INTRODUCTION TO

REACT JS

WORKSHOP

BY AYUSHYA AMITABH

BY

AYUSHYA AMITABH

PUBLICIS SAPIENT

Full Stack Developer

Front-end Technologies

React • Angular

Back-end Technologies

Java • C++ • Node JS • Graph QL

Data Stores

MongoDB • MySQL • Redis

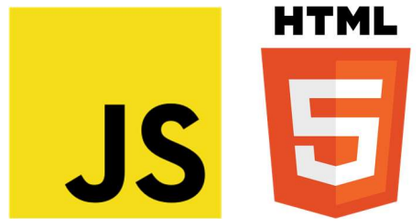
React

What Is React?

React is JavaScript library for building user interfaces.

The main advantage of React is the ability to break down complex UI's into smaller isolated dynamic pieces called "components".

One great side-effect was the creation of "JSX" – a JavaScript syntax extension that allows for HTML syntax within JavaScript code itself.

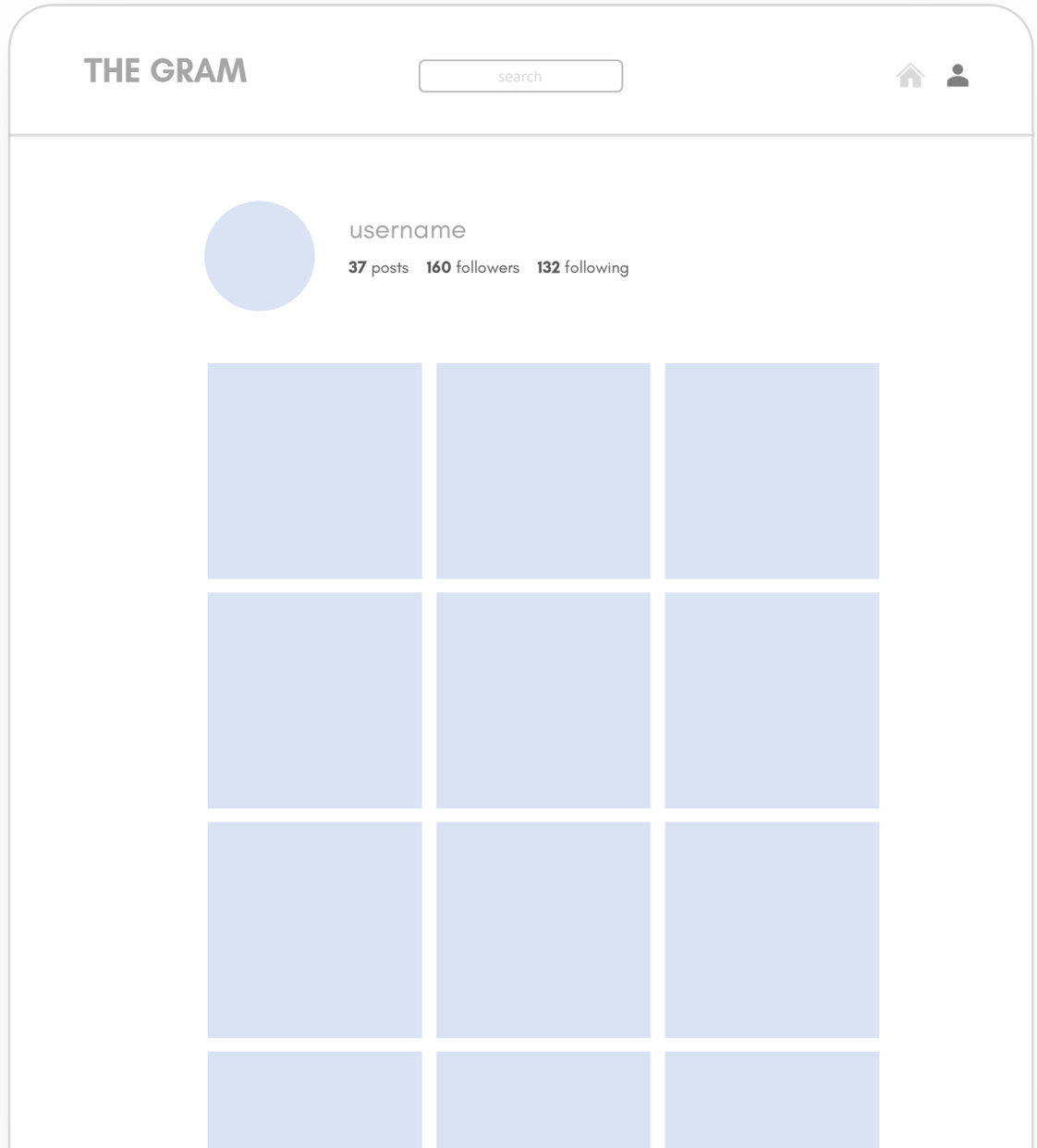


React

Why Pick React?

Before React's component based approach we would need to re-render the whole page in order to change the view. Or rely on complex manual DOM manipulation to change the view.

*Angular had approached the same problem in a different way

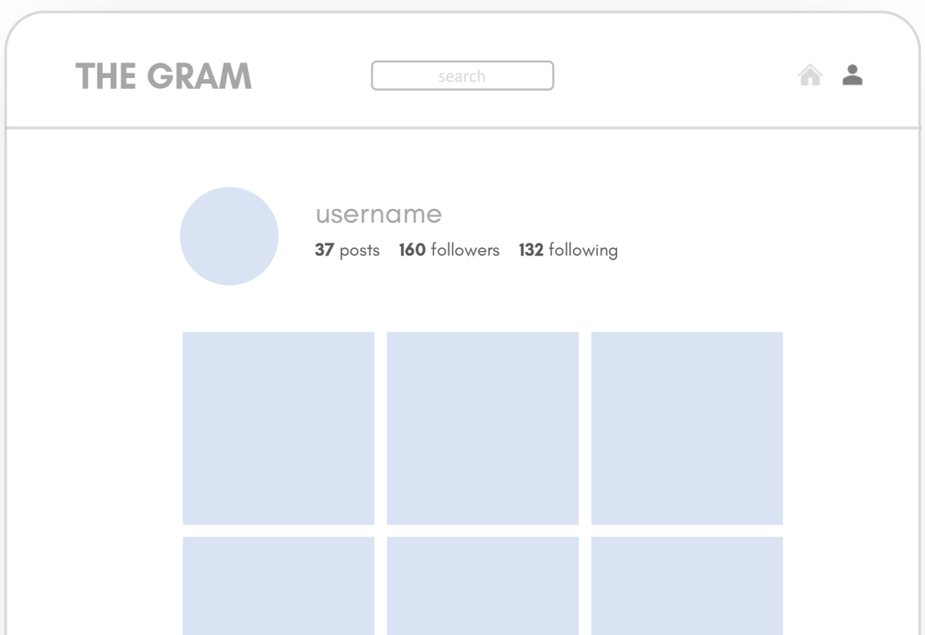


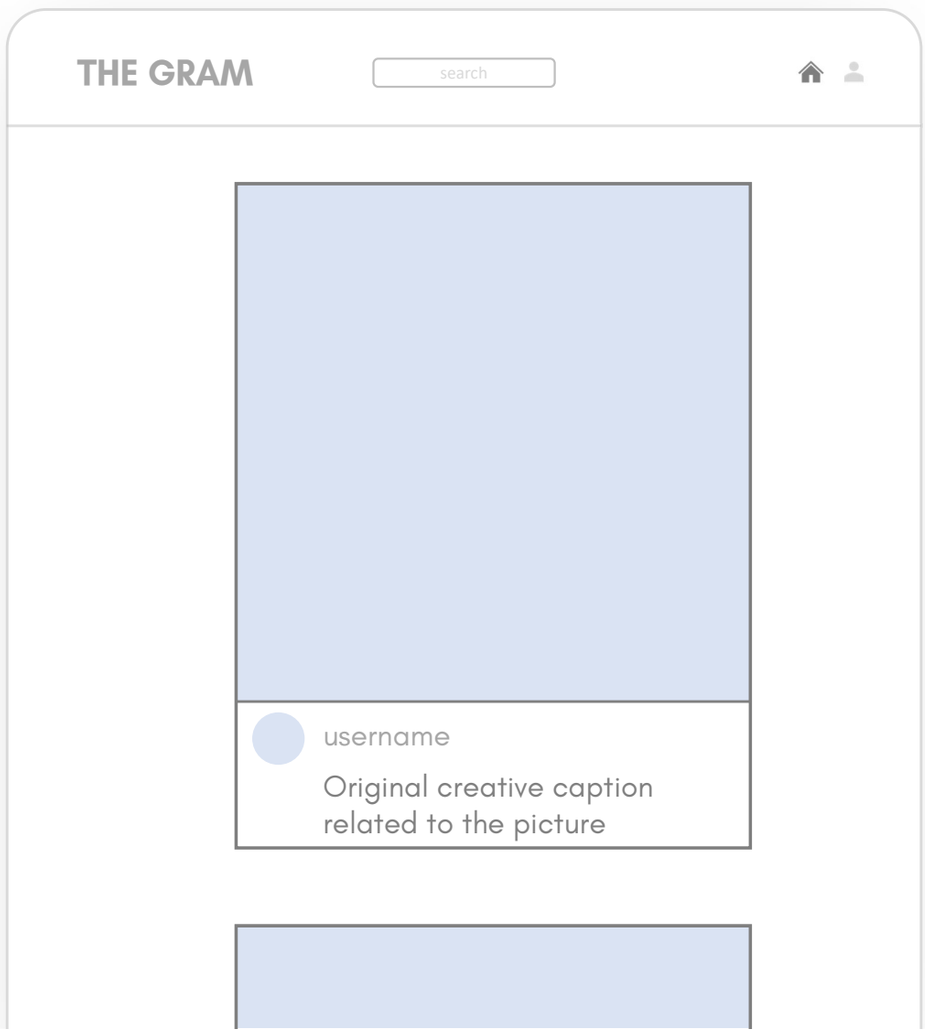
React

Why Pick React?

Before React's component based approach we would need to re-render the whole page in order to change the view. Or rely on complex manual DOM manipulation to change the view.

