



Kan-CLI

Command line Kanban board

```

Welcome to
Kan-CLI

What do you want to do? (Use ↑/↓ arrow keys, press Enter to select)
> Sign Up
Exit
```



App Overview

“A **kanban board** is an agile project management tool designed to help visualize work, limit work-in-progress, and maximize efficiency (or flow).” - <https://www.atlassian.com/agile/kanban/boards>

Kanban boards allow individuals or teams to:

- Organise tasks into groups.
- Visualise the status of tasks within a project.
- Limit work in progress and maximise efficiency.

Kanban Board Example

The screenshot shows a Trello board titled "Agile Sprint Board" with a blue background. The board is organized into five columns: "Agile Development Template:", "Backlog", "Sprint Backlog", "In Progress", and "8.9.17 Sprint - Complete".

- Agile Development Template:** Contains five cards with text about agile workflow, code accessibility, collaboration, backlog management, and sharing updates. Each card has a "More" (three dots) icon at the bottom.
- Backlog:** Contains three cards. The first card has labels "Meta", "Verified on branch", "Bugs", "Blocked", and "Regression". The second card is "(3) Pre-load card attachments" with 1 comment. The third card is "(8) renderable CardDetailView" with 4 comments. There is a "Blocked" label at the bottom.
- Sprint Backlog:** Contains two cards. The first card has a label "New Team / boards tab" and text "(8) Clicking the collection beneath a board should filter by collection, not open collections pop-over". The second card has a label "Security Issue" and text "(1) Add post-message-io" with 3 comments. There is a "+ Add another card" button at the bottom.
- In Progress:** Contains four cards. The first card has a label "Web" and text "Multiple due dates". The second card has a label "Verified on branch" and text "(5) EditableView" with 1 comment. The third card has a label "Blocked" and text "(21) Update CSS" with 8 comments. The fourth card has text "(1) Attach URLs from comment" with 2 comments. There is a "+ Add another card" button at the bottom.
- 8.9.17 Sprint - Complete:** Contains four cards. The first card has a label "Bugs" and text "(8) Let the server choose the default name when creating a card from a URL" with 6 comments and 2/3 attachments. The second card has text "(3) plugins: attachment preview icon" with 3 comments. The third card has a label "Verified on staging" and text "(1) Decouple board page list CSS" with 2 comments. The fourth card has a label "Verified on staging" and text "(2) Restructure KnownUrls" with 3 comments. There is a "+ Add another card" button at the bottom.

The right sidebar shows a list of boards with a search bar and a "Show Menu" button.



Features

Users can:

- Create boards for different projects
- Create multiple Lists within each board
- Create cards and add them to lists (and limit the number of cards that can be added)
- Move cards between lists
- Switch between boards



Classes

Board

- A title (String)
- A creation date (Time object)
- A collection of lists (Hash data type; List title => List object)

The user will be prompted to create a board object when the app is run. They can also create more boards from the menu. Each time a board is created it is set as the “current board”. The user is able to change the current board.

List

- Title (String)
- Card limit (Integer - the maximum number of cards the list can contain, user defined)
- Cards (Hash: card id => card object)

List objects can be created at any time from the menu, they are stored within the currently active board object. Lists are stored in a hash - the key is the title and the value is the list object itself.



Classes

Cards

- ID
- Description
- Creation date

Functionality

Boards have methods for adding and deleting lists, as well as displaying in the terminal.

Lists have functionality for adding and deleting cards. Moving cards is achieved by adding to the new list and then deleting from the old list.

Cards don't have functionality as they are just placeholders for task description.



Code

```
# Store all of the users  
variables
```

```
# which describe the  
state of the application
```

```
$state = {  
  "user" => {},  
  "boards" => {},  
  "current board" => nil  
}
```

The state variable stores all of user input.

- The username and password of the current user
- Each time a user creates a board it is added to the “boards” hash.
- The currently active board is used to determine which board should be displayed in the terminal. This variable is updated whenever the user creates a new board.



Code

Control flow is handled through a combination of the `input_loop` function and `display_menu` function

```
# Methods as array elements (https://stackoverflow.com/questions/13948910/ruby-methods-as-array-elements-how-do-they-work)
input_flow = [method(:create_board), method(:create_list), method(:create_card)]

# Start the loop at a given position
if start_position == "board"
  input_flow.each { |m| m.call }
elsif start_position == "list"
  input_flow[1..2].each { |m| m.call }
elsif start_position == "card"
  input_flow[2].call
end

# Show the user their updated board
$state["current board"].display_board
```




Code

display_menu

```
# Define the menu
menu = $prompt.select("What next?") do |menu|
  menu.default 1

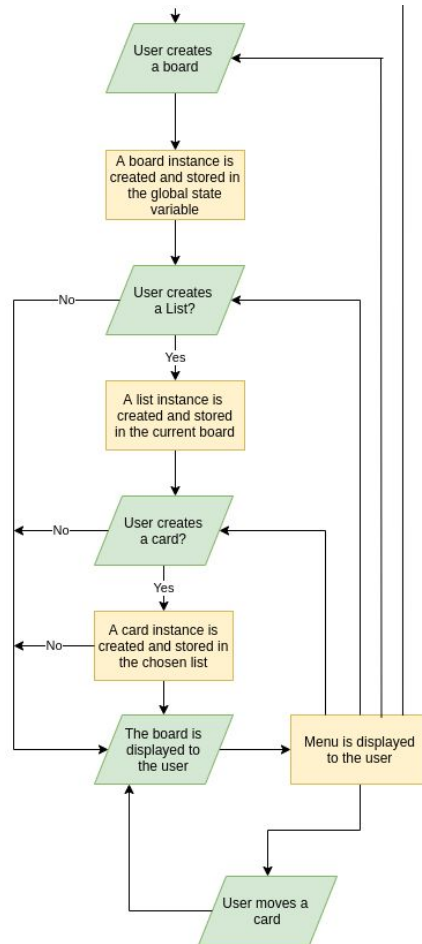
  menu.choice 'Create a New Board', 5
  menu.choice 'Switch Board', 6
  menu.choice 'Add a List', 1
  menu.choice 'Add a Card', 2
  menu.choice 'Move card', 3
  menu.choice 'Exit', 4
end
```



Code

display_menu

```
# Handle user input
case menu
when 1
  | input_loop("list")
when 2
  | input_loop("card")
when 3
  | move_card
when 4
  | exit
when 5
  | input_loop("board")
when 6
  | switch_board
end
```





Demonstration