Development Walkthrough: KanbanEase - A Kanban Board Application

Sherjan F. Marun

Professional Learning Practice – Front-End Development

ABSTRACT:
The development of KanbanEase was based on the concept of the "Kanban method," originally developed by Toyota to organize tasks and manage their status. Initially, the application was sketched on paper to serve as a visual reference during development. The necessary technologies were selected, including React, Tailwind CSS, Vite, and libraries such as @hello-pangea/dnd and react-icons. To start the project, it was decided to begin with a basic ToDo application, which helped solidify fundamental concepts, such as adding and deleting tasks using React hooks. Next, the "drag and drop" functionality was implemented within a single task container, starting with the Y-axis. Once this was working, a second container was added to allow dragging on the X-axis, and the code was adjusted to enable the reordering of tasks both within the same container (Y-axis) and between different containers (X-axis). During development, CSS was used to enhance the visualization of partial progress. Issues related to duplicate indexes, which caused unexpected behavior when dragging and dropping tasks within the same container and between different containers, were identified and resolved. Additionally, functions were implemented to add and delete containers, which act as workspaces for tasks. Dynamic forms were also created to add tasks to each container, ensuring that only one form was active at a time. A similar approach was applied to task editing forms, where an autofocus function was implemented to activate the text area when the form is opened. Finally, a function was developed to close the active form when clicking outside the corresponding container. Tailwind CSS was used to style the application, resulting in a visually appealing and functional interface.

Keywords: *Kanban, React, Drag and Drop, Tailwind, Task Management.*

1. Introduction

Origin of the Kanban Methodology

The Kanban method originated in the Japanese automotive industry, specifically at Toyota, during the 1940s. It was developed by Taiichi Ono and his team, who were inspired by the supermarket system to optimize efficiency and reduce waste in production. The term "Kanban" means "visual card" in Japanese, and cards were used to control production at each stage of the process, ensuring production was aligned with demand and enabling a quick response to changes in orders.

Today, the Kanban method has transcended the manufacturing industry and has been implemented in various fields, such as software development and project management. It has become a flexible and visual approach to improving organization, workflow, and efficiency in different contexts.[1]

Advantages of the Kanban Methodology

- The paperwork to manage replenishment processes is minimal.
- It allows for a reduction in the number of resources dedicated to managing order needs or stock.
- The workers themselves are responsible for maintaining the process.
- Stock between processes is reduced by aligning the "customer-supplier" relationship, whether internal or external, thanks to the tight or "pull" flow.
- It ensures the necessary stock to continue work.
- It facilitates replenishment closer to the point of use by reducing stock and allowing for more continuous replenishment, which enables serving in smaller quantities and, therefore, using smaller containers that occupy less space at the line edges.

It allows for controlling production and knowing its status at any moment.[2]

How Does the Kanban Method Work?

Today, Kanban boards have largely evolved into virtual versions with columns representing the different stages of work (although traditional Kanban boards using whiteboards and post-its are still in use). Each Kanban card on the board represents a specific task, moving through the stages of work as it progresses. While tasks are often assigned to individual team members, everyone collaborates on a single Kanban board for increased efficiency and coordination.

The Principles of the Kanban Methodology

To optimize tasks, the methodology is based on five fundamental principles:

- Visualization: Kanban is fully transparent, allowing one to understand at which stage of development the project is.
- Prioritization: When addressing the backlog of tasks, the responsible person has a clear idea of the next item to tackle.
- Continuous Improvement.
- Leadership at All Levels.
- Assured Quality.

Autonomous Production Management

Kanban is essentially a tool that enables employees to take responsibility for managing their own work. Imagine a worker removing a Kanban card from a box or drawer of parts before using them in assembly. The worker then sends this Kanban card back to the previous process as an order for new parts to replace the ones being used. This worker is supporting a significant part of the task of managing part requests and stock management.[3]

2. Technologies and Libraries

Technologies:

HTML, CSS, JavaScript, React version.18, Tailwind css version.4.0, Vite js version.5.5.2 [4 – 5 - 6]

Libraries:

@hello-pangea/dnd versión 16.6.0, React Icons versión 5.2.1 [7 - 8]

3. Methodology

Research and Concept: Research was conducted on the functionality and purpose of Toyota's Kanban method.

The app was initially sketched out on paper to provide a visual reference for development.

