



Welcome to

# Mastering Vue.js



# Hello!

**I am Thiago Passos**



[passos.com.au](https://passos.com.au)



[github.com/thiagospassos](https://github.com/thiagospassos)



[@thiagospassos](https://twitter.com/thiagospassos)

# My Goals?

- For you to be as excited about Vue.js as I am
- To hit the ground running
- Learn the process from start to finish
- For you to have fun



@thiagospassos

# About You?

- What's your name?
- What's your background?
- What do you expect from it?
- What's your main focus?



@thiagospassos

# Agenda



**Getting Started**



**Automated Testing**



**Beyond the Basics**



**Consuming APIs**



**Dynamic Forms and  
Validation**



**State Management**





# Mastering Vue.js Getting Started

# Agenda

Background

Working with Vue

Core Features

# Agenda

Background

Working with Vue

Core Features



# History

- Dynamic HTML 1995
- jQuery 2006
- Knockout.js 2010
- AngularJS 2010
- Backbone.js 2010



# Evan You

I figured, what if I could just extract the part that I really liked about Angular and build something really lightweight.



@thiagospassos



# What's Vue?

- A progressive JavaScript framework
- Incrementally adoptable, core is focused on view only
- Additional libraries support more sophisticated features



# Why Choose Vue?

## Approachable

Already know  
HTML, CSS and  
JavaScript?  
Read the guide  
and start  
building things  
in no time!

## Versatile

An incrementally  
adoptable  
ecosystem that  
scales between a  
library and a full-  
featured  
framework

## Performant

20KB min+gzip  
Runtime, Blazing  
Fast Virtual DOM,  
Minimal  
Optimization  
Efforts



@thiagospassos



# Features

- Reactivity
- Data Binding
- Components
- Events Handling
- Animation / Transition
- Computed Properties
- Templates
- Directives
- Routing
- State Management
- Lightweight
- Much more!





# Agenda

Background

Working with Vue

Core Features

# Essential Skills

HTML



JS



# Recommended Tools



Visual Studio Code



Vue VSCode  
Snippets



Vue  
Dev Tools



# Demo

- It would be completely irresponsible if we didn't start with a **Hello, World!**
- Show core features with a ToDo App



# Agenda

Background

Working with Vue

Core Features



# The Vue Instance

Started with a root Vue instance

Passed in an options object:

- el
- data
- methods
- computed

Additional options

```
var app = new Vue({  
  el: '#app',  
  data: {  
  },  
  methods: {  
  },  
  computed: {  
  },  
})
```



# Template Syntax

Text

{{ message }}

Attributes

v-bind:id="..." or :id="..."

Events

v-on:click="..." or @click="..."

Modifiers

@submit.prevent="..."



# Computed Properties

Reactive and Cached

```
computed: {  
  itemCount: function() {  
    return this.items.length;  
  }  
}
```

Normal binding

```
{{ itemCount }}
```



# Class and Style Bindings

Class

```
:class="{ done: t.done }"
```

Multiple Classes

```
:class="{ done: t.done, ... }"
```

Styles

```
:style="{ color: todoColor }"
```

Style object\*

```
{ todo: todoStyles }
```



# Conditional Rendering

v-if

```
<div v-if="items.length > 1">  
  {{ remaining }} item(s) remaining.  
</div>
```

v-else-if

```
<div v-else-if="items.length === 1">  
  Last item, get it done!  
</div>
```

v-else

```
<div v-else>  
  All done!  
</div>
```





# List Rendering

v-for

```
<a v-for="filter in filters">
  {{ filter }}
</a>

<a v-for="(filter, index) in filters">
  {{ index }} - {{ filter }}
</a>
```



