Framework Handson

Building Components

project name: proj-basics

In this we will start from the template project – rename it to proj-basics, build several more components in the cmps folder and use them in the home page.

Challenge #1 - animal-list component

This component receives a prop: animalInfos and render a list of animals and their count, use an HTML <table>:

Data model:

```
animalInfos = [
        {type: 'Malayan Tiger', count: 787},
        {type: 'Mountain Gorilla', count: 212},
        {type: 'Fin Whale', count: 28},
    ]
```

The search link opens a new tab with a search for such animal in google
(e.g.: `https://www.google.com/search?q=Malayan Tiger`)

{coding academy

Challenge #2 – season-clock component

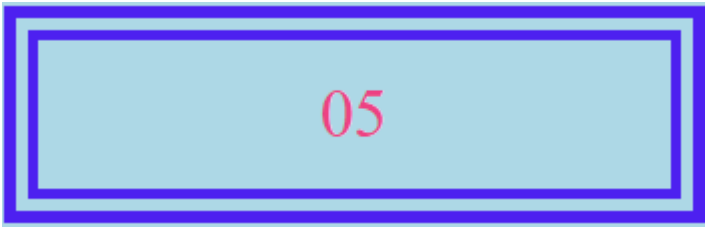
This component shows the current season, month name and day name, when clicking the component, its background should toggle between dark and light colors

Component state: `isDark`

- Show the current season 
- Add a running clock - use an interval to update the state every second, remember to clear the interval when component is unmounting.

{coding academy

Challenge #3 – count-down component



This component has 2 props: *startFrom* and *onDone*.
The component shows a counter going down from startFrom to 0

This is how that component is used (React syntax) for counting 10 seconds:

```
<CountDown startFrom={10} onDone={()=>{
    console.log('Done!')
}} />
```

- During the last 6 seconds, the seconds should make in Red
- When time ends call the onDone function

Bonus

If the component receives a *toTime* property (timestamp), it ignores the *startFrom* prop and renders a timer going down to the *toTime* prop

This is how we can use that component for counting 10 seconds:

```
<CountDown toTime={Date.now() + 1000*10} onDone={()=>{
    console.log('Its Time!')
}} />
```

In the onDone function, animate the clock (use the provided function animateCSS in utilService and a ref)
Bonus: play an audio file (mp3))

{coding academy

Challenge #4 - watcher-app component

Watcher data model:

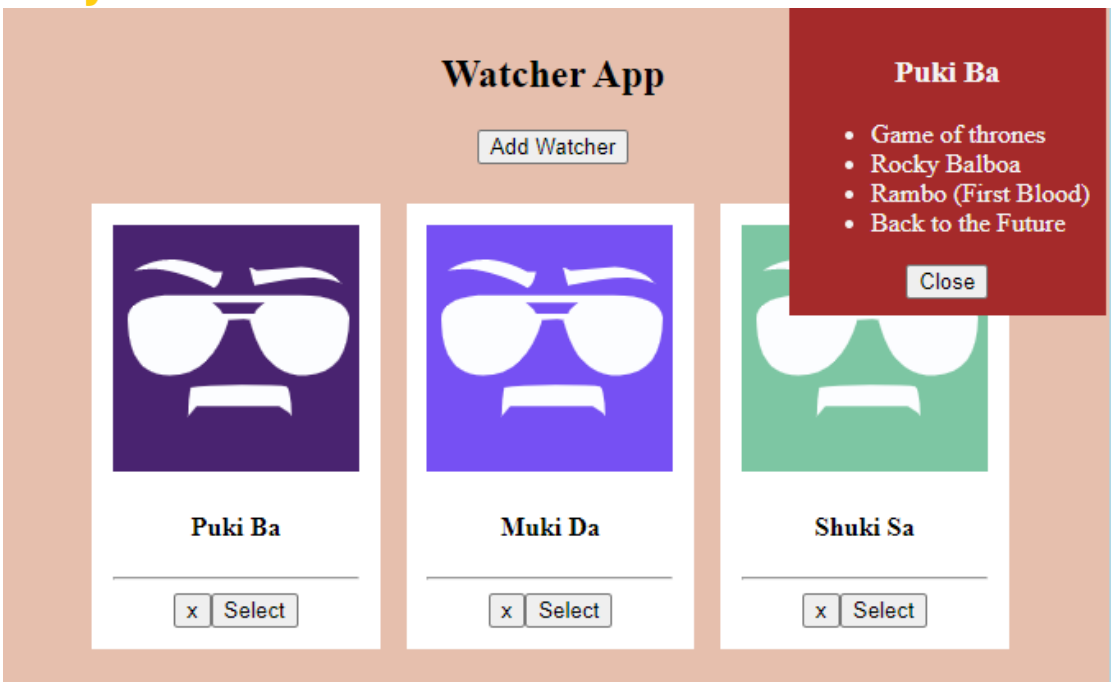
```
{
    id: 'w101',
    fullname : 'Puki Ba',
    movies: ['Rambo', 'Rocky']
}
```

Component state:

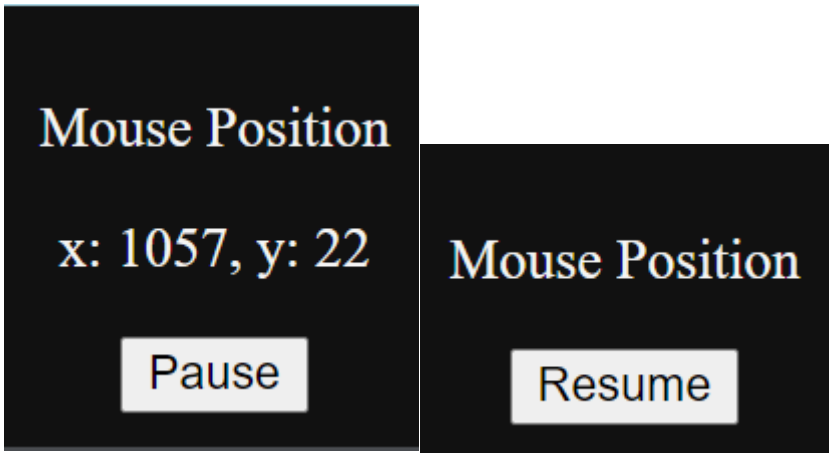
```
watchers
selectedWatcher
```

This component renders a watchers array (retrieved asynchronously from a service) and allows adding, removing and selecting a watcher.
For adding a user, start simple and use the prompt() function
(Bonus: improve to a nice input)

When user is selected – render a modal showing the user name and his movies list, with a close button:



Challenge #5 – mouse-monitor component



This component renders a section at the bottom-right of the page constantly showing the current position of the mouse.

When Pause is clicked, the section no longer monitors the mouse (you will need to remove the event listener)

The initial state of the component:

```
isOn : true,
pos: {
    x: 0,
    y: 0
}
```

TIP

```
addMouseListener(){
   document.addEventListener('mousemove', updatePos)
}
```