State Machines as Specification

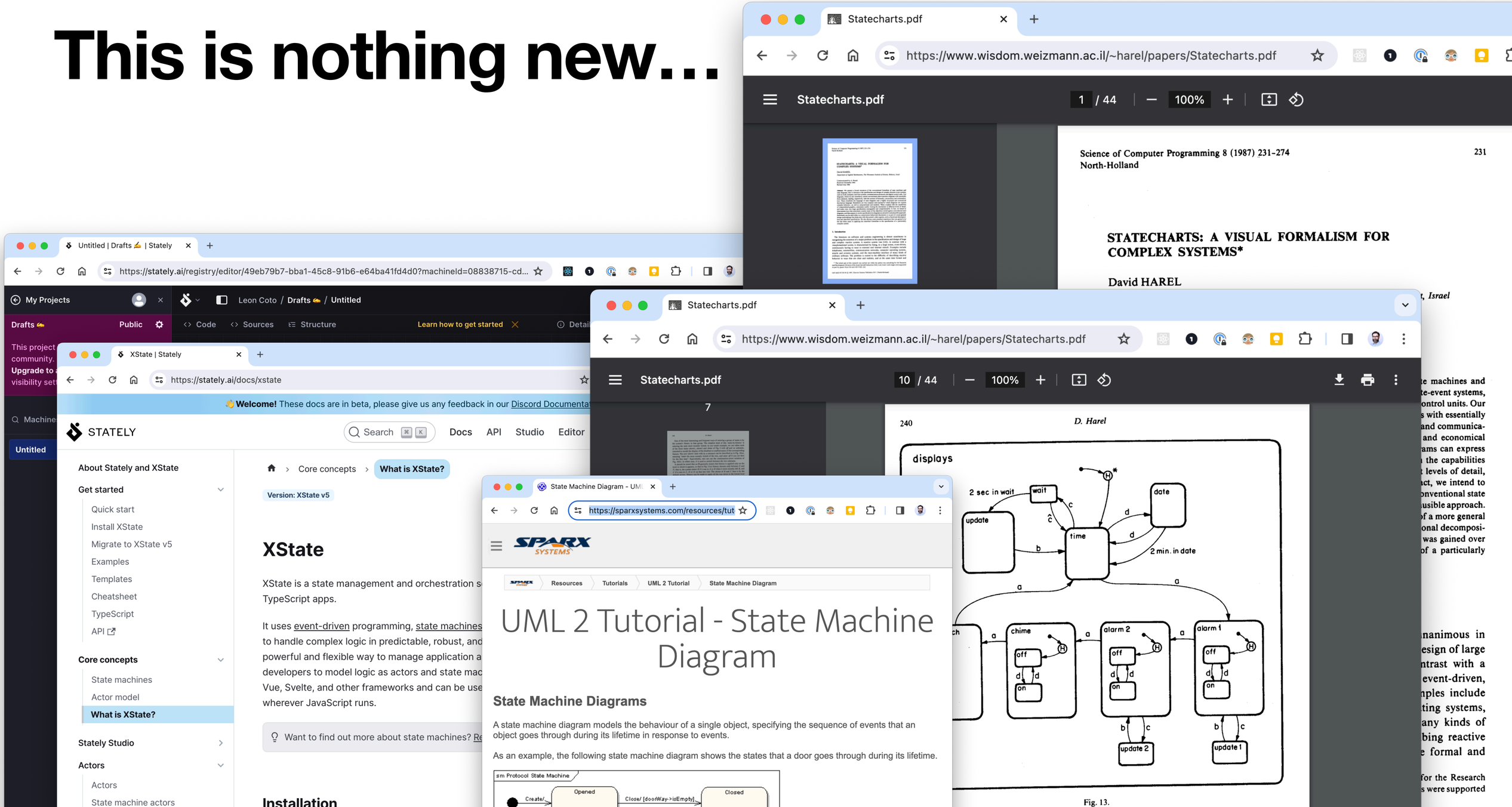
Modelling State with Mermaid and React-RxJS

Leon Coto, March 2024

The Premise

- UI Applications can be represented as state machines.
- A state machine is a state type plus an interpreter for state transitions.
- RxJS and React-RxJS are well suited to implement state machines specified with Mermaid.

