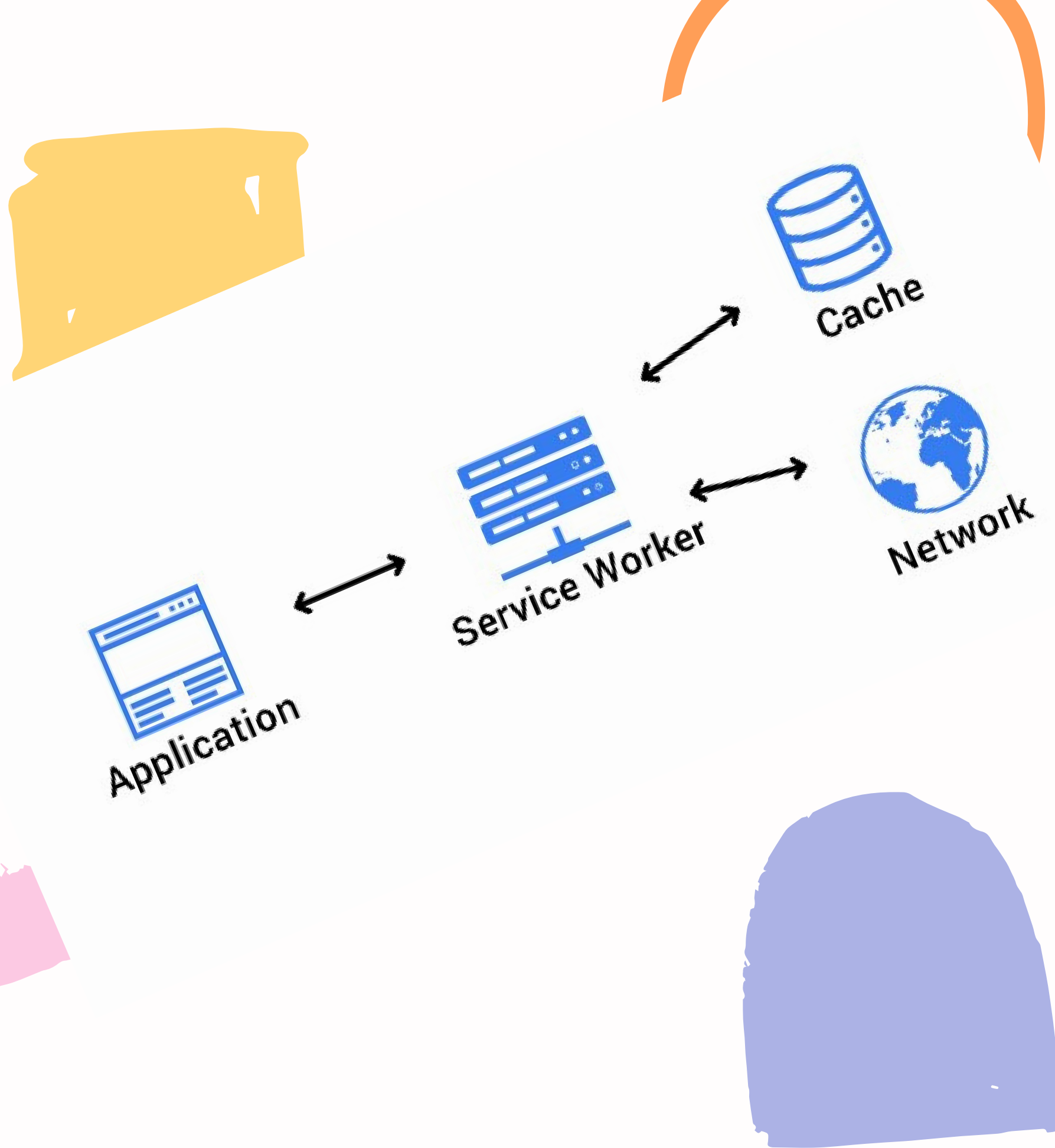


The background features several abstract, hand-painted elements. At the top left, there's a light brown shape. In the top center, a large light blue shape is partially covered by a pink brushstroke. To the right, a yellow shape is decorated with dark green dots. In the center, there's a small orange circle. At the bottom right, a blue shape is partially visible, with a light blue arc overlapping it.

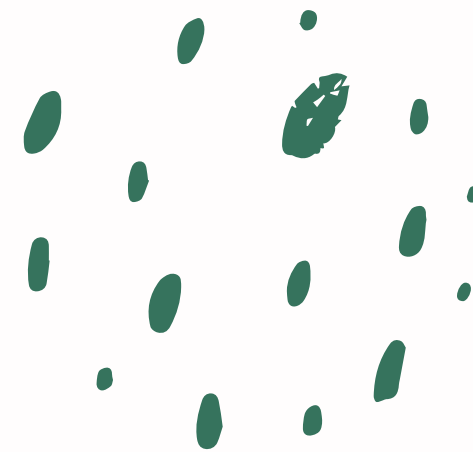
# MSW EXAMPLE

BY: DANIEL DE PAZ



# What is a service worker?

It's essentially a JavaScript file that runs separately from the main browser thread, intercepting network requests, caching or retrieving resources from the cache, and delivering push messages.



# What is MSW?

Is an API mocking library that uses Service Worker API to intercept actual requests.

**Mock Service Worker**  
Seamless API mocking library



# Why mock an API?

- To have standard responses
- Not rely on the backend



```
import {render, screen} from '@testing-library/react'
import userEvent from '@testing-library/user-event'
import {client} from '../utils/api-client'

jest.mock('../utils/api-client')

test('...', () => {
```

```
__tests__/checkout.js

import * as React from 'react'
import {render, screen} from '@testing-library/react'
import userEvent from '@testing-library/user-event'

beforeAll(() => jest.spyOn(window, 'fetch'))

// assuming jest's resetMocks is configured to reset mocks
// we don't need to worry about cleanup
```

# Different ways to do it

- Mock out the client
- We mock out window.fetch
- Create a mock server that intercepts all requests and handle it



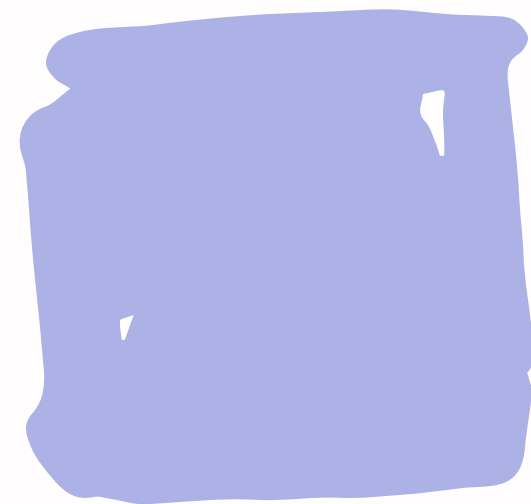
## Mock Service Worker

Seamless API mocking library



# MSW vs MirageJS

Service worker vs replace native XMLHttpRequest and Fetch



EXAMPLE





# Thank you!

Questions? @DanielDePaz on slack