

## Kanban Local-Storage.md

### • Reading from Local Storage

```
const ticketsFromLS = JSON.parse(localStorage.getItem("myTickets"));  
let ticketsArr = ticketsFromLS;
```

String → array  
if present

// [];  
if nothing exist  
return null

### • Loading Stored Tickets into UI

```
function init() {  
  // import stored tickets into UI  
  ticketsArr.forEach(function(ticket) {  
    // loops through each ticket  
    createTicket(ticket.ticketTask, ticket.ticketId, ticket.ticketColor);  
    // recreates UI  
  });  
  init();  
}
```

init() → calls init() immediately when page loads

### • Saving a newly Added Ticket

```
ticketsArr.push({  
  ticketTask: task,  
  ticketId: id,  
  ticketColor: color,  
});  
localStorage.setItem("myTickets", JSON.stringify(ticketsArr));
```

→ pushes a new ticket into ticketsArr.

writes updated task

Array → string  
because LS stores only string

- 
- Kanban Q.11 → Add a new Ticket to ticketsDB in local storage
- Kanban Q.13 → Create ticket UI from ticketsDB in local storage

Kanban  
Q. 12

Deleting a ticket from ticketsDB in local storage.

Sol.

```
let ticketArr = JSON.parse(localStorage.getItem("ticketsDB"));
let ticketID = e.currentTarget.querySelector(".ticket-id").innerHTML;

function getTicketIdx(id) {
  for (let i = 0; i < ticketArr.length; i++) {
    if (ticketArr[i].ticketId === id) {
      return i;
    }
  }
  return -1; // if no ticket matches
}

let idx = getTicketIdx(ticketID);
if (idx !== -1) {
  ticketArr.splice(idx, 1); // remove item and rebuilds array
}
localStorage.setItem("ticketsDB", JSON.stringify(ticketArr));
};
```

if id matches,  
we return the index  
where it found

remove item  
and rebuilds  
array

## # Kanban Summary

- Kanban ex. 4: Viewing Input Modal on (+).

```
let modalFlag = false
```

```
{ let  
  let DOM (selection)  
  let
```

```
addBtn.addEventListener('click', function() {  
  if (modalFlag == false) {  
    modalCont.style.display = 'flex';  
    modalFlag = true;  
  }  
});
```

- Kanban ex. 5: Automat Ticket Creation from input.

```
modalCont.addEventListener('keydown', function(e) {  
  let key = e.key
```

```
  if (key == "shift") {
```

```
    createTicket(TaskArea.value, modalPriorityColor);  
    modalCont.style.display = "none";  
    addFlag = false;
```

← clears  
task area  
before closing

```
    TaskArea.value = "";
```

```
  }  
});
```

value → prewritten  
value

value  
↳ input  
value

// adding tickets to DOM

```
function createTicket(ticketTask, ticketColor) {  
  ticketId++  
  let ticketCont = document.createElement("div");  
  ticketCont.setAttribute("class", "ticket-cont");  
  ticketCont.innerHTML =
```

```
<div class="ticket-color ${ticketColor}"> </div>
```

```
<div class="ticket-id"> # ${ticketID} </div>
```

```
<div class="task-area"> ${ticketTask} </div>
```

```
{  
{
```

```
mainCont.appendChild(ticketCont);
```

```
}
```

input  
values

Kanban

## Q.6 Filter with Colors

```
let filterButtons = document.querySelectorAll('.color');
```

```
filterButtons.forEach(function(btn) {  
  btn.addEventListener('click', function() {  
    let selectedColor = btn.classList[0];
```

// 1. Remove all existing tickets from DOM

```
let allTickets = document.querySelectorAll('.ticket-content');
```

```
allTickets.forEach(function(t) {  
  t.remove();  
});
```

~~// 2. Filter ticketsArr~~  
~~return ticketsArr.filter(function(ticketObj) {~~  
 ~~return ticketObj.ticketColor === selectedColor~~

// 2. filter ticketsArr

```
let filteredTickets = ticketsArr.filter(function(ticketObj) {  
  return ticketObj.ticketColor === selectedColor;  
});
```

// 3. Add filtered tickets back to DOM

```
filteredTickets.forEach(function(ticketObj) {  
  createTicket(ticketObj.ticketText, ticketObj.ticketColor,  
    ticketObj.ticketID);  
});
```

```
});
```

```
});
```

## ○ Kanban Q.7 : Locking Mechanism

```

let ticket ContArr = document.querySelectorAll('!ticket-cont')
for (let i = 0; i < ticketContArr.length; i++) {
  ticketContArr[i].querySelector('!button').addEventListener(
    'click', function(e) {

      let ticket = ticketContArr[i];
      let buttonDiv buttonDiv = ticketContArr[i].children[3];
      let button = buttonDiv.querySelector('!button');

      if (buttonDiv.classList.contains('ticket-locked')) {
        button.innerText = 'Unlocked';
        buttonDiv.classList.remove('ticket-locked');
        buttonDiv.classList.add('ticket-unlock');
        ticketTaskArea.setAttribute("ContentEditable", "true");
      } else {
        button.innerText = 'locked';
        buttonDiv.classList.remove('ticket-unlock');
        buttonDiv.classList.add('ticket-locked');
        ticketTaskArea.setAttribute("ContentEditable", "false");
      }
    }
  )
}

```

select third div

ContentEditable

## ○ Kanban Q.8 Delete Button

```

let delCont = document.querySelector('!remove-btn');
let AllTickets = document.querySelectorAll('!ticket-cont');
let deleteMode = false; // toggle flag

delCont.addEventListener('click', function() {
  deleteMode = !deleteMode; // toggle

  if (deleteMode) {
    alert("activated");
    delCont.style.backgroundColor = "red";
    AllTickets.forEach(function(ticket) {
      ticket.addEventListener('click', deleteTicket);
    });
  }
});

```

```

else {
  alert ("de-activated");
  del Cont.style.backgroundColor = "inherit";

  All Tickets. For Each (function(ticket) {
    ticket.removeEventListener('click', deleteTicket);
  });
}

// Function To delete Ticket
function deleteTicket() {
  if (deleteMode) {
    this.remove();
  }
}

```

remove event Target task

### // Kanban Q.9 : Clicking and Changing Ticket Colors

```

let ticket = document.querySelector(".ticket-cont");
let colors = ["lightpink", "lightgreen", "lightblue", "black"];
handleColor(ticket);

function handleColor(ticket) {
  let ticketColorBand = ticket.querySelector(".ticket-color");
  ticketColorBand.addEventListener("click", function() {

    let currentColor = ticketColorBand.classList[0];
    let currentIndex = colors.indexOf(currentColor);
    let nextColorIndex = (currentIndex + 1) % colors.length;
    let nextColor = colors[nextColorIndex];
    ticketColorBand.classList.remove(currentColor);
    ticketColorBand.classList.add(nextColor);
  });
}

```