Keep Your Control Structures Clean

Avoid Deep Nesting

Using Factory Functions & Polymorphism

Prefer Positive Checks
(*if isEmpty* vs *if isNotEmpty*)

Utilize Errors

Use Guards & Fail Fast

```
if (email.includes('@')) {
  // do stuff
}
```

```
if (!email.includes('@')) {
  return;
}

// do stuff
```

return; → Fail fast

if block → Guard

runs before main code runs

```
if (user.active) {
  if (user.hasPurchases()) {
    // do stuff
  }
}
```

```
if (!user.hasPurchases()) {
    return;
}

if (user.active) {
  // do stuff
}
```

return; → Fail fast

if block → Guard

Embrace Errors & Error Handling

Throwing + handling errors can replace if statements and lead to more focused functions

Simple rule: If something is an error ➔ Make it an error

```
if (!isEmail) {
  return {code: 422, message: ‘Invalid input’};
}
```

```
if (!isEmail) {
  const error = new Error(‘Invalid input’);
  error.code = 422;
  throw error;
}
```

ACADEMIND

Error Handling is "One Thing"!