"

Allows the specification of global variables via the command line, in a key=value format. Multiple CLI global variables can be added by using --global-var multiple times, like so: --global-var "foo=bar" --global-var "alpha=beta".