*Angular had approached the same problem in a different way





React

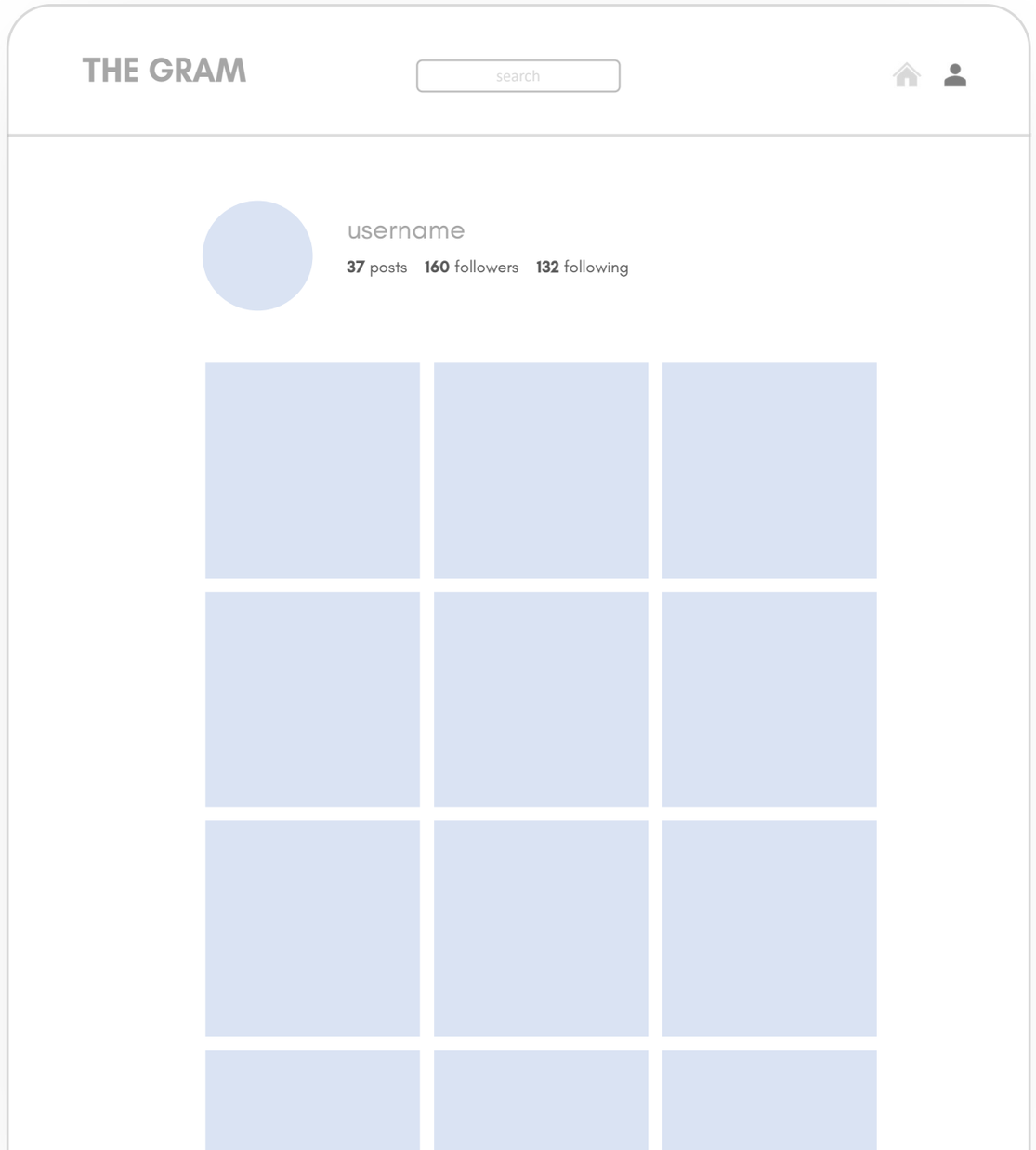
Getting Started

*Prerequisite: Install Node JS & NPM

```
$ npx create-react-app the-gram
```

```
$ cd the-gram
```

```
$ npm start
```



React

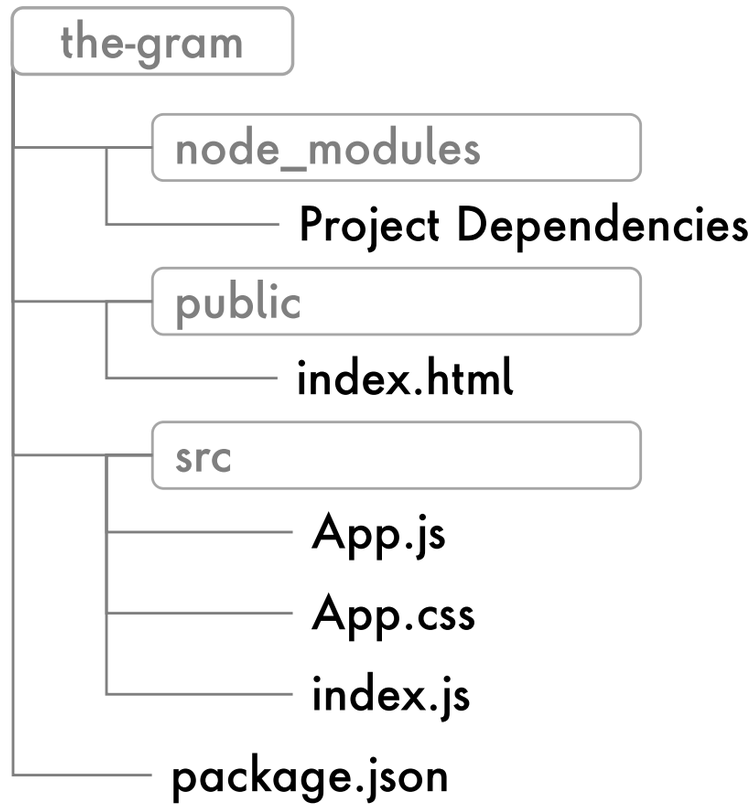
Getting Started

*Prerequisite: Install Node JS & NPM

```
$ npx create-react-app the-gram
```

```
$ cd the-gram
```

```
$ npm start
```



React

Linking HTML & JavaScript

public/index.hml

```html
<!DOCTYPE html>
<html lang="en">
  <body>
    <div id="root"></div>
  </body>
</html>
```

src/index.js

```jsx
import React from 'react';
import ReactDOM from 'react-dom';
import App from './App';

ReactDOM.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
  document.getElementById('root')
);
```

React

Linking Components

src/index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import App from './App';

ReactDOM.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
  document.getElementById('root')
);
```

src/App.js

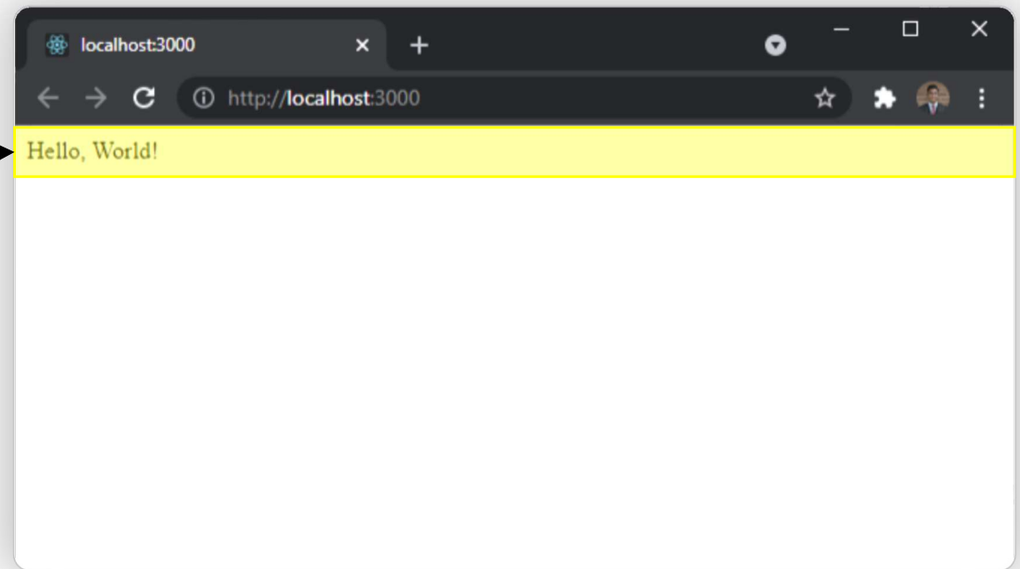
```
function App() {
  return (
    <div className="App">
        Hello, World!
    </div>
  );
}

export default App;
```

React

Seeing Is Believing

src/App.js

```
function App() {
  return (
    <div className="App">
      Hello, World!
    </div>
  );
}

export default App;
```



React

Components

Stateless Components are also called functional components – you can think of these as "dumb" components. They care only about rendering the UI based on data they have readily available.

Functional components are better fits when a component doesn't need to manage any data "state" on its own – i.e., they don't keep track of changing data.

```
// Stateless Component
// In ES5 Notation
function App() {
  return (
    <div className="App">
        Hello, World!
    </div>
  );
}
```

```
// Stateless Component
// In ES6 Notation
const App = () => (
  <div className="App">
      Hello, World!
  </div>
);
```

React

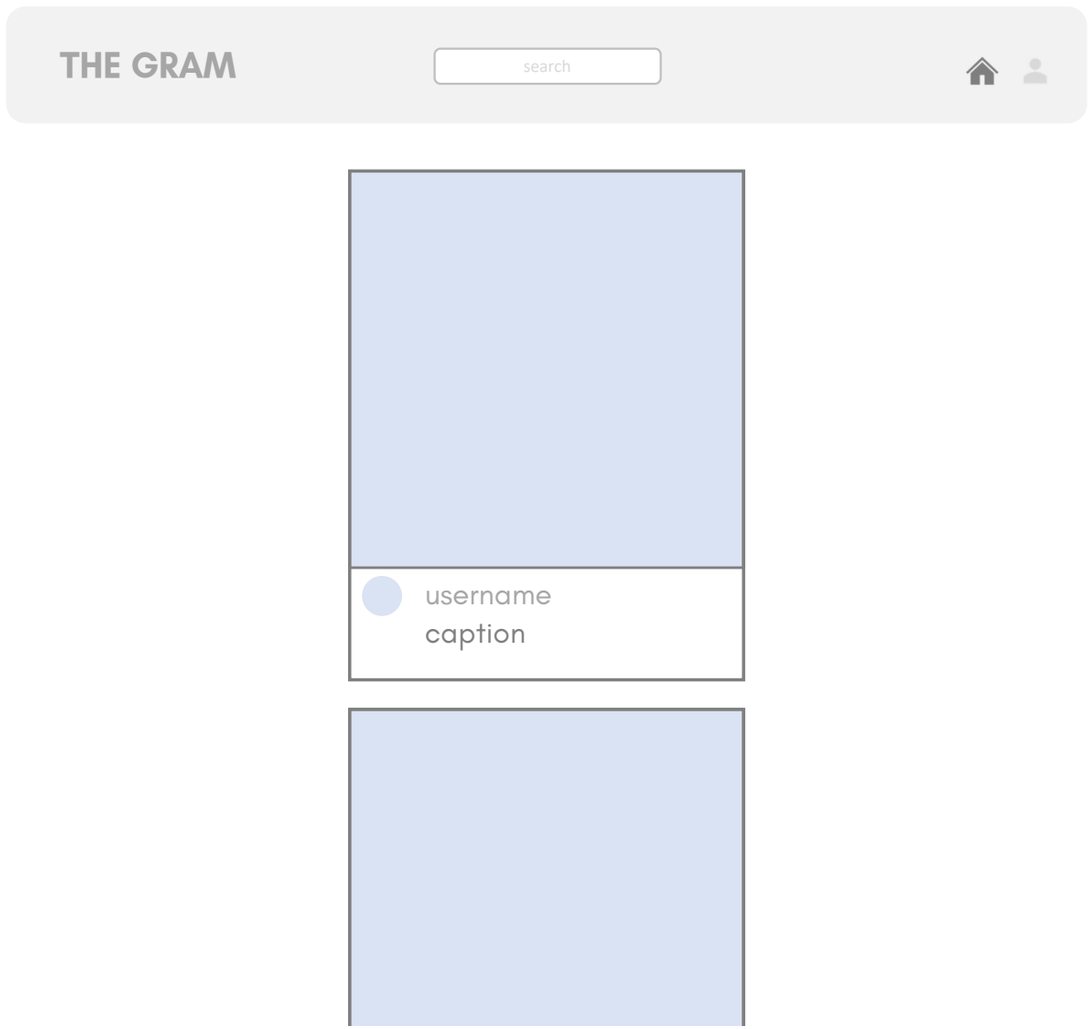
Components

A single image on The Gram for example, doesn't need to care about changing the caption, user name, or image, it just needs to display whatever data it has.

React

Components

Our navigation bar however will need to keep track of data – it needs to know what page we're on, and if what the last searched for value is.



React

Components

Stateful Components are components that need to keep track of some data that can be changed.

Examples can be a component keeping track of number of counts on a button, or keeping track of the value of an input.

We can update our “Hello, World!” to take a user’s name instead.