# Event Handling

v-on or @

```
<a @click="activeFilter = filter">  
  {{ filter }}  
</a>
```



# Form Binding

v-model

```
<input v-model="todo" />
```



# Your Turn!

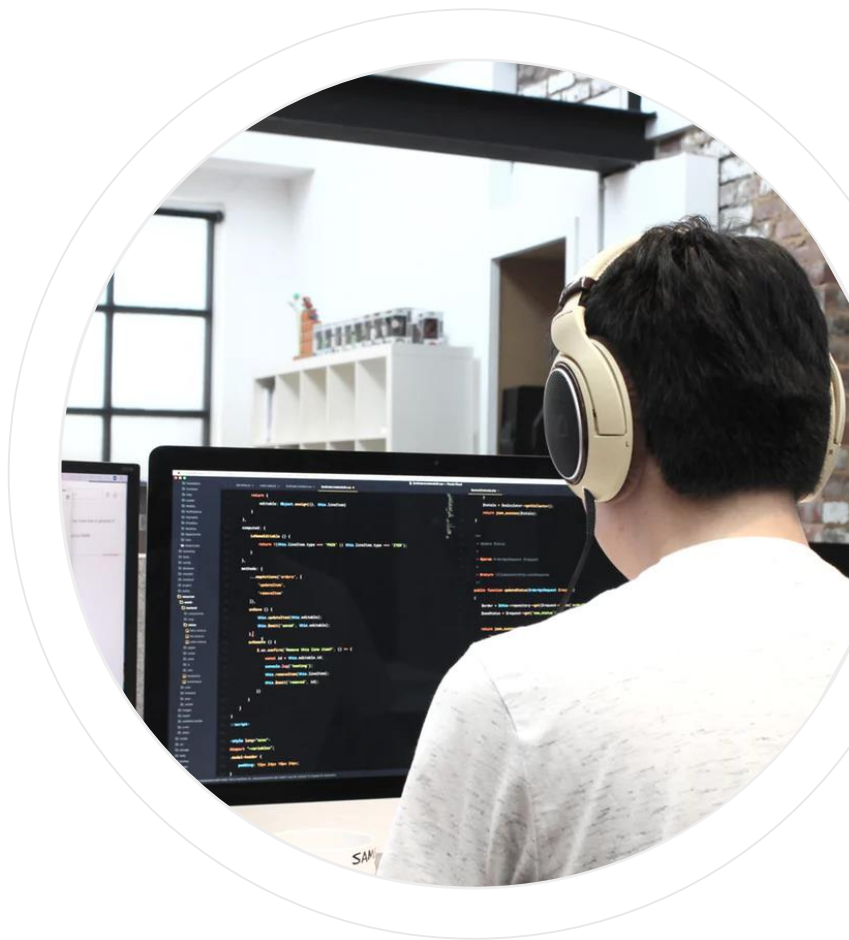
Follow the instructions to complete the code exercises and challenges



Workshop Guide  
[bit.ly/masteringvuejs](https://bit.ly/masteringvuejs)



Workshop Code  
[bit.ly/vuejs-oslo-2019](https://bit.ly/vuejs-oslo-2019)



@thiagospassos



# Mastering Vue.js **Beyond the Basics**



# Agenda

New App with the CLI

Views, Routes and more...

Debugging and Style Guide

# Agenda

New App with the CLI

Views, Routes and more...

Debugging and Style Guide

**What do  
you  
need?**



**What do  
we  
suggest?**



**Vetur  
ESLint  
Prettier  
Vue VSCode Snippet**





# Agenda

New App with the CLI

Views, Routes and more...