This is nothing new…



A Mermaid Primer

https://mermaid.js.org/

A state diagram is a type of diagram used in computer science and related fields to describe the behavior of systems. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction.

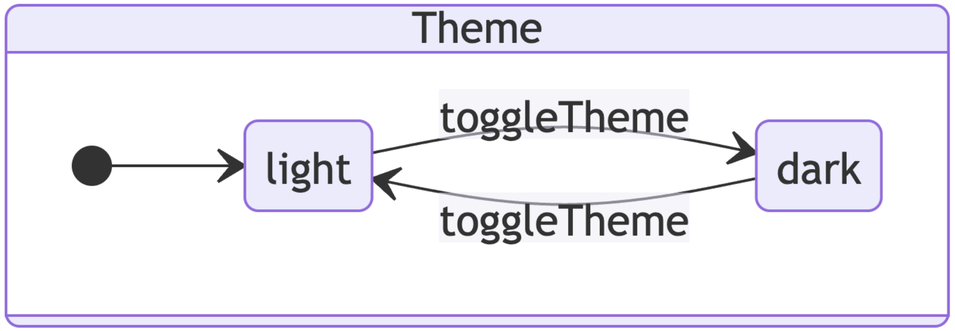
- VSCode Integration https://marketplace.visualstudio.com/items?itemName=bierner.markdown-mermaid
- Github Integration https://github.blog/2022-02-14-include-diagrams-markdown-files-mermaid/

Mermaid Demo

?

Writing a Specification

The State Diagram

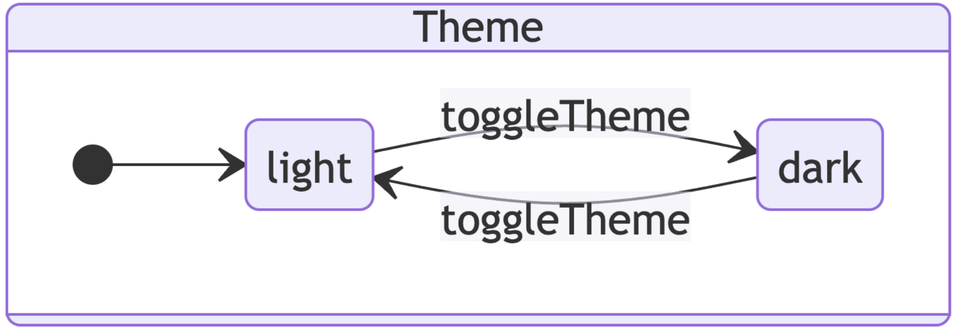


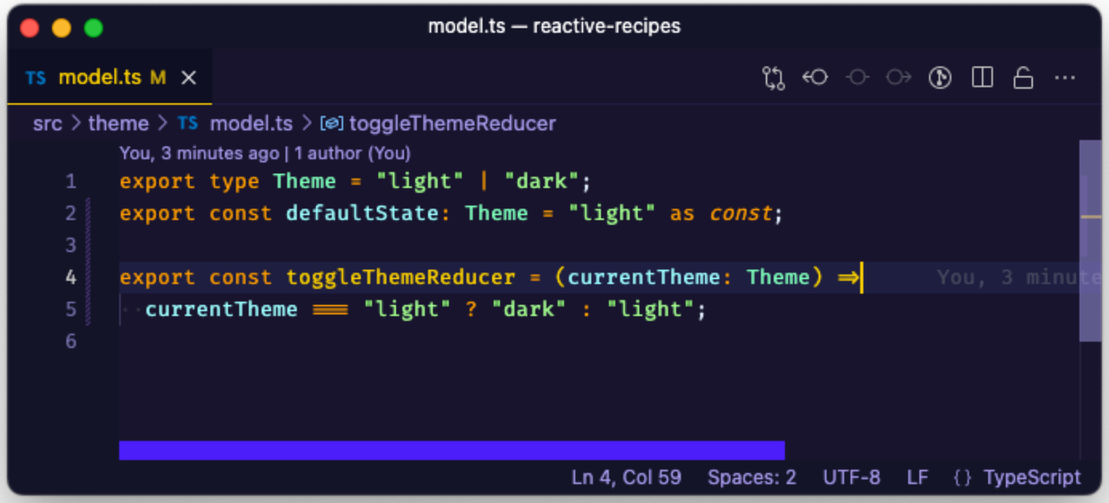
- The state name is `Theme`.
- Theme is composed by two sub-states: `light` and `dark`.
- The default sub-state is `light`.
- There is a single state transition named `toggleTheme`.
- There is no terminal state.