```
// Stateful Component
// In ES6 Notation
const App = () => {
  const [userName, setUserName] = useState(‘’);
  return (
    <div className="App">
      <input
        value={userName}
        onChange={(e) => setUserName(e.target.value)}
      />
      <div>Hello, {userName}!</div>
    </div>
  );
}
```

React

Props

To make stateless components more useful, we need to be able to pass dynamic values to it. This allows the stateless component to define the “structure” while the parent component can worry about keeping track of the actual data itself.

A good example is looking at our The Gram’s feed – we want to keep the basic structure of the UI the same but, we want to display different user names, captions, and images.

Props make this possible – a stateless component defines the structure while the parent tells it the values.

React

Props

Props can be values, functions, or even different stateless or stateful components entirely!

```
const StatelessChild = (props) => (
    <div className="stateless-child">
        <h6>THIS IS A STATELESS CHILD (1)</h6>
        Hello, {props.userName}!
    </div>
);
```

```
const StatefulParent = () => {
  const [userName, setUserName] = useState('World’);
  return (
    <div className="stateful-parent">
      <h6>THIS IS A STATEFUL PARENT (1)</h6>
      <input
        value={userName}
        onChange={(e) => setUserName(e.target.value)}
      />
      <StatelessChild userName={userName} />
    </div>
  );
}
```

React

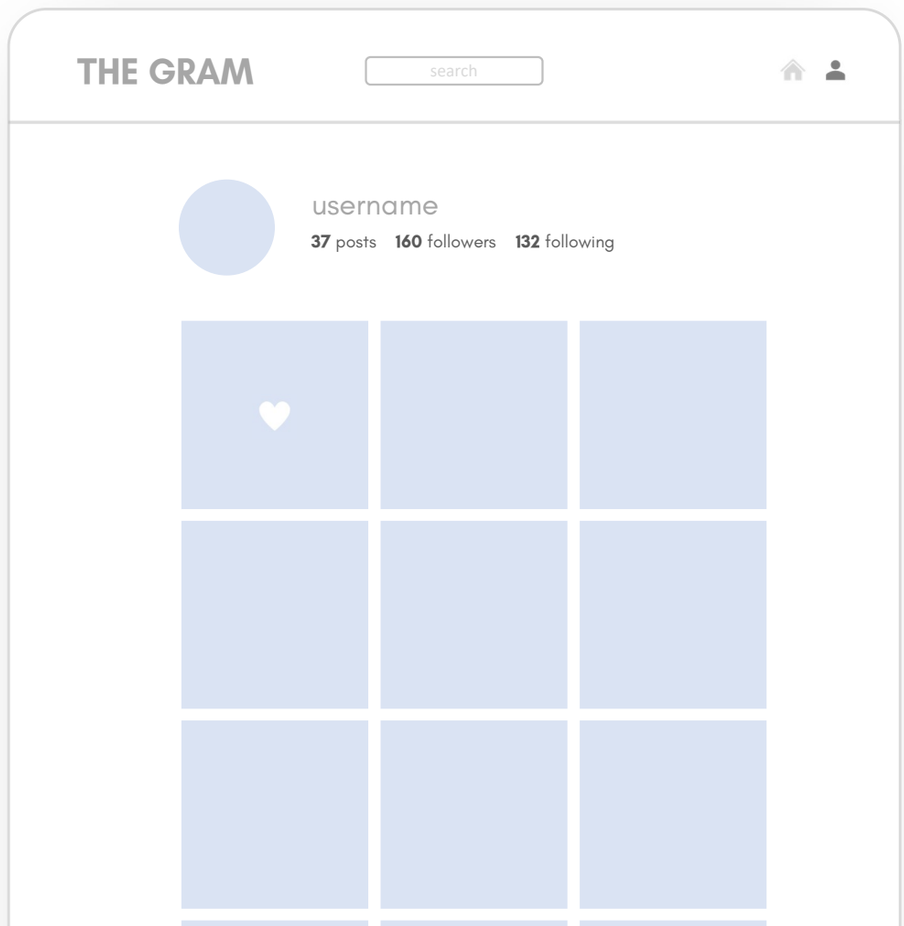
Props

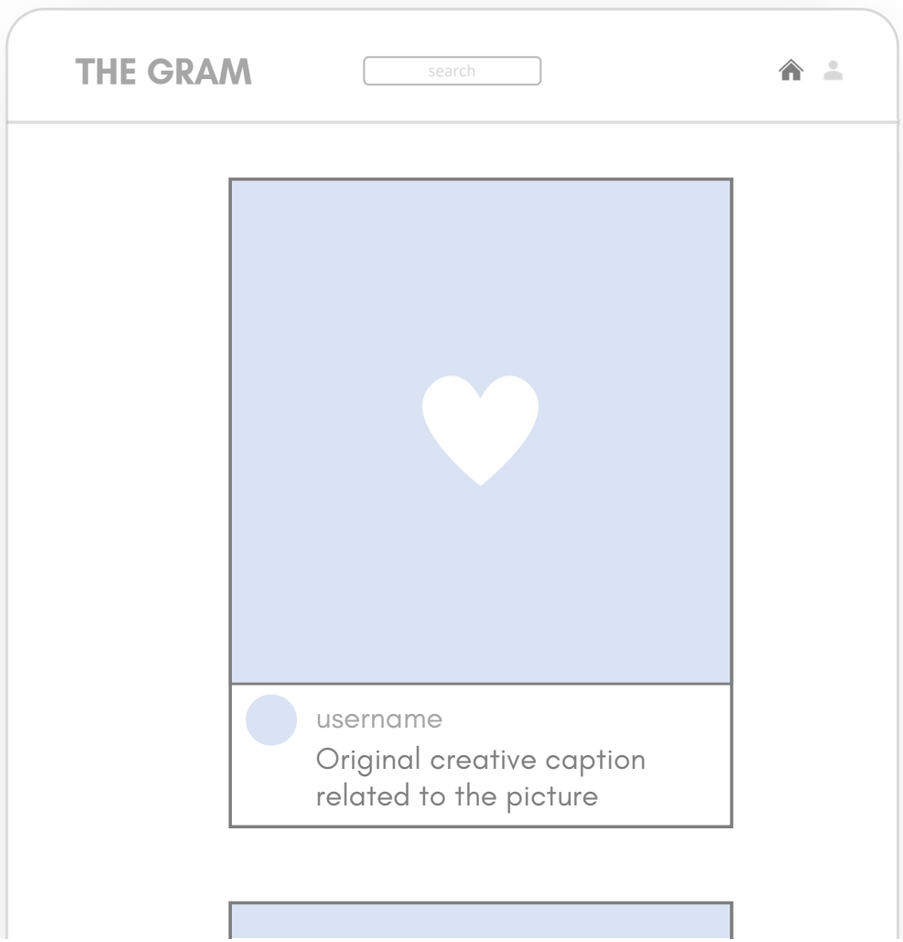
```
const StatefulParentTwo = () => {
    const [userName, setUserName] = useState('World');

    const changeHandler = (ev) => {
        setUserName(ev.target.value)
    }

    return (
        <div className="stateful-parent-two">
            <h6>THIS IS A STATEFUL PARENT (2)</h6>
            <StatelessChild userName={userName} />
            <StatelessChildTwo
                value={userName}
                onChange={changeHandler}
            />
        </div>
    );
}
```

```
const StatelessChild = (props) => (
    <div className="stateless-child">
        <h6>THIS IS A STATELESS CHILD (1)</h6>
        Hello, {props.userName}!
    </div>
);
```