Debugging and Style Guide



# Agenda

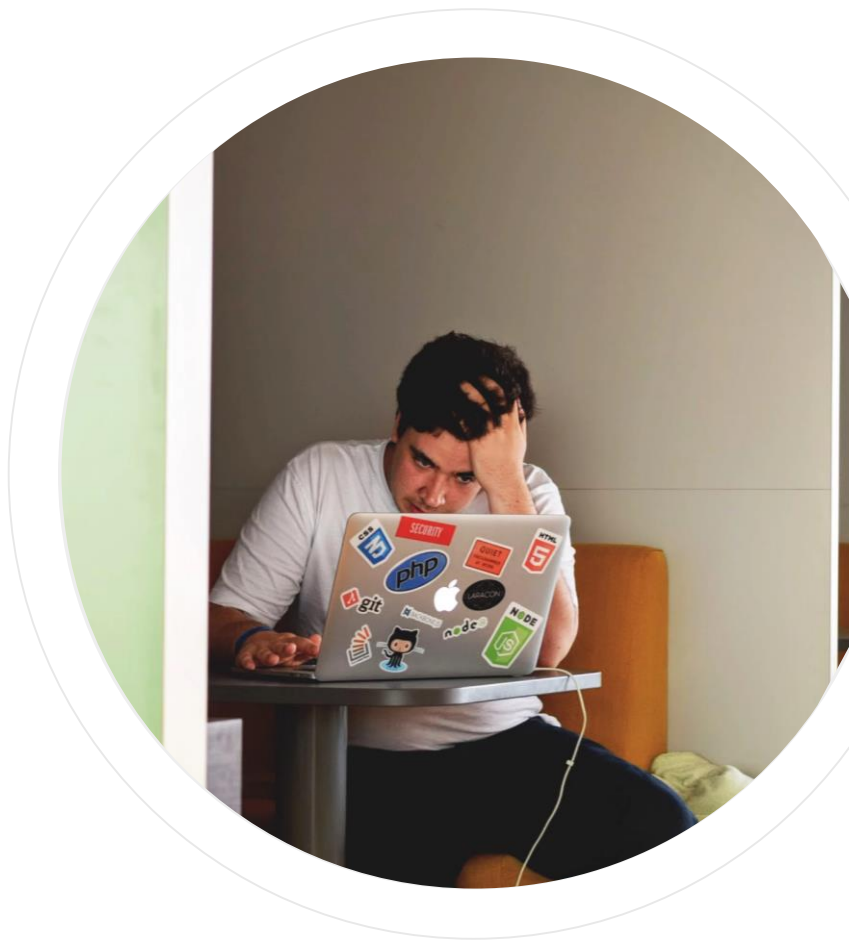
New App with the CLI

Views, Routes and more...