Deriving a Model

The State Type

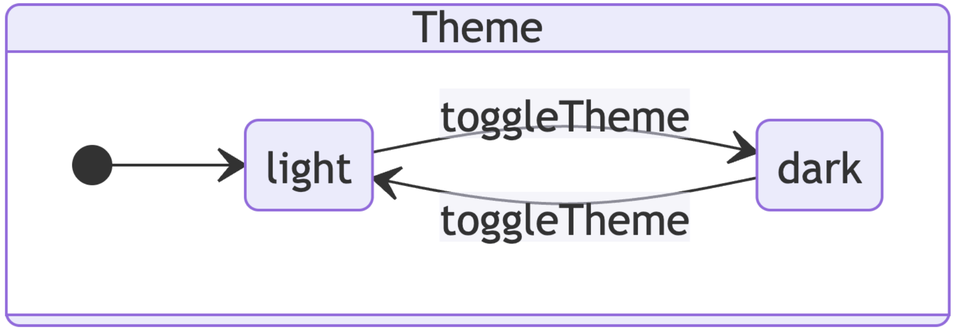
- The name of the state becomes the type of our state in our domain.
- The value of the Theme type becomes an union type of all the possible states.
- The default state is a member of the Theme type.





Writing an Interpreter

The State Machine



- The state machine is an `Observable<Theme>`, that interprets the state transitions in Theme.
- State transitions are represented with React RxJS signals*
- The interpretation of the signals is private to our state machine. That is, handling emissions from `toggleTheme$`.
- The public API of our state machine is the triggers for state transition (`toggleTheme`), and the interpreted state (`useTheme` and `theme$`)

state.ts — reactive-recipes

TS state.ts M

src > theme > TS state.ts > ...

You, 1 second ago | 1 author (You)

```
import { bind } from "@react-rxjs/core";
import { createSignal } from "@react-rxjs/utils";
import { scan, startWith } from "rxjs";
import { defaultState } from "./model";

const [toggleTheme$, toggleTheme] = createSignal();

const [useTheme, theme$] = bind(
  toggleTheme$.pipe(
    scan(
      (currentTheme) => (currentTheme === "light" ? "dark" : "light"),
      defaultState
    ),
    startWith(defaultState)
  )
);

export { toggleTheme, useTheme, theme$ };
```

Writing an Interpreter

Refactoring the Interpreter

- It should be possible to extract state transition functions as pure functions.
- Pure functions can be unit tested trivially.
- The state layer clearly deals with handling side effects and composing signals only.

state.ts — reactive-recipes

model.ts M

src > theme > model.ts > toggleThemeReducer

You, 48 seconds ago | 1 author (You)

```typescript
export type Theme = "light" | "dark";
export const defaultState: Theme = "light" as const;

export const toggleThemeReducer = (currentTheme: Theme) =>
  currentTheme === "light" ? "dark" : "light";
```

state.ts M

src > theme > state.ts > ...

You, 2 minutes ago | 1 author (You)