```
const StatelessChildTwo = (props) => (
    <div className="stateless-child-two">
        <h6>THIS IS A STATELESS CHILD (2)</h6>
        <input
            value={props.value}
            onChange={props.onChange}
        />
    </div>
);
```

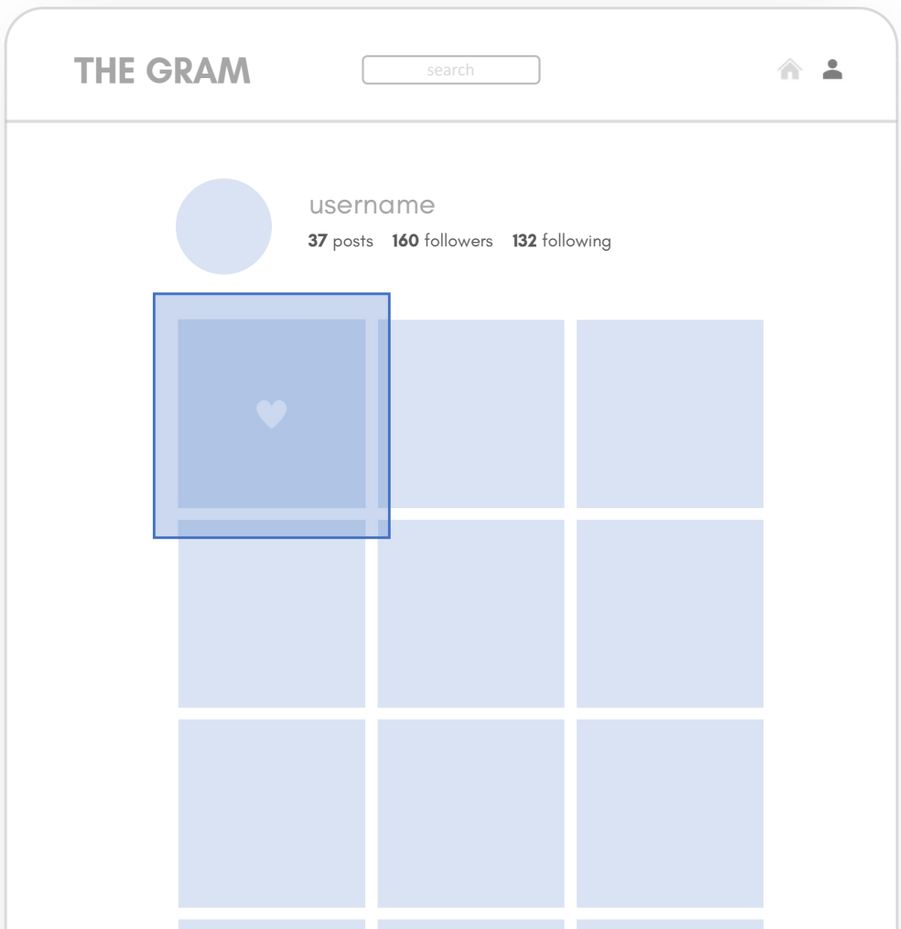
React | The Gram

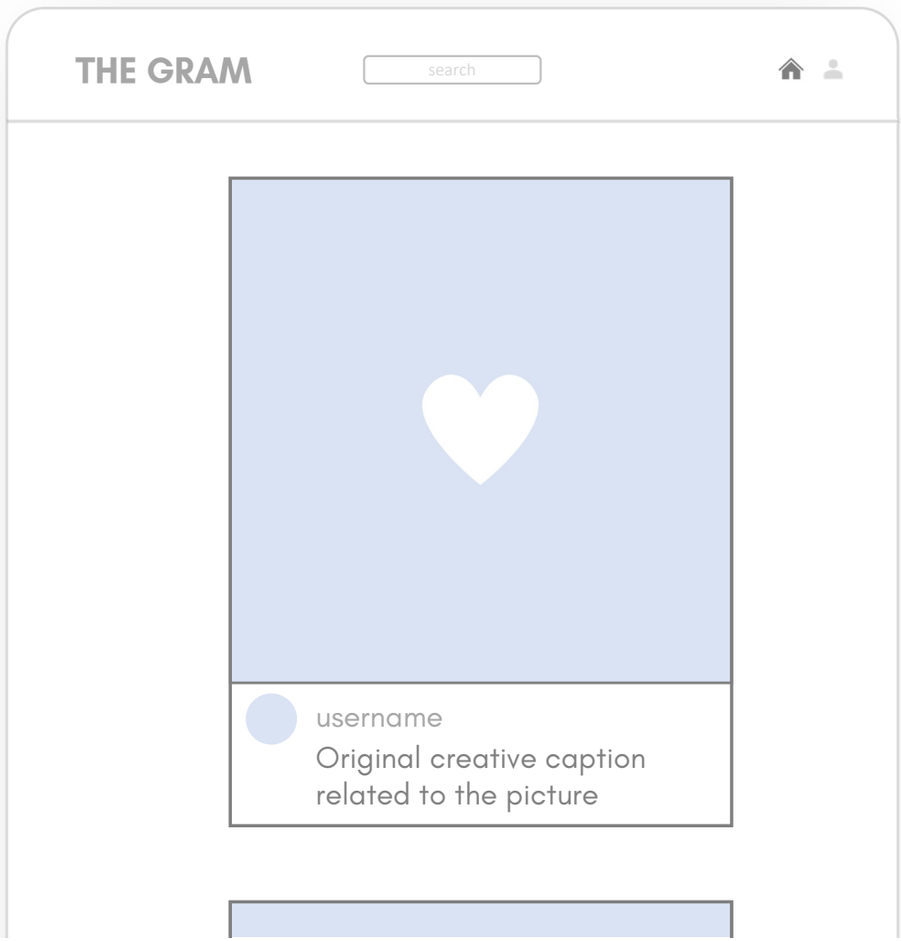




React | The Gram

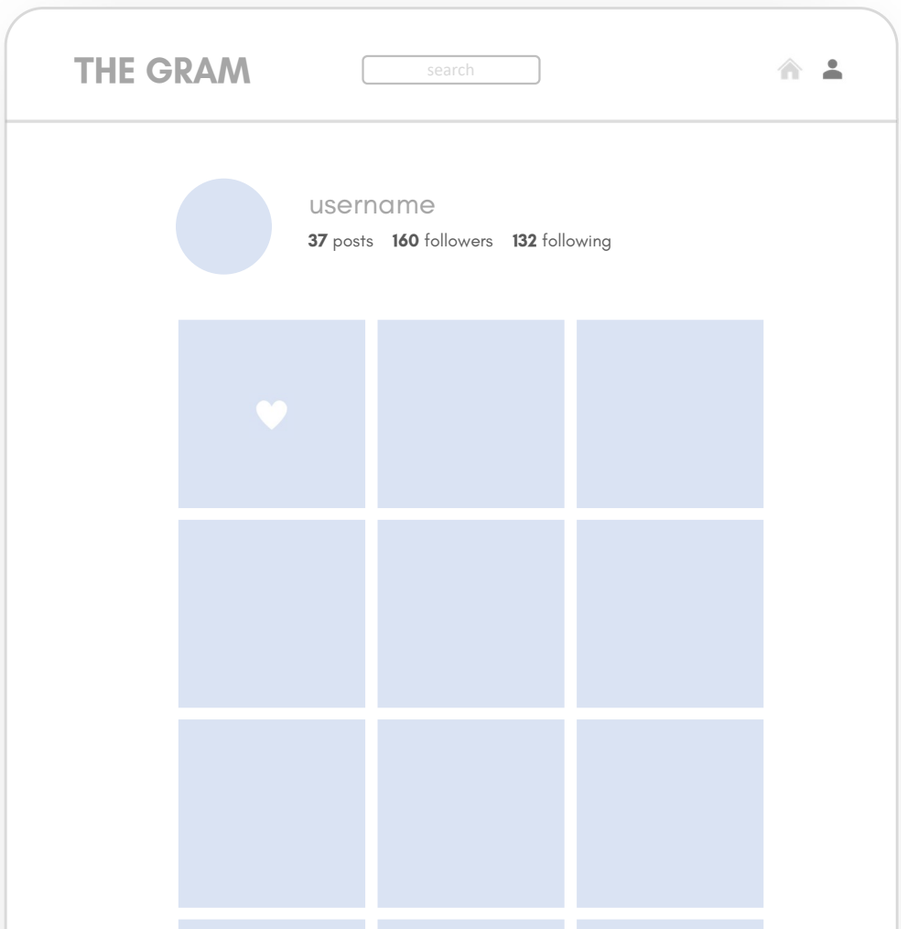
Break It Down

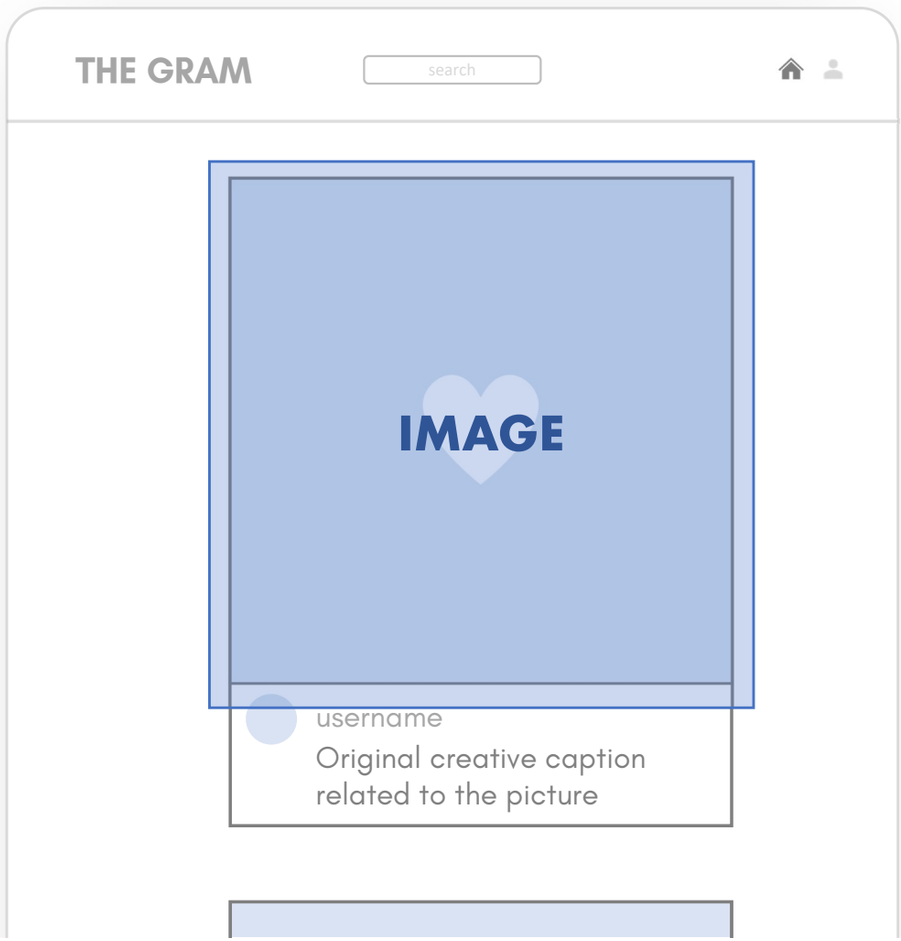




React | The Gram

Break It Down

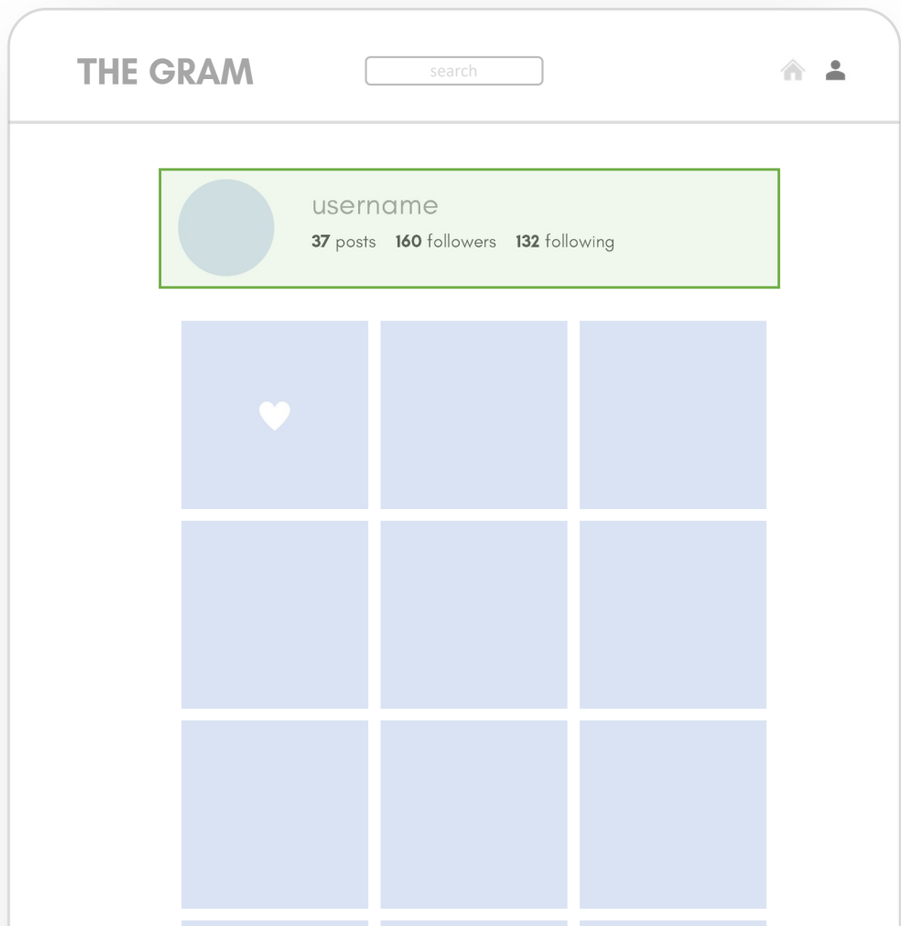


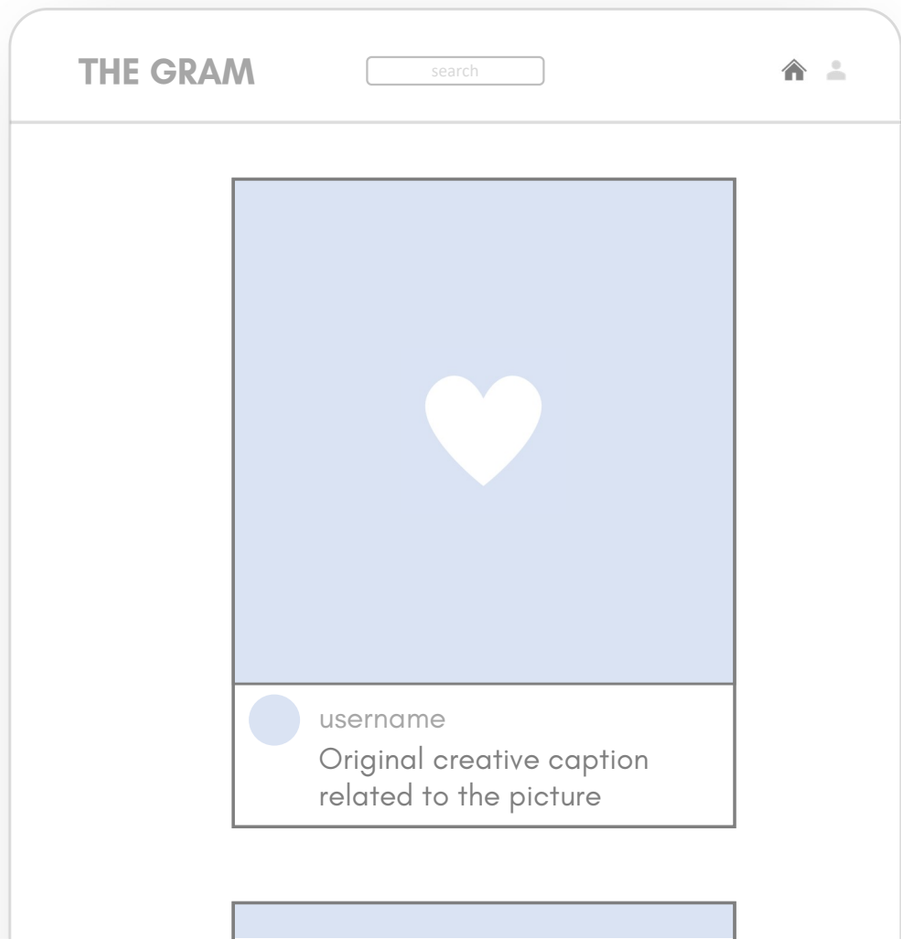


React | The Gram

Break It Down

IMAGE

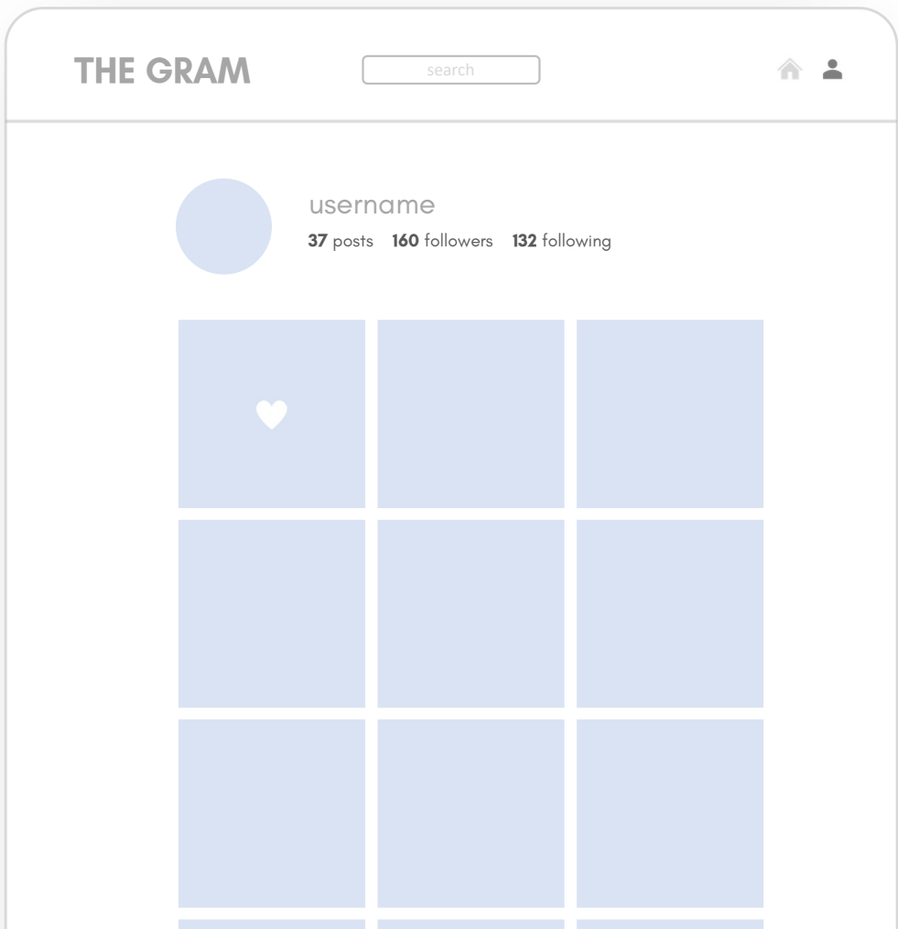