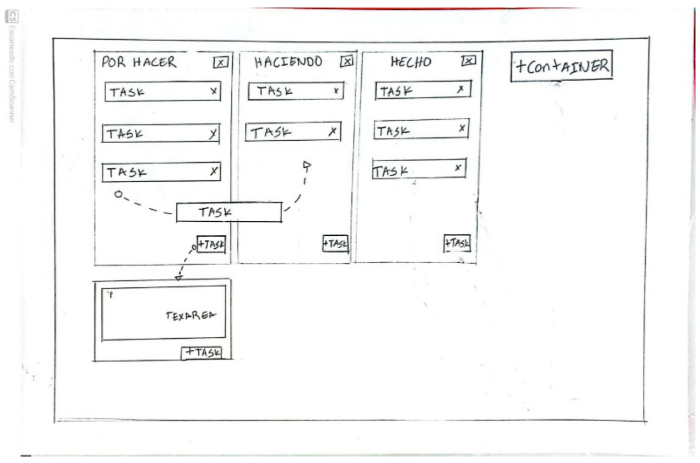


Figure 1. *Sketch of the app*

Technology Selection:

React, Tailwind CSS, Vite, and libraries such as @hello-pangea/dnd and react-icons were selected.

Project Start:

Once the development environment with React and Vite was set up, we began by creating a basic ToDo app that allowed us to add and delete tasks using React hooks useState and useEffect.

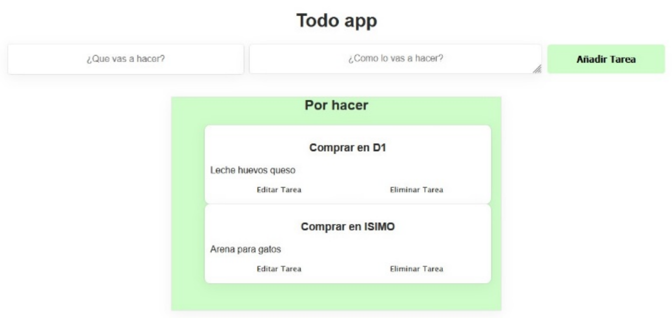
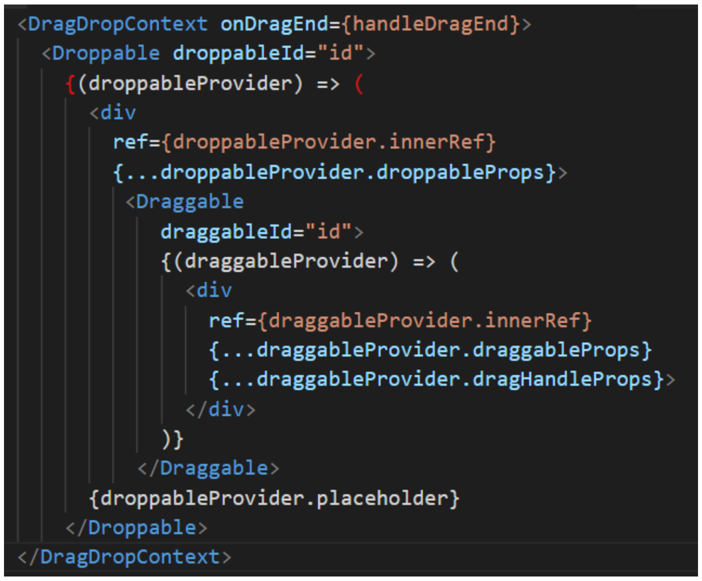


Figure 2. *Screenshot of the initial ToDo app*

The structure indicated by the documentation of the dnd library was used to handle drag functionality. [4]



```
<DragDropContext onDragEnd={handleDragEnd}>
  <Droppable droppableId="id">
    {(droppableProvider) => (
      <div
        ref={droppableProvider.innerRef}
        {...droppableProvider.droppableProps}>
         <Draggable
           draggableId="id">
           {(draggableProvider) => (
             <div
               ref={draggableProvider.innerRef}
               {...draggableProvider.draggableProps}
               {...draggableProvider.dragHandleProps}>
             </div>
           )}
         </Draggable>
      {droppableProvider.placeholder}
    </Droppable>
</DragDropContext>
```