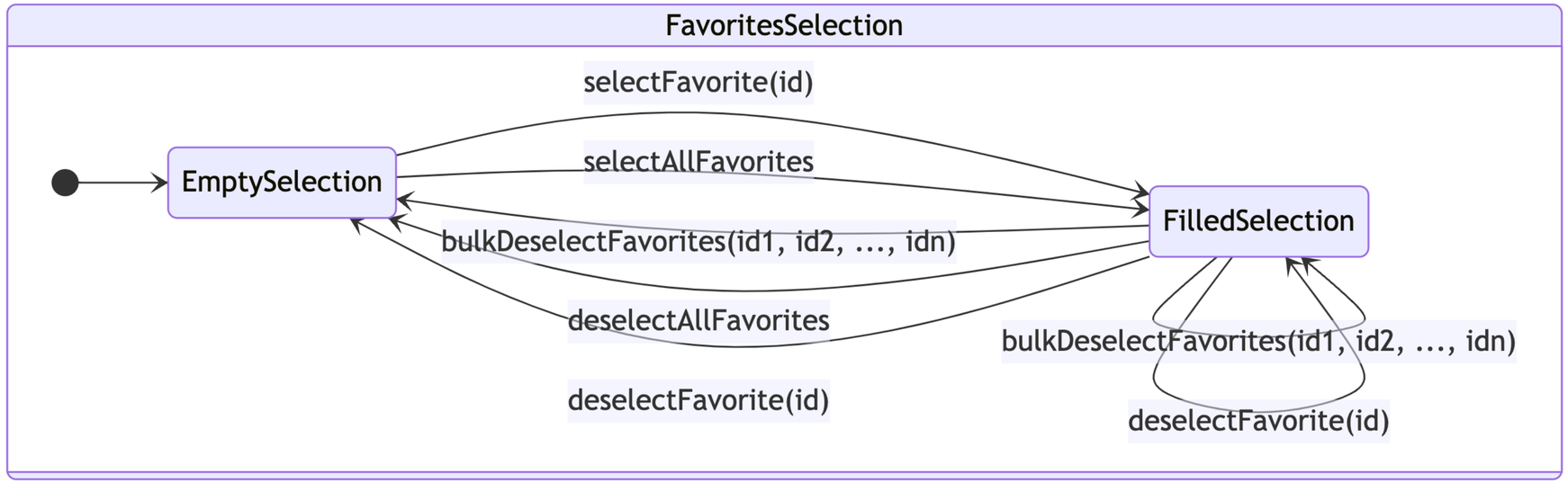
```typescript
import { bind } from "@react-rxjs/core";
import { createSignal } from "@react-rxjs/utils";
import { scan, startWith } from "rxjs";
import { defaultState, toggleThemeReducer } from "./model";

const [toggleTheme$, toggleTheme] = createSignal();

const [useTheme, theme$] = bind(
  toggleTheme$.pipe(
    scan(toggleThemeReducer, defaultState),
    startWith(defaultState)
  )
);

export { theme$, toggleTheme, useTheme };
```

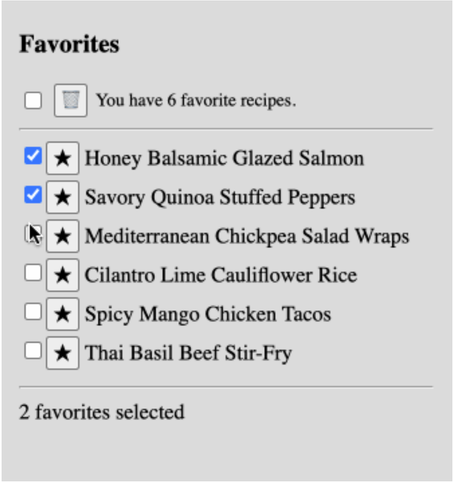
main* 0 0 0 Live Share Select Postgres Server TypeScript Quokka Spell Prettier

?