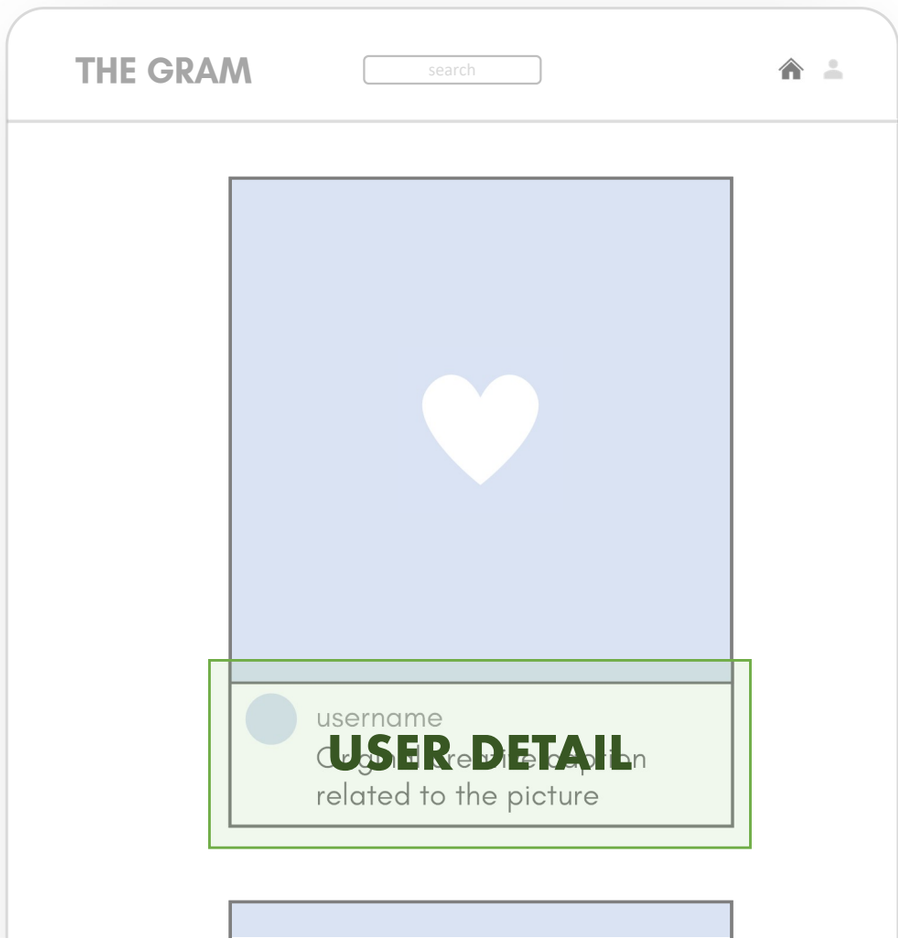


React | The Gram

Break It Down

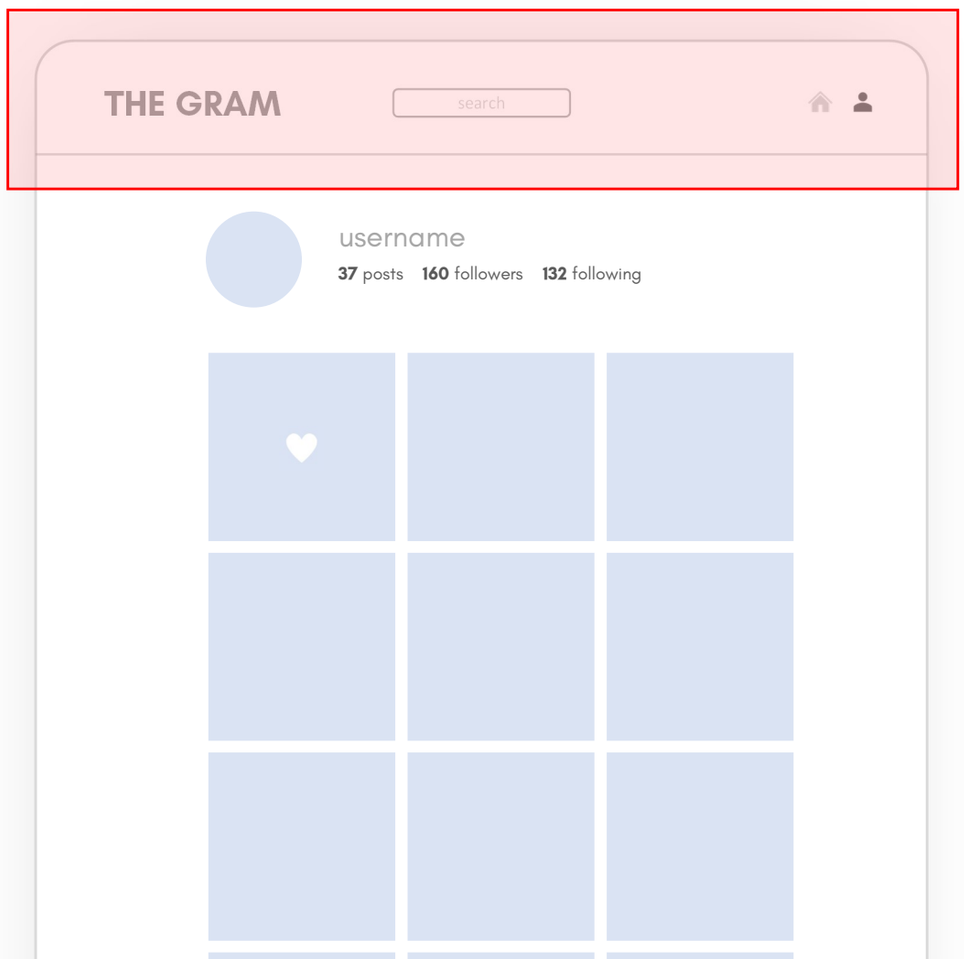
IMAGE

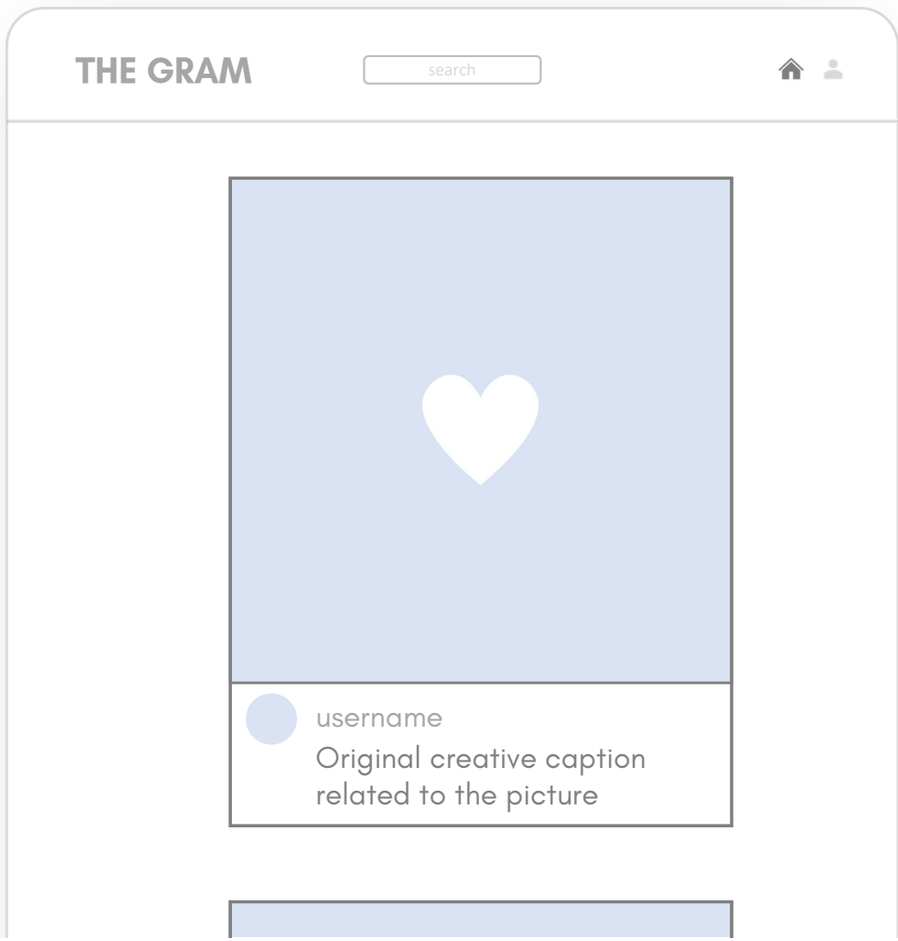




React | The Gram

Break It Down



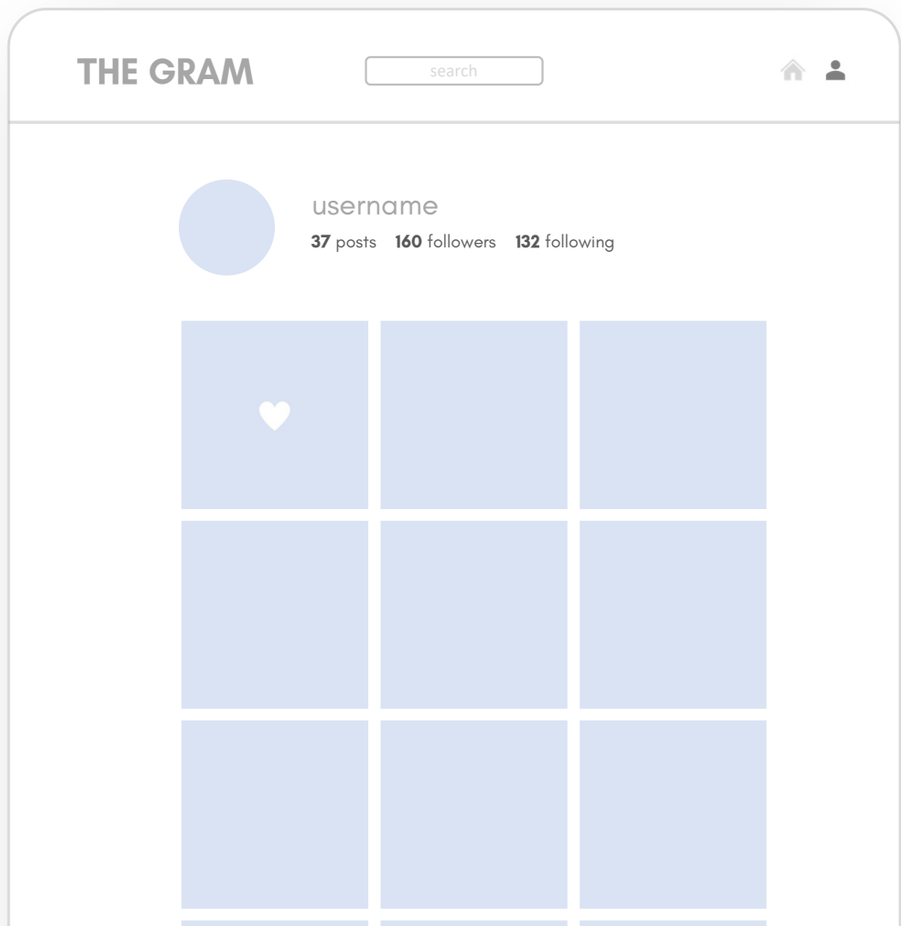


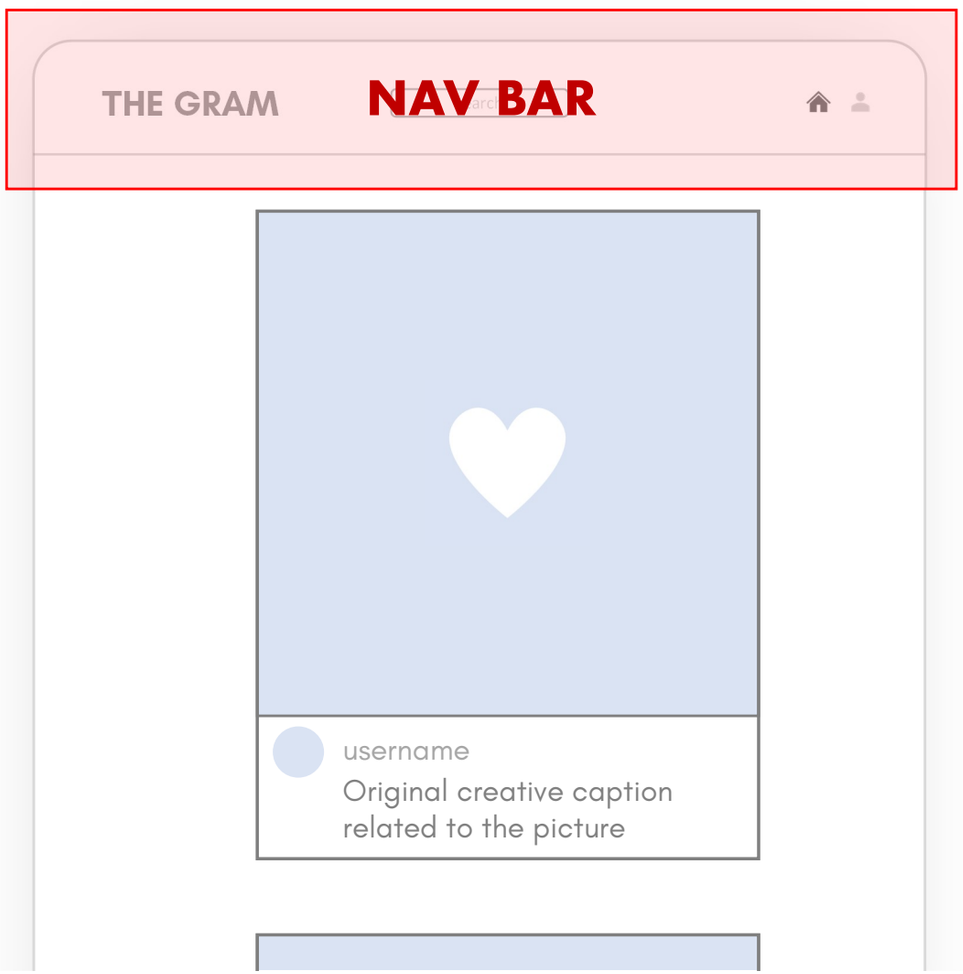
IMAGE

USER DETAIL

React | The Gram

Break It Down



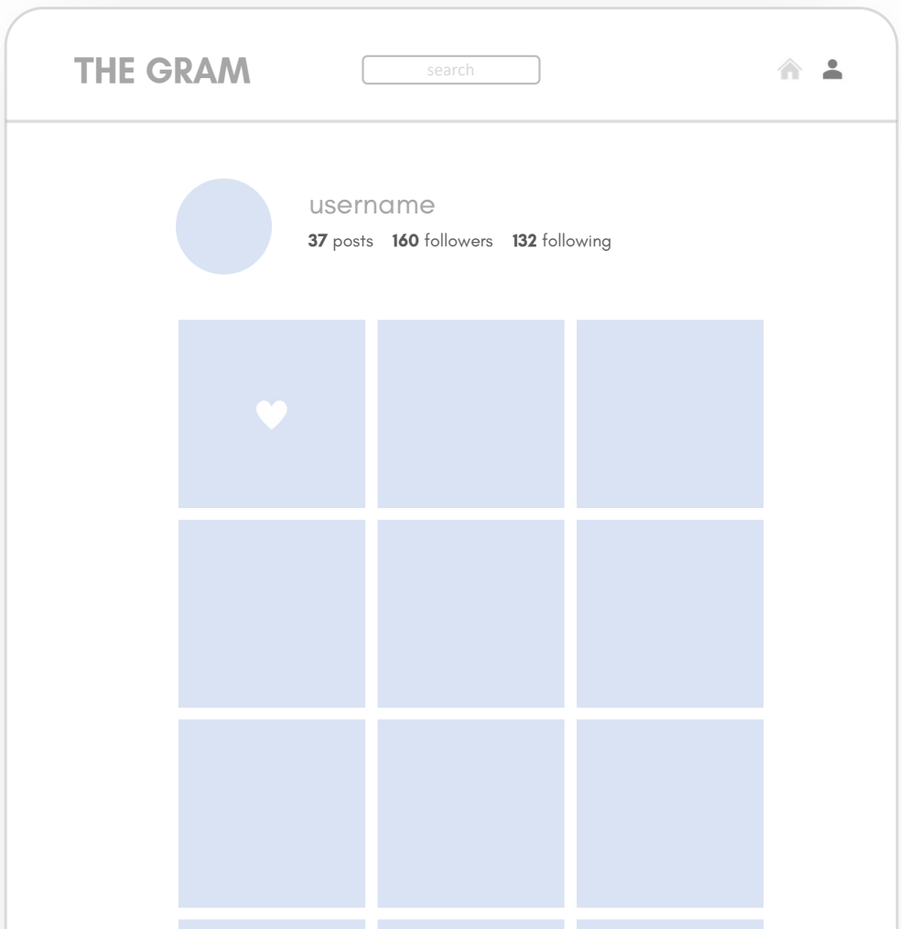


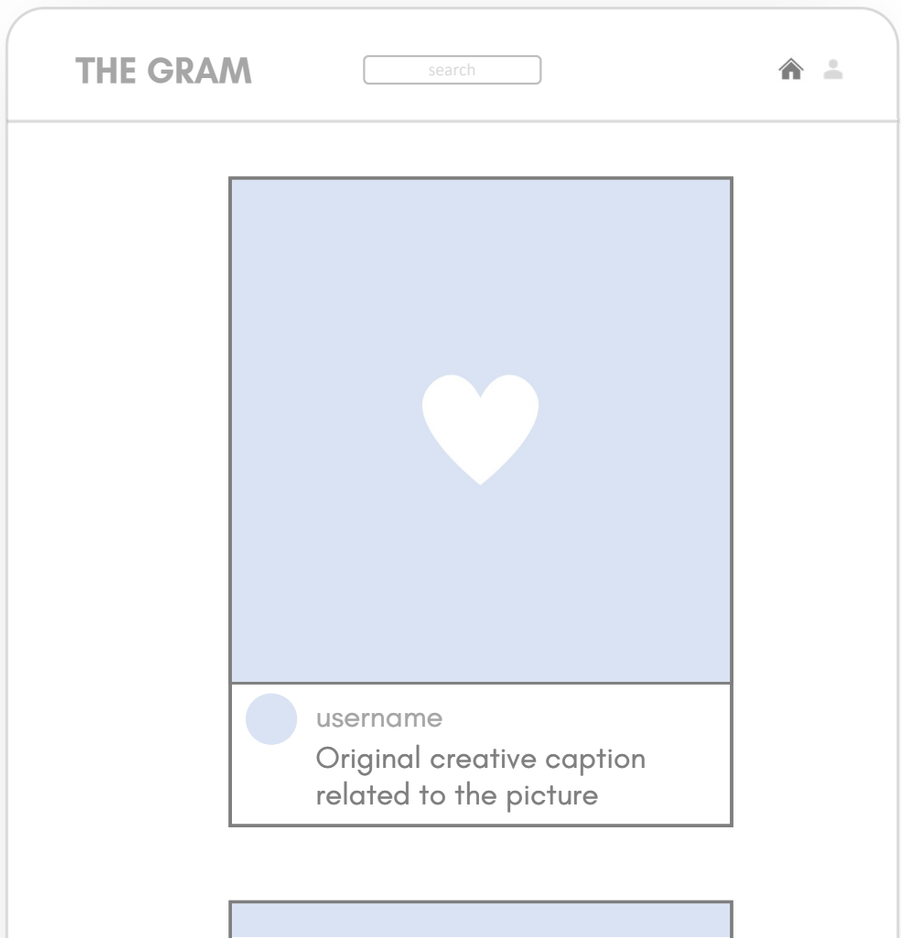
IMAGE

USER DETAIL

React | The Gram

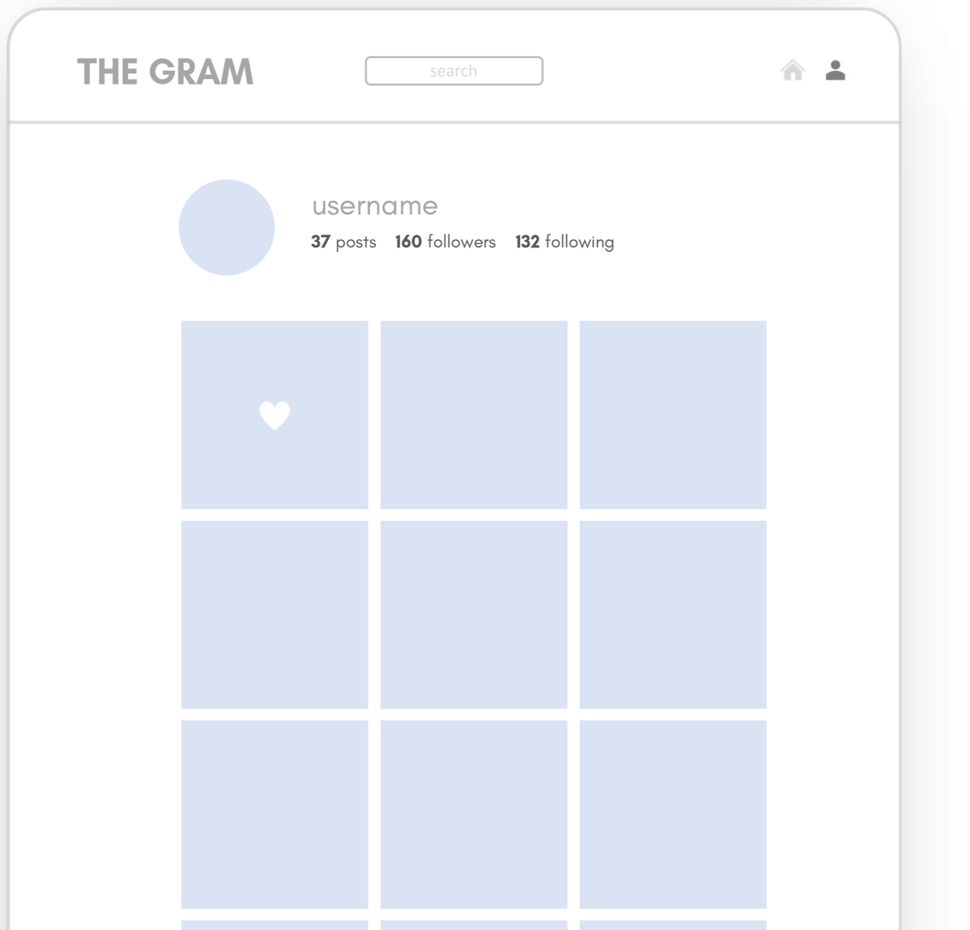
Break It Down

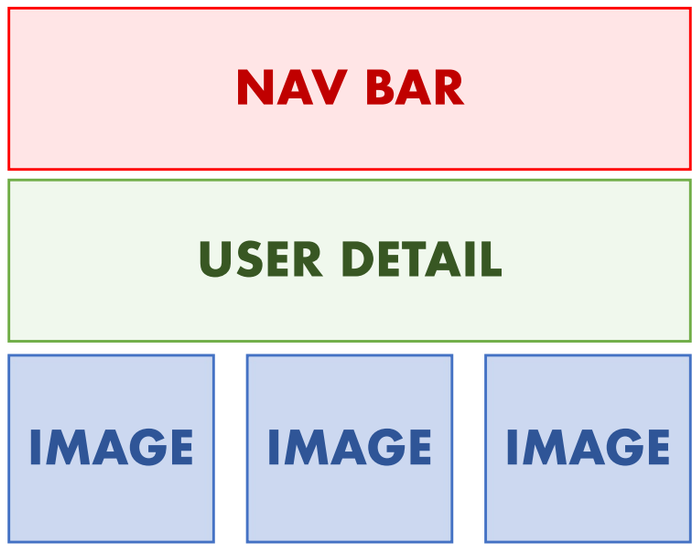




React | The Gram

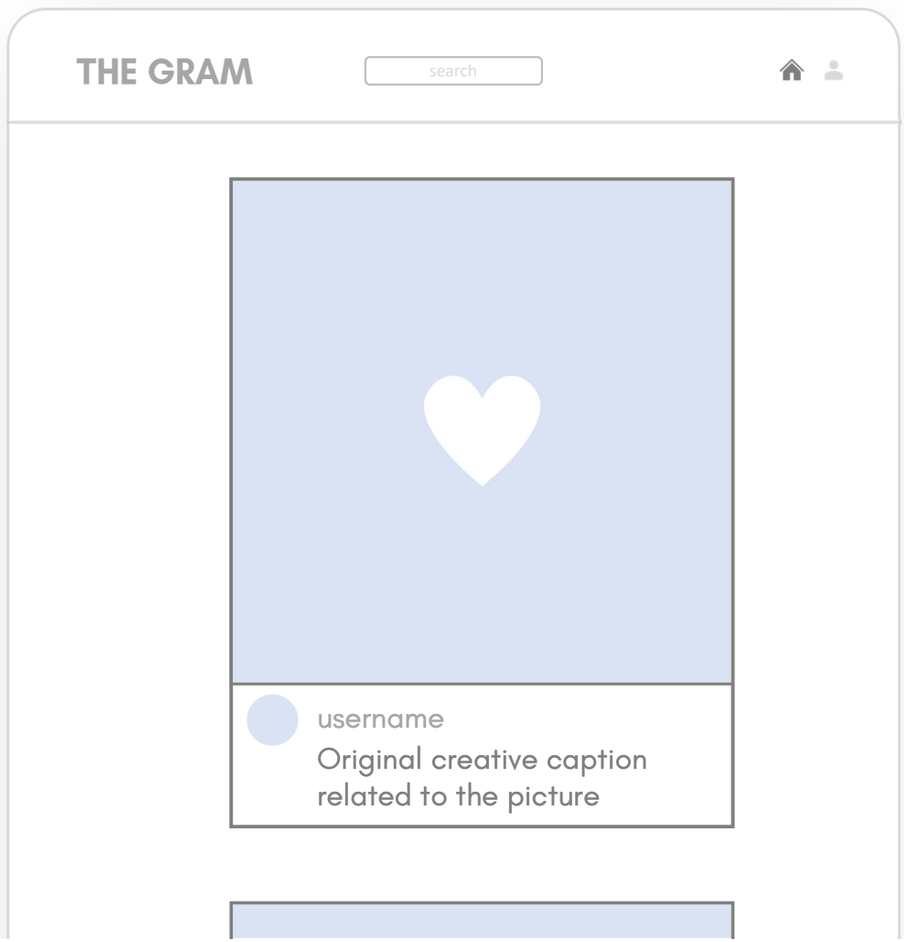
Break It Down