Debugging and Style Guide

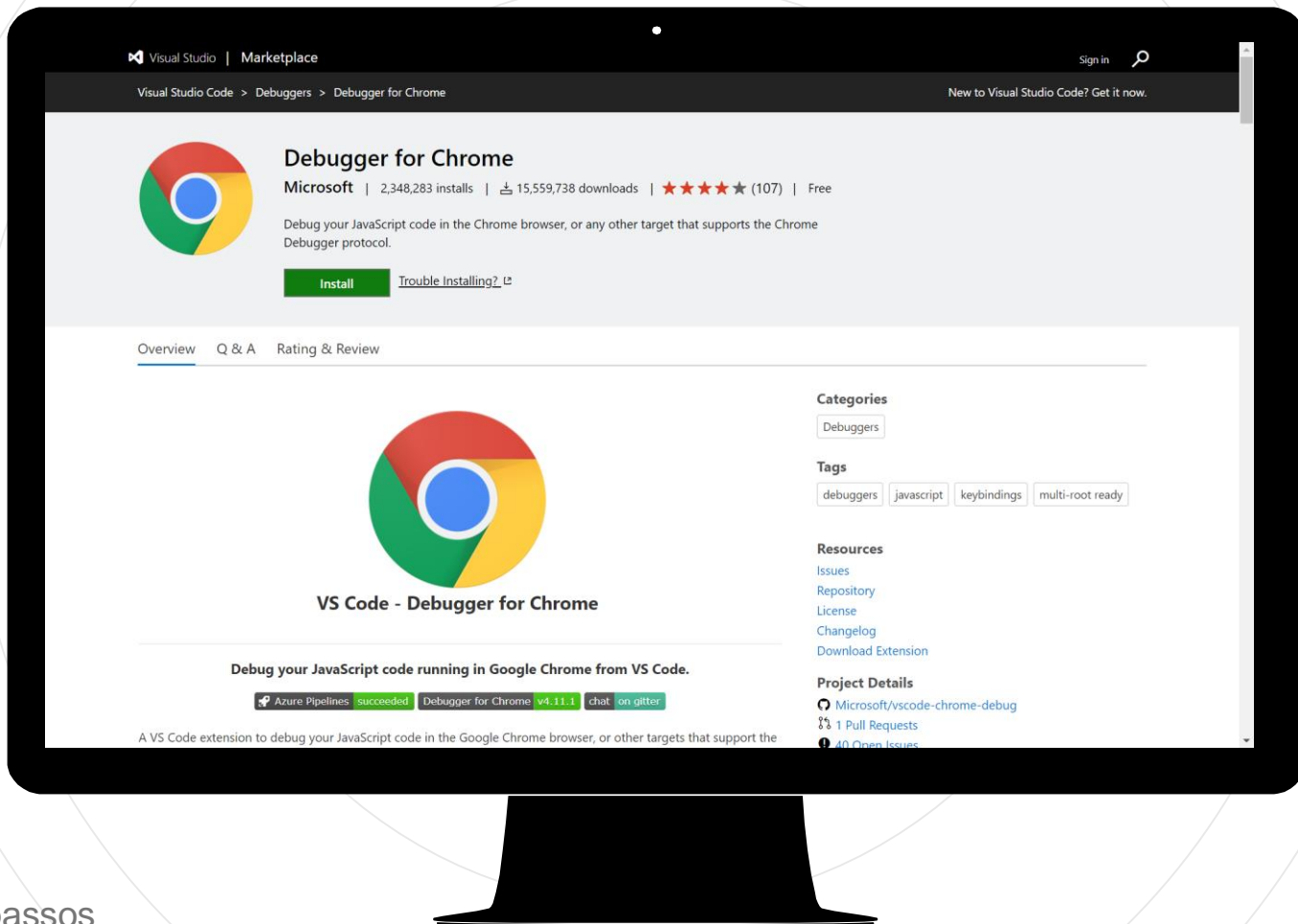
# Debugging

[bit.ly/vue-debugging](https://bit.ly/vue-debugging)