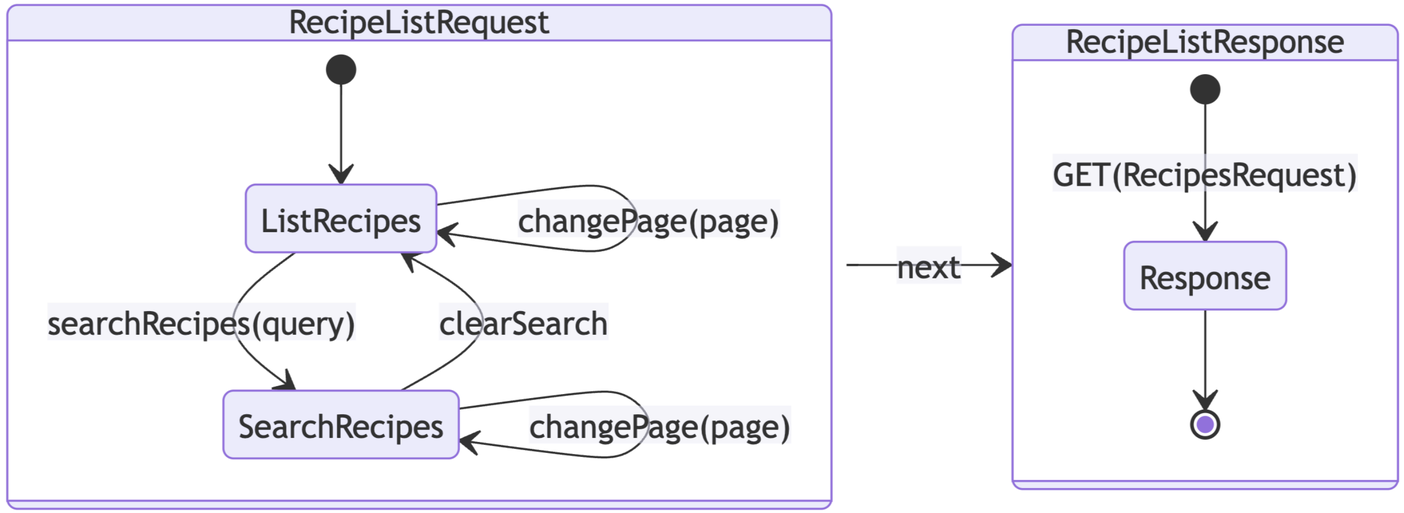
Live Demo

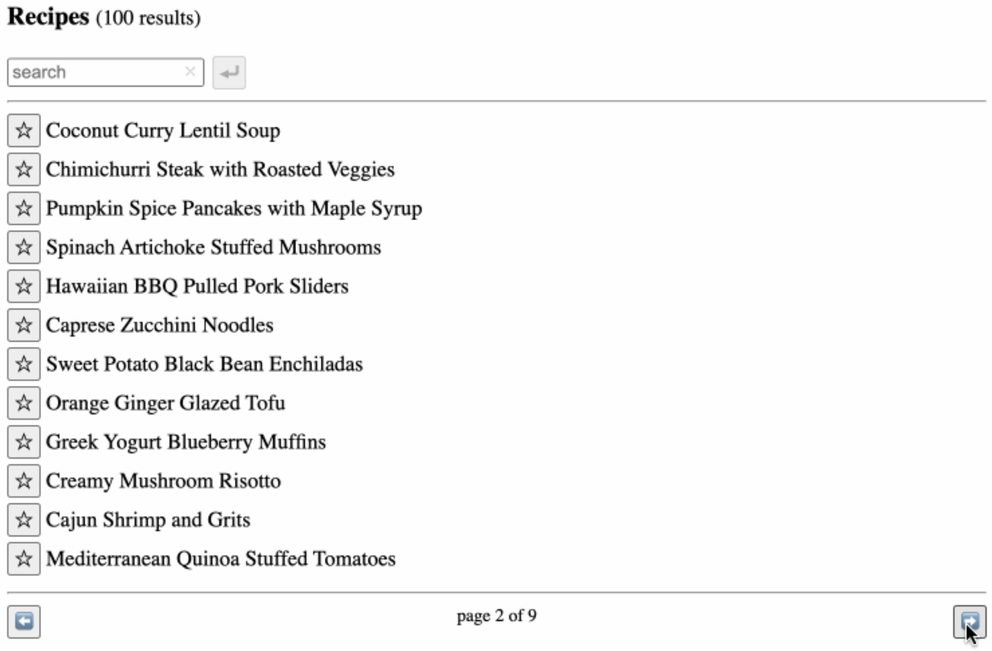
Managing Favorites





Parameterising Services





References

- Example implementation [WIP] https://github.com/grancalavera/reactive-recipes
- *Domain Modeling Made Functional* https://pragprog.com/titles/swdddf/domain-modeling-made-functional/
- *Statecharts: a visual formalism for complex systems* https://www.sciencedirect.com/science/article/pii/0167642387900359
- Mermaid website https://mermaid.js.org/
- Mermaid VSCode Integration https://marketplace.visualstudio.com/items?itemName=bierner.markdown-mermaid
- Github Mermaid Support https://github.blog/2022-02-14-include-diagrams-markdown-files-mermaid/