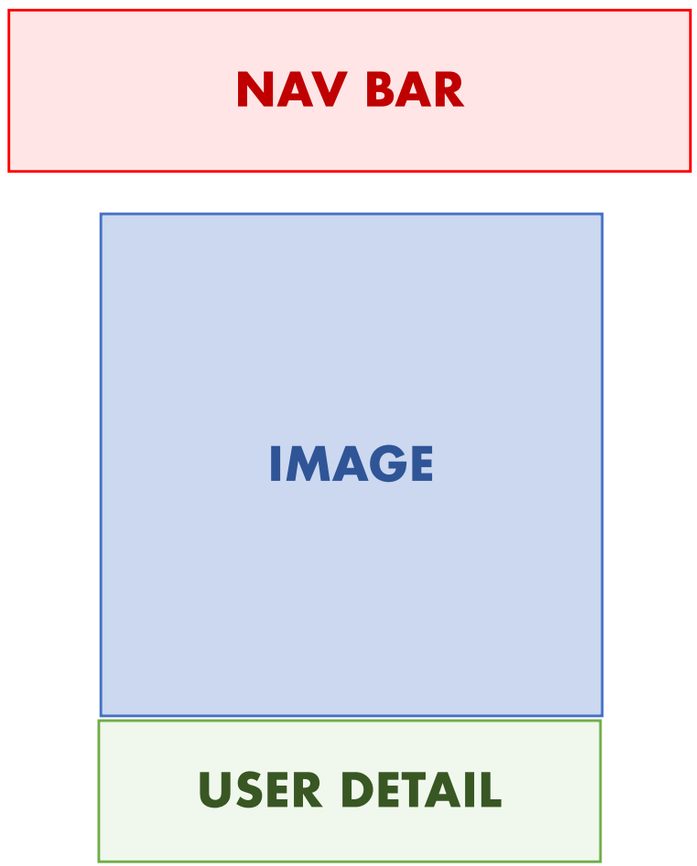




React | The Gram

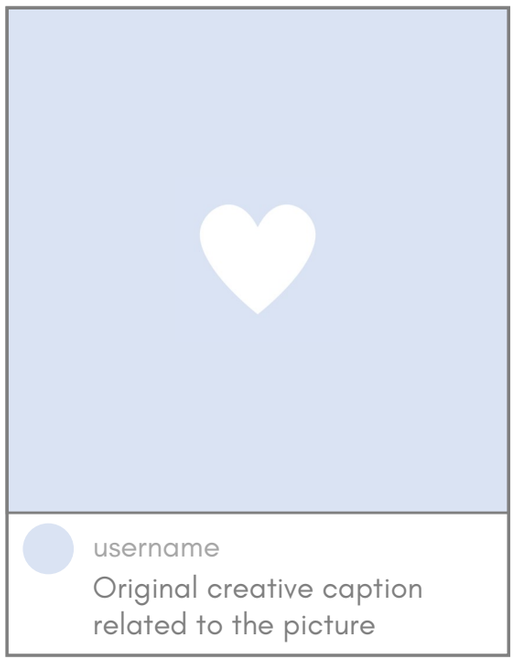
Break It Down

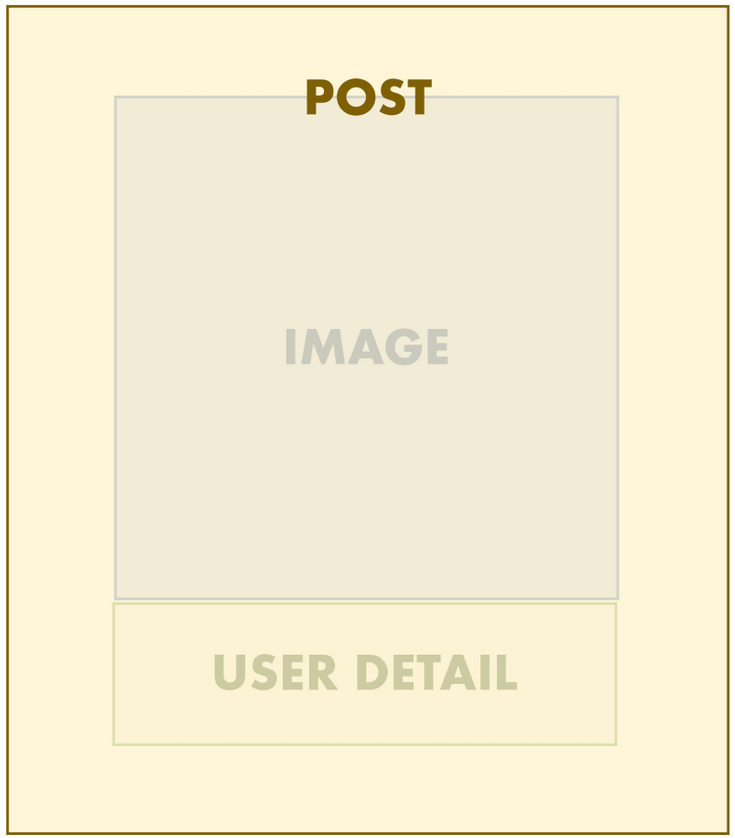




React | The Gram

Break It Down





React | The Gram

IMAGE

STATELESS

Props
Image URL
Is Liked
On Click

USER DETAIL

STATELESS

Props
User Name
Lower Half
Size

POST

STATEFUL

Props
User Name
Lower Half
Size
Image URL

State
Is Liked

Methods
On Click

NAV BAR

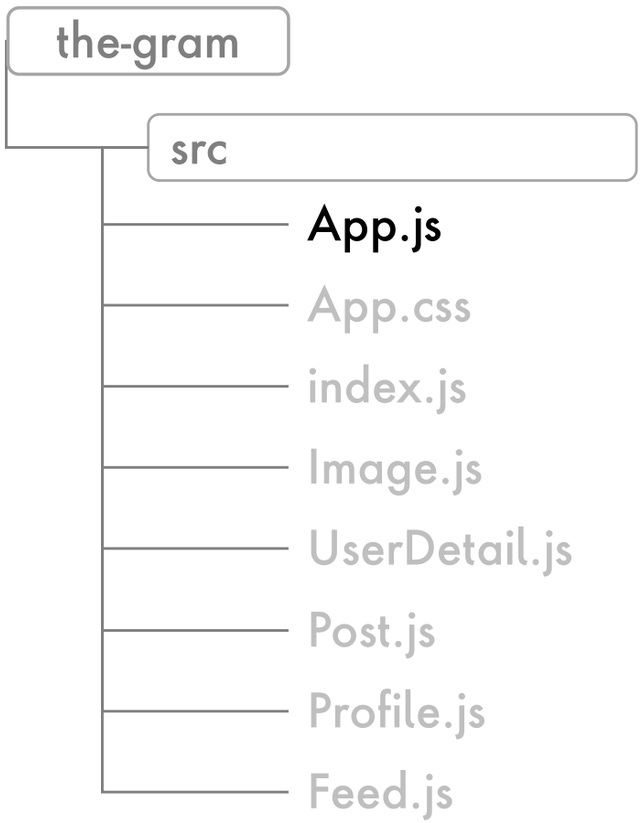
STATEFUL

Props
*(Hooks/Listeners)

State
Current Tab
Search Value

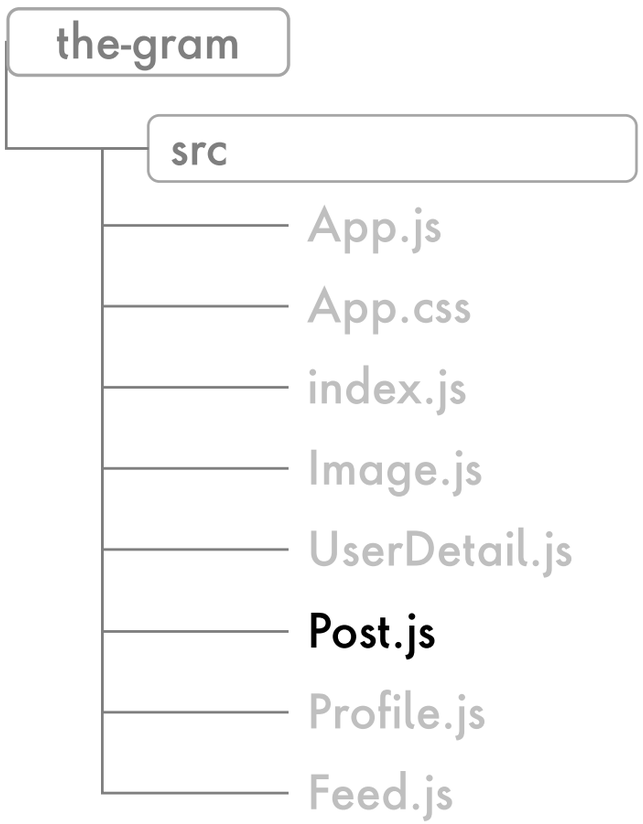
Methods
On Tab Change
On Search

React | The Gram



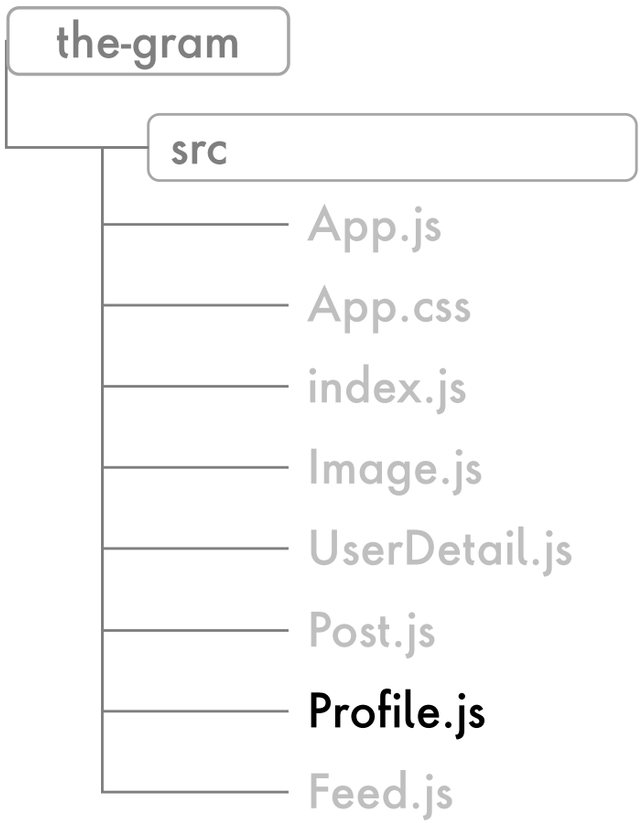