Figure 3. *Shows the code structure for drag and drop*

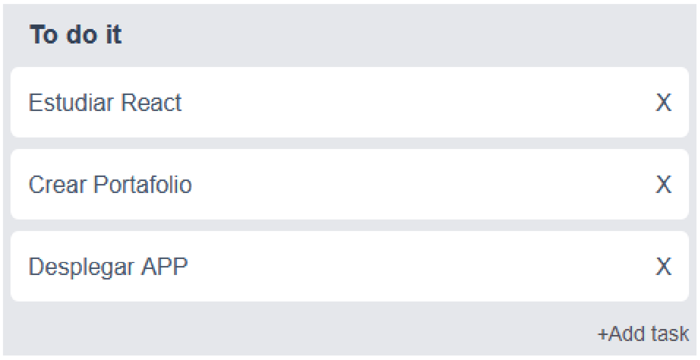
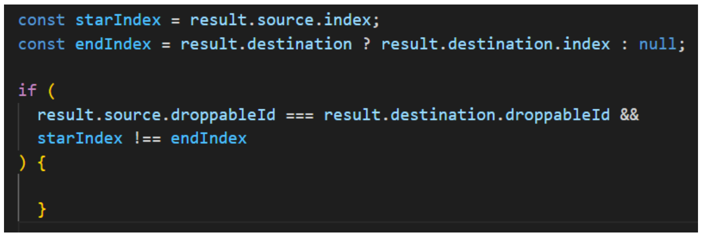


Figure 4. *First container where drag functionality works on the Y axis*

To handle the drop, the onDragEnd event was used to call the handleDragEnd() function.

It detects if the drag exists and if the movement is within the same container, then updates the state accordingly by reordering the items.



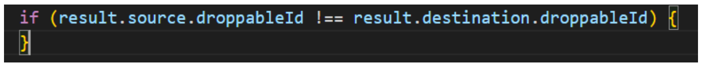
```
const starIndex = result.source.index;
const endIndex = result.destination ? result.destination.index : null;

if (
  result.source.droppableId === result.destination.droppableId &&
  starIndex !== endIndex
) {

  }
```

Figure 5. *Conditionals that detect if the drag and drop occur within the same container*

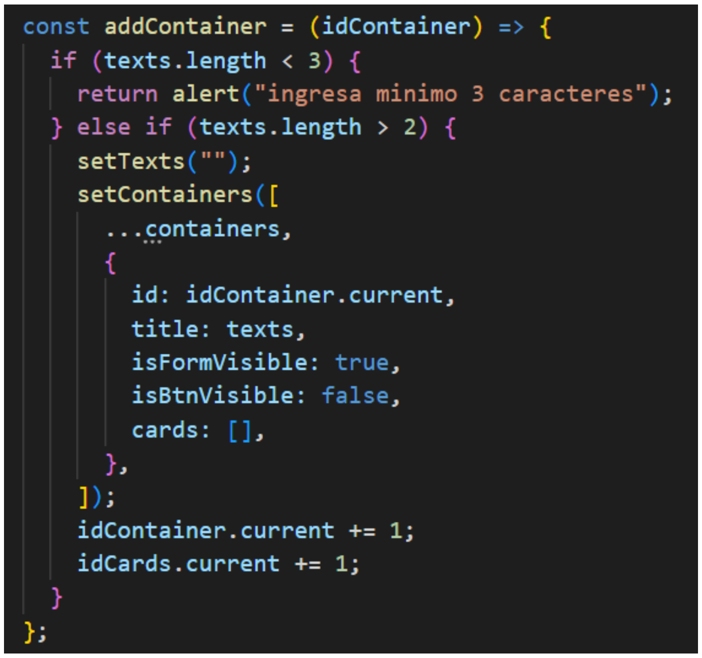
It detects if the drag exists and if the movement is outside the container, and whether the destination container exists. It then updates the state by removing the task from the source container and adding it to the destination container, using methods like splice.



```
if (result.source.droppableId !== result.destination.droppableId) {
}
```

Figure 6. *Conditionals that detect if the drag and drop occur in a different container*

The code was adjusted to dynamically add containers.



```
const addContainer = (idContainer) => {
  if (texts.length < 3) {
    return alert("ingresa minimo 3 caracteres");
  } else if (texts.length > 2) {
    setTexts("");
    setContainers([
      ...containers,
      {
        id: idContainer.current,
        title: texts,
        isFormVisible: true,
        isBtnVisible: false,
        cards: [],
      },
    ]);
    idContainer.current += 1;
    idCards.current += 1;
  }
};
```

