Animated Responsive Menu – README

This README explains how to edit, customize, and extend the animated, responsive navigation you got (top desktop bar + mobile bottom bar with a sliding pill indicator).

What's included

```
index.html   → Markup for desktop top-nav + mobile bottom-nav
style.css    → Styling, themes, layout, and responsive behavior
script.js    → Indicator positioning + activation logic
```

How it works (quick tour)

- **Two navs** share the same items: a **top nav** (desktop ≥ 900px) and a **bottom nav** (mobile < 900px). Only one is visible at a time via CSS media queries.
- Each item has a `data-route` like `home`, `search`, etc. The script uses this to set active state and optional **hash-based deep-linking** (e.g. `#messages`).
- A floating `.indicator` pill is absolutely positioned under the active item. JS measures the target item and sets `left`, `top`, `width`, and `height` to slide the pill smoothly.
- Core customization is done via **CSS variables** in `:root` (colors, radii, timing) and via **media queries**.

Edit menu items (labels, links, icons)

Desktop (top nav)

Edit the list in `index.html`:

```
<nav class="top-nav" aria-label="Primary">
  <ul class="menu" id="topMenu">
    <li class="menu-item is-active"><a href="#home" data-route="home">Home</
a></li>
    <li class="menu-item"><a href="#search" data-route="search">Search</a></
li>
    <!-- add/remove items here -->
    <span class="indicator" aria-hidden="true"></span>
  </ul>
</nav>
```

Mobile (bottom nav)

Each item is a `<button>` with optional inline SVG. Keep `data-route` values **identical** to desktop items so activation stays in sync:

```
<li class="menu-item" role="tab" aria-selected="false">
  <button type="button" data-route="favorites" aria-label="Favorites">
    <!-- replace or edit the SVG icon -->
    <svg viewBox="0 0 24 24" class="icon" aria-hidden="true">…</svg>
    <span class="label">Favorites</span>
  </button>
</li>
```

> **Tip:** If you add a new item, add it to **both** menus and keep the same order for a consistent feel.

Changing routes (URLs)

- By default, clicking updates the URL hash (`#home`, `#search`, ...) without reloading the page.
- To navigate to real pages instead, change the `href` values to real URLs and **optionally** remove the line in `script.js` that updates the hash (see **Disable deep-linking** below).

Styling & theme

Open `style.css` and tweak the variables in `:root`:

```
:root{
  --bg: #0f1221;
  --card: #171a2e;
  --text: #e8e8ef;
  --muted: #a5a6b3;
  --accent: #6c7cff;      /* pill + focus color */
  --ring: rgba(108,124,255,.35); /* pill glow */
  --shadow: 0 10px 30px rgba(0,0,0,.35);
  --radius: 14px;         /* menu container radius */
  --transition: 300ms cubic-bezier(.2,.7,.2,1);
}
```

Indicator look

```
.menu .indicator{ box-shadow: 0 12px 36px var(--ring); background: var(--
accent); }
```

- Make the pill more round/flat by changing **border-radius** in layout-specific rules. - Speed/feel is controlled by `--transition`.

Desktop width, centering, padding

Inside the desktop media query (`@media (min-width: 900px)`):

```
.top-nav .menu{
  width: clamp(50vw, 72vw, 1100px); /* min 50% of viewport, up to 1100px */
  margin-inline: auto;              /* center the whole bar */
  padding: 10px 18px;               /* inner left/right padding */
}
```

- Increase the middle value (`72vw`) for a wider bar by default. - Raise the max (`1100px`) if you want it to grow on very large screens.

Mobile spacing

- Icons + labels are centered by `flex` rules on the buttons:

```
.bottom-nav .menu .menu-item button{ flex-direction: column; align-items: center; justify-content: center; }
```

Animation & indicator behavior

The indicator is sized/positioned in `script.js`:

```
function positionIndicator(menuEl, indicator, itemEl){
  const itemRect = itemEl.getBoundingClientRect();
  const menuRect = menuEl.getBoundingClientRect();
  const isTop = menuEl.id === 'topMenu';
  const pad = 8; // vertical inset
  const rawWidth = itemRect.width;
  const width = Math.min(Math.max(rawWidth - (isTop ? 12 : 10), isTop ? 90 : 56), 180);
  const height = Math.max(menuRect.height - pad * 2, 30);
  const left = Math.round(itemEl.offsetLeft + itemEl.offsetWidth / 2 - width / 2);
  const top = pad;
  Object.assign(indicator.style, { width: `${width}px`, height: `${height}px`, left: `${left}px`, top: `${top}px` });
}
```

Things you can tweak: - **Minimum/maximum width** of the pill (`isTop ? 90 : 56`, `180`). - **Vertical padding** (`pad`) to shift the pill up/down inside the track. - **Transition timing** in CSS (`--transition`).

Add / remove items

- Add/remove `<li class="menu-item">…</li>` blocks.
- Keep a matching `data-route` and text/icon pair in both menus.

- JS automatically finds the first item as active if none is marked `is-active`.

Accessibility

- **Keyboard**: Press **Enter/Space** on focused item to activate.
- **Focus ring** is visible; adjust in CSS under `.menu a:focus-visible, .menu button:focus-visible`.
- Mobile tabs use `role="tab"` and update `aria-selected`.
- Keep button `aria-label` in mobile for icon-only understanding.

Disable or change deep-linking

- To disable hash updates, remove this block in `script.js` inside `activateItem`:

```
const route = itemEl.querySelector('[data-route]')?.dataset.route;
if (route) history.replaceState(null, '', `#${route}`);
```

- To default to a specific item on load, set `is-active` on that item in **both** menus.

Change the responsive breakpoint

- The switch between mobile/desktop happens at `@media (min-width: 900px)`.
- Raise/lower `900px` depending on your design.

Common tweaks

1) Bigger pill on desktop

```
@media (min-width: 900px){ .top-nav .menu .indicator{ height: calc(100% - 12px); top: 6px; } }
```

2) Snappier animation

```
:root{ --transition: 200ms cubic-bezier(.3,.8,.2,1); }
```

3) More contrast on inactive labels

```
.menu a, .menu button{ color: #768; }
.menu-item.is-active a, .menu-item.is-active button{ color: #fff; }
```

4) Different icon set - Replace the inline `<svg>` with your own (24×24 viewBox recommended). Keep the class `icon` so size inherits.

5) Make the bar sticky on desktop

```
.site-header{ position: sticky; top: 0; }
```

Running locally

1. Put the three files in the same folder.
2. Open `index.html` directly in a browser, or serve with a simple static server.
3. Python: `python -m http.server 8080`
4. Node (serve): `npx serve` or `npx http-server`

Troubleshooting

- **Pill looks misaligned**: Ensure the `.menu` container has the same vertical padding used in JS (`pad = 8`) and isn't clipped (`overflow: visible`).
- **Active item not highlighted**: Make sure exactly one `.menu-item` has `is-active` at load, or let JS pick the first.
- **Items wrap on desktop**: Increase `.top-nav .menu` width or reduce link padding.

Extending

- Convert the menu into a reusable component (React/Vue) by mapping over an array of `{ route, label, icon }` items.
- Add a badge (e.g., unread count) with an absolutely positioned small circle inside the button.
- Hook into your router: on route change, call `activateItem(menuEl, relatedMenuItem)` to move the indicator.

Enjoy customizing! If you want, I can tailor a version for your design system (colors, spacing scale, typeface) or convert this to a React component with props for items and active route.