```
// Combines NavBar, Feed, & Profile
// Uses sample data exported from data.js

const App = () => {
  const [activeTab, setActiveTab] = useState(null);
  return (
    <div>
      <NavBar
        onChangeActiveTab={changedTab}
        onUpdateSearchValue={onSearch}
      />
      <Switch>
        <Route
          path="/theGram/profile“
          render={() => <Profile data={data} />}
        />
        <Route
          path="/theGram/"
          exact
          render={() => <Feed posts={data.feed} /> }
        />
      </Switch>
    </div>
  )
}
```

React | The Gram



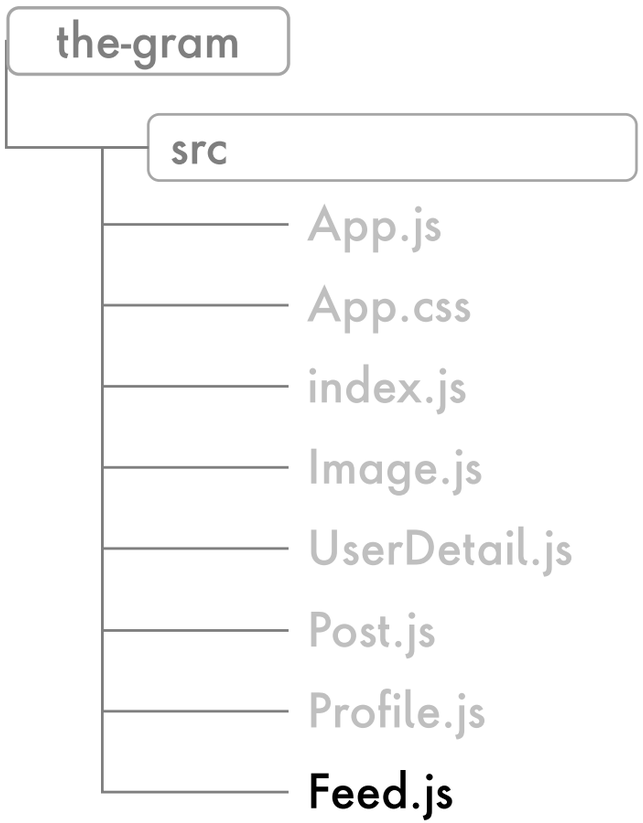

