Video 7

Autocomplete / Type ahead

FrontEnd Machine Coding Mock Interview

Statement

Autocomplete Low-level Design.

Tech stack

Vanilla JS, HTML & CSS

Must have

→ On typing inside the search box, it should suggest option.

→ Users should be able to select one of the option

Good to have :-

→ Search should be performant enough.

→ It should avoid unnecessary network call.

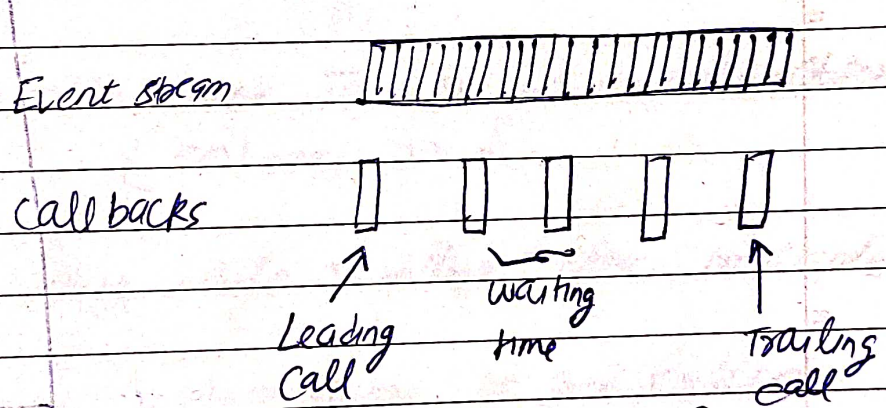
→ It should be re-usable & customizable

→ It should persist previously fetch data

Debounce

```
const debounce = (func, delay) => {
  let inDebounce;
  return function () {
    const context = this;
    const args = arguments;
    clearTimeout(inDebance);
    inDebance = setTimeout(() => func.
inDebounce = setTimeout(() => func.apply(context, args),
                        delay);
```

Throttle

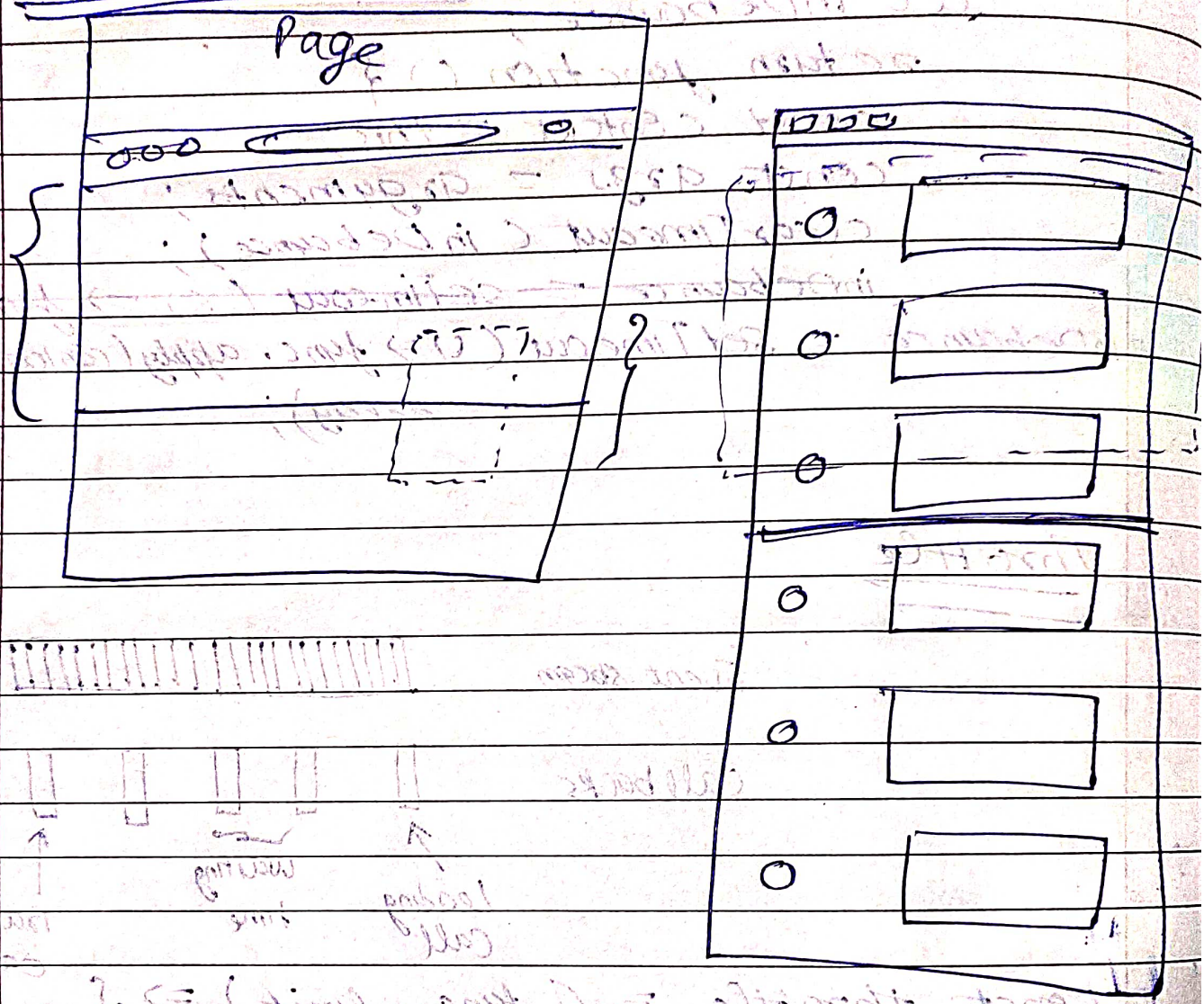



```
const throttle = (func, limit) => {
  let lastFunc;
  let lastRan;
  return function () {
    const context = this;
    const args = arguments;
    if (!lastRun) {
      func.apply(context, args);
      lastRan = Date.now();
    } else { clearTimeout(lastFunc);
      lastFunc = setTimeout(function () {
        if ((Date.now() - lastRan) >= limit) {
          func.apply(context, args)
          lastRan = Date.now() }
```

}, limit - (Date.now() - lastRan)

}

Intersection observer



JavaScript Snippet

```
var scroller = document.querySelector('#scroller');
var sentinel = document.querySelector('#sentinel');
var counter = 1;

function loadItems(n) {
    for (var i = 0; i < n; i++) {
        var newItem = document.createElement('div');
        newItem.classList.add('item');
        newItem.textContent = 'Item' + counter++;
        scroller.appendChild(newItem);
    }
}

var intersectionObserver = new IntersectionObserver(
    entries => {

// If the browser is busy while scrolling happen, multiple entries can
// accumulate before invocation of this callback.
// As long as any one
// of the notification for the sentinel within the
// scrolling viewport,
// we add more content.

    if (entries.some(entry => entry.intersectionRatio > 0)) {
        loadItems(10);
// appendChild will move the existing element, so there
// is no need to
// remove it first.
        scroller.appendChild(sentinel);
    }
    loadItems(5);
    chromeSample.setStatus('loaded up to item' + counter);
```

});

~~intersectionObserver.observe(sentinel);~~

Abort Controller

```
const controller = new AbortController();
const res = fetch('/', { signal: controller.signal });
controller.abort();
console.log(res);
```