Figure 7. *addContainer Function*

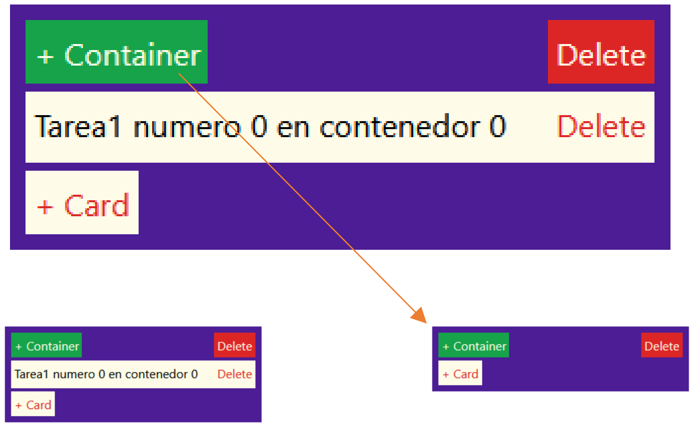
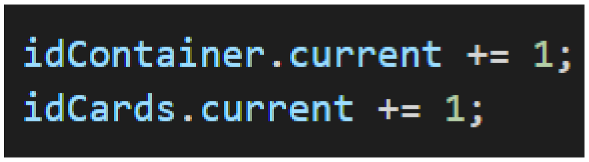


Figure 8. *Containers being added*

Issues with duplicate indexes were identified, causing any card with the same index to move when another card was selected. This was resolved by using the useRef hook to ensure that the next index is always available when adding a card or a new container, preventing duplicate indexes.[5]



```
idContainer.current += 1;
idCards.current += 1;
```

Figure 9. *Update of IDs*

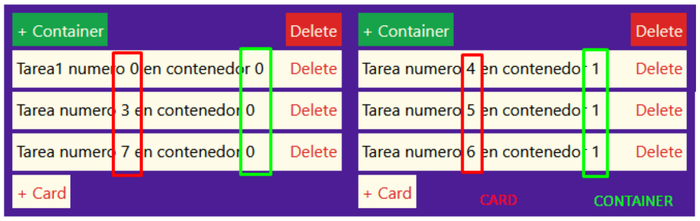

+ Container
Delete
Tarea1 numero 0 en contenedor 0
Tarea numero 3 en contenedor 0
Tarea numero 7 en contenedor 0
Tarea numero 4 en contenedor 1
Tarea numero 5 en contenedor 1
Tarea numero 6 en contenedor 1
+ Card
CARD
CONTAINER

Figure 10. *Non-repeating IDs*

It was determined that the problem was resolved, and development continued. The drag and drop functionality worked correctly.

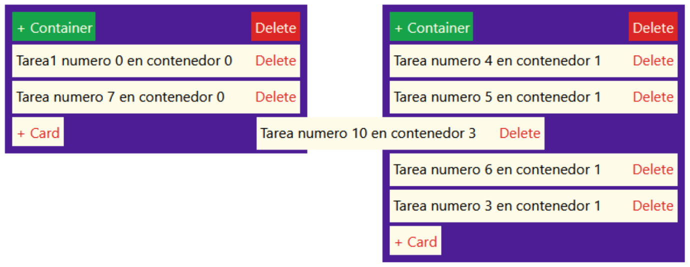

+ Container
Delete
Tarea1 numero 0 en contenedor 0
Tarea numero 7 en contenedor 0
+ Card
Tarea numero 10 en contenedor 3
Tarea numero 4 en contenedor 1
Tarea numero 5 en contenedor 1
Tarea numero 6 en contenedor 1
Tarea numero 3 en contenedor 1

Figure 11. *Drag and drop functioning correctly*

The app was styled using Tailwind CSS.

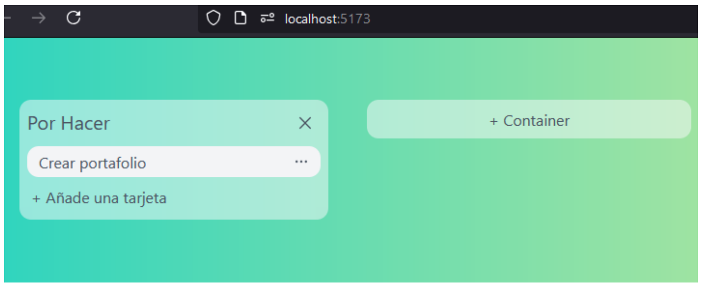


Figure 12. *App updated with Tailwind CSS*

Forms were added to create cards and containers.