```jsx
// Post is a Stateful Component
// It combines Image & User Detail
// Keeps track of changing liked status

const Post = (props) => {
  const [liked, setLiked] = useState(props.liked || false);

  const updateLikedStatus = () => {
    setLiked(!liked);
  }

  return (
    <div className="gram-user-card">
      <Image
        doubleClick={updateLikedStatus}
        isLiked={liked}
        color={props.color}
      />
      <UserDetail
        size="small"
        userName={props.userName}
        lowerHalf={<p>{props.caption}</p>}
      />
    </div>
  );
}
```

React | The Gram




```
// Profile is a Stateless Component
// Profile & Post use the same components but, in
// different orders, and with different use cases

const Profile = (props) => (
    <div className="user-profile">
      <UserDetail size="large"
        userName={props.data.profile.userName}
        lowerHalf={
          <div>
            {props.data.posts.length} posts
            {props.data.profile.followersCount} followers
            {props.data.profile.followingCount} following
          </div>
        }
      />
      <div className="profile-posts">
        {
          props.data.posts.map((post, i) => (
            <Image
              key={`image-${i}`}
              isLiked={post.liked}
              color={post.color} />
          ))
        }
      </div>
    </div>
  );
```

React | The Gram




```
// Feed is a Stateless Component
// It is simply a group of Posts being rendered
// from an array of given data

const Feed = (props) => (
  props.posts.map((post, i) => (
    <Post
      key={`post-${i}`}
      index={i}
      caption={post.caption}
      color={post.color}
      userName={post.userName}
      liked={post.liked}
    />
  ))
)
```

React

THE GRAM

https://react-workshop-bolt.web.app

https://github.com/ayushyamitabh/BoltReactWorkshop