#ref #incomplete

---

1 | **Web Development!**

1.1 | **cool websites**

#todo add a note here, and populate!

1.2 | **front end resources**

- of course, react
  - and a great component library for it
- tailwind!!!
  - remember to follow the installation guide specific to your stack:
    - normal
    - cdn
    - react
    - next
  - the v helpful keybind-driven docs
- nextjs
- threejs
  - and react-three-fiber KBxR3F
- keycode.info
  - for getting keycodes and such efficiently!
- if you need cross-platform, check out
  - ionic
  - and capacitor!
- website for free graphics

1.3 | **back end**

- supabase
  - easiest database manager i have ever seen. genuinely, get a db up and running in just a few minutes w/ this
    - free, easy, auth, v little boilerplate, and can be realtime if need be
- also, of course, firebase

- else, mongodb is very nice. checkout the MERN stack
- redis can really help with improving speed and reducing database hits, as well as making data easier to work w/
  - at the cost of more mem usage…

1.4 | **new discoveries, tips, and tricks**

---

x, wrapped by lines. n *****

nextjs is so #cool. w/ `_app.js`, you can easily wrap every component, thus providing super easy global state handling w/ a react component and logic!

---

tailwind can have custom args like `mt-[22px]`!

---

react's `useReducer` is so cool. docs

```
function reducer(state, action) {
  switch (action.type) { // switch statments with logic, that act on objects passed in!
    case 'increment':
      return {count: state.count + 1};
    case 'decrement':
      return {count: state.count - 1};
    default:
      throw new Error();
  }
}
function Counter() {
  const [state, dispatch] = useReducer(reducer, initialState);
  return (
    <>
      Count: {state.count}
      <button onClick={() => dispatch({type: 'decrement'})}>-</button>
      <button onClick={() => dispatch({type: 'increment'})}>+</button>
    </>
  );
}
```