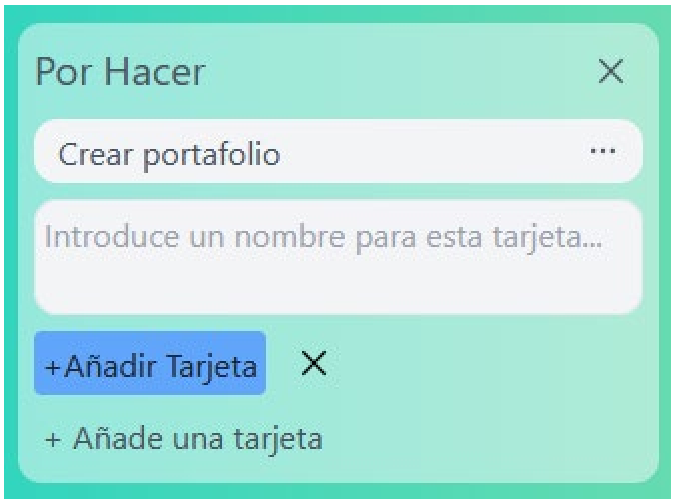


Figure 13. *Add task form*

Autofocus was implemented on the active input using the useEffect and useRef hooks. [6]

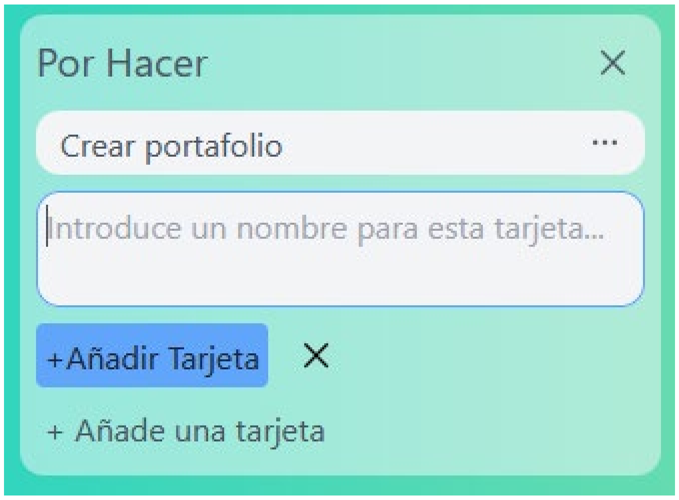
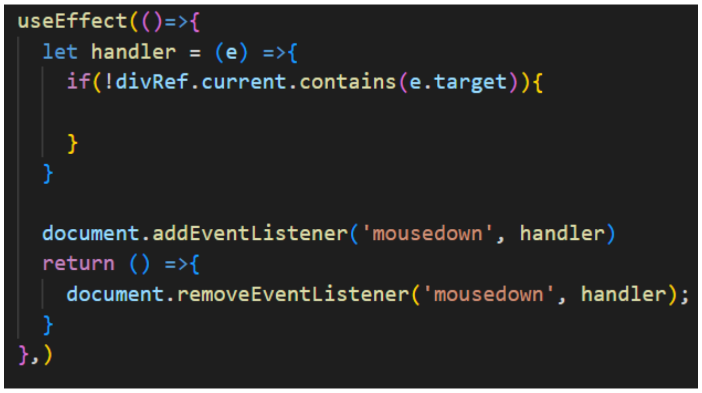


Figure 13. *Input with autofocus*

It was identified that having multiple active forms at the same time was inconvenient for the user, so a function called "onClickOut" was implemented. This function automatically closes the form if it detects a click outside the div that contains it.



```
useEffect(()=>{
  let handler = (e) =>{
    if(!divRef.current.contains(e.target)){

    }
  }

  document.addEventListener('mousedown', handler)
  return () =>{
    document.removeEventListener('mousedown', handler);
  }
},)
```

Figure 14. *"onClickOut" Function*

A stable point of basic functionality for the Kanban app was reached, and Quality of Life (QOL) improvements were implemented.

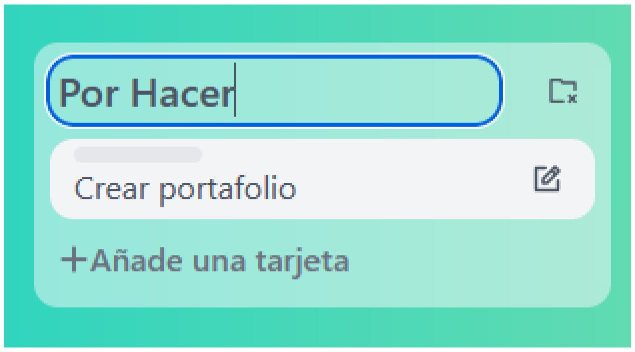


Figure 15. *Title Editor onClick*

The function to edit the container's title was implemented. Users can click on the title to edit it, and save the changes by clicking outside the div or pressing Enter.

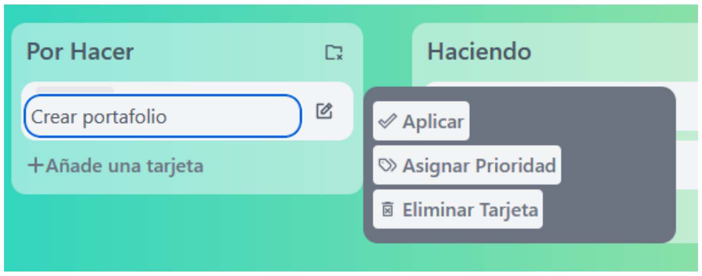


Figure 16. *Edit Task Button and Form*

A button was added to each card that allows the user three different functions:

1. Edit the content of the card
2. Delete the card
3. Assign a priority to the task
4. Apply the changes

The functionality was also added to allow the user to save changes to the card by clicking outside the div or pressing Enter.

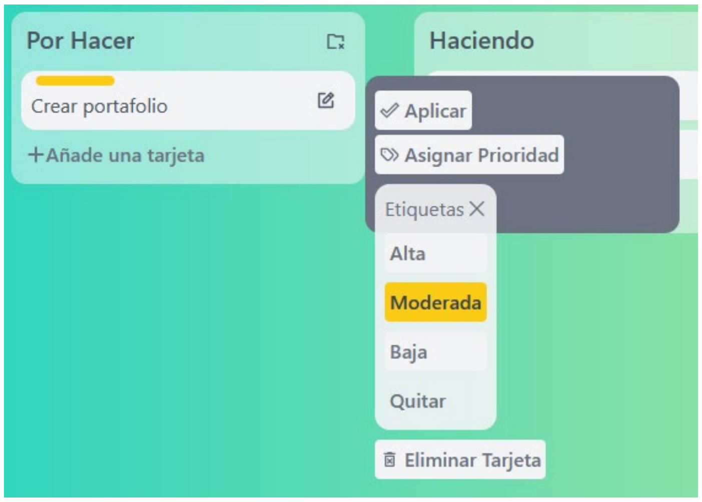


Figure 17. *Priority Assignment Form*

An intuitive priority system was implemented, allowing users to assign priority to tasks based on the color they choose for the label, indicating how "urgent or important" the task is.

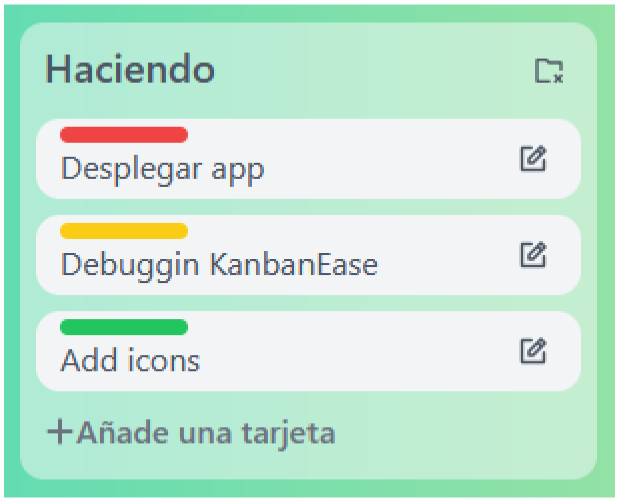


Figure 18. *Priority labels functioning*

At this point, it was decided to make the first deployment of the app on GitHub Pages. For this, the app was named KanbanEase, and an aesthetic logo was designed using Adobe Illustrator. [9]

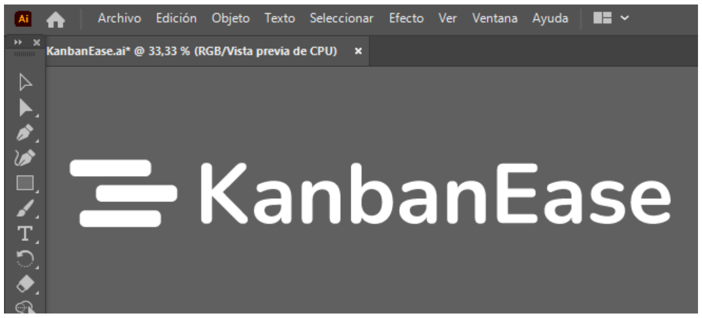


Figure 19. *Design of the KanbanEase logo in Illustrator*

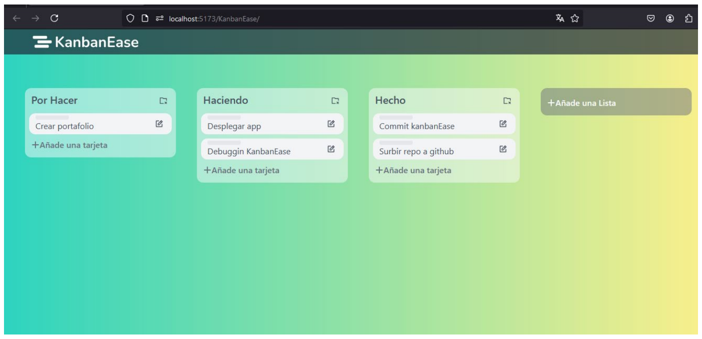


Figure 20. *App in its first deployment versión*

Visual improvements were made to the app, concluding the first version of the KanbanEase app.

4. Results and Analysis

Results:

- Successful implementation of drag and drop functionality on both the X and Y axes.
- Dynamic forms for adding and editing tasks, with autofocus and auto-close features.
- Accessible UI design using Tailwind CSS.
- Intuitive design enhanced user experience, making the app easy to use for both novice and experienced users.

Analysis:

- After resolving initial issues with duplicate indices, the application operates stably without significant errors.
- Compared to other Kanban tools like Trello, KanbanEase provides a straightforward, no-login user experience focused specifically on task organization, without additional elements that could confuse or distract the user.

5. Conclusions

This project provided an opportunity to solidify key knowledge in various areas, including development environment setup with Vite, state management and functionality using React hooks, styling implementation with Tailwind CSS, and the use of external libraries such as React Icons and @hello-pangea/dnd.

Future versions are planned to include code refactoring, utilizing useContext to avoid prop drilling and improve project maintainability. Additionally, integration with the Local Storage API will allow users to retain their tasks even when refreshing the page or closing the browser.

New features are also considered, such as options for importing and exporting tasks to facilitate backup and sharing of work organization. Moreover, a choice between light and dark themes will be added, providing a more personalized user experience.

6. References

1. Ono, T., & Team. (1940s). Kanban: A Visual System for Managing Production. Toyota.

2. Taiichi Ohno. (1988). Toyota Production System: Beyond Large-Scale Production. Productivity Press.

3. Womack, J.P., & Jones, D.T. (1996). Lean Thinking: Banish Waste and Create Wealth in Your Corporation. Simon & Schuster.

4. React Team. (2024). React Documentation. Retrieved from https://reactjs.org/docs/getting-started.html.

5. Tailwind CSS Team. (2024). Tailwind CSS Documentation. Retrieved from https://tailwindcss.com/docs

6. Vite Team. (2024). Vite Documentation. Retrieved from https://vitejs.dev/

7. @hello-pangea/dnd Team. (2024). React DnD Documentation. Retrieved from https://github.com/hello-pangea/dnd

8. React Icons Team. (2024). React Icons Documentation. Retrieved from https://react-icons.github.io/react-icons/

9. Adobe Systems Incorporated. (2024). Adobe Illustrator Documentation. Retrieved from https://helpx.adobe.com/illustrator.html