@thiagospassos





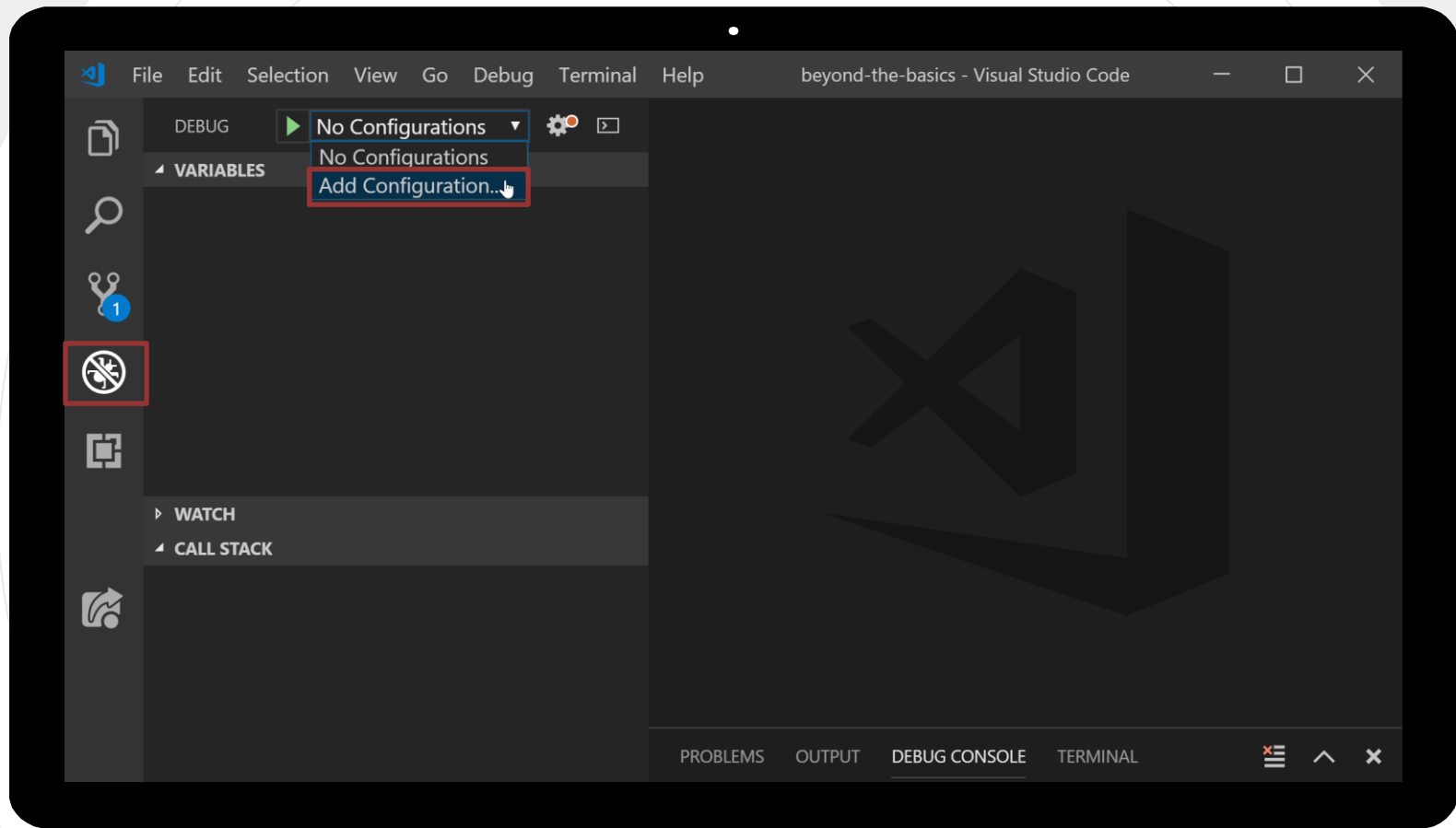
# Vue.config.js

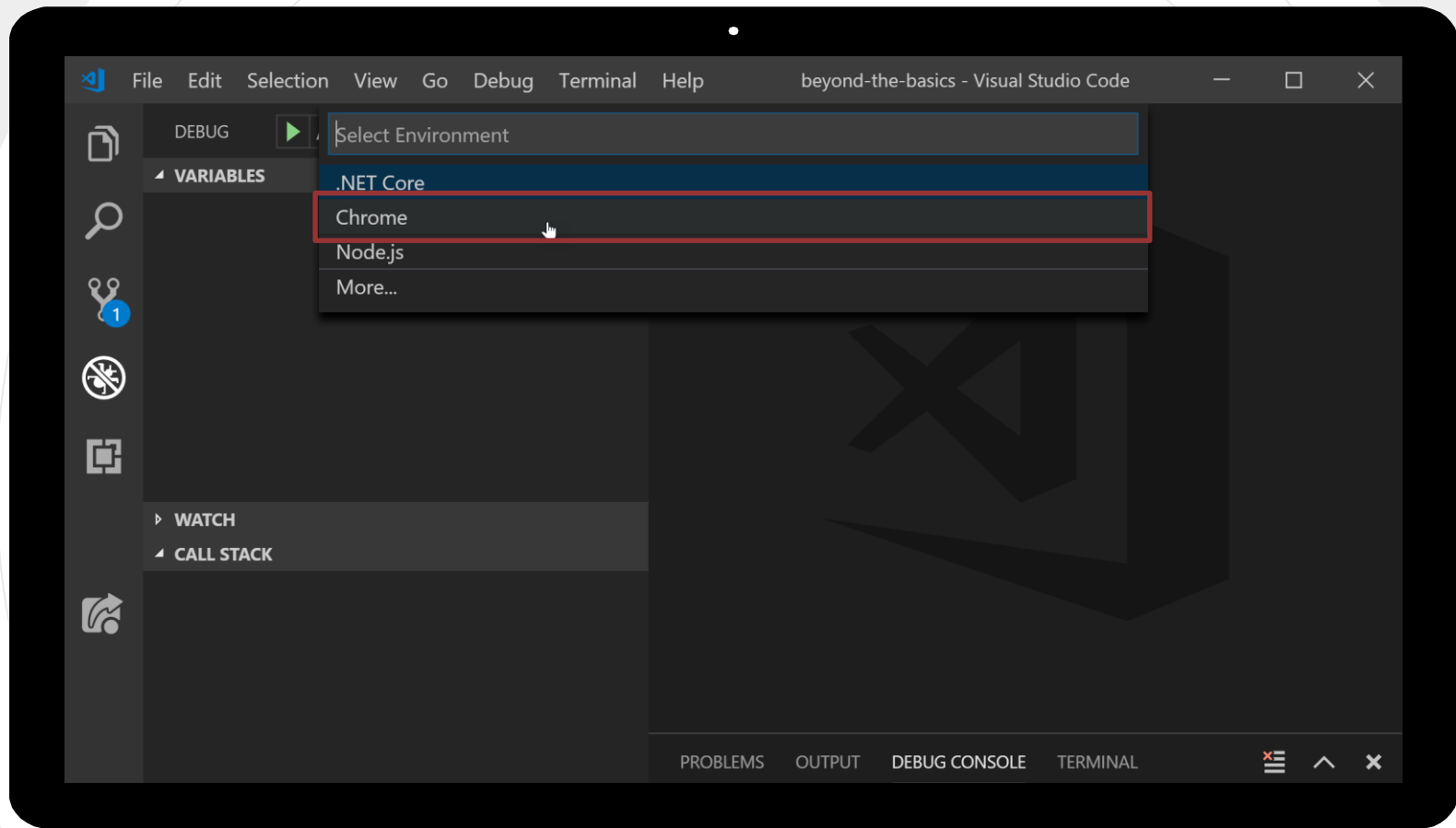
```
File Edit Selection View Go Debug Terminal Help masteringvuejs-code-demo - Visual Studio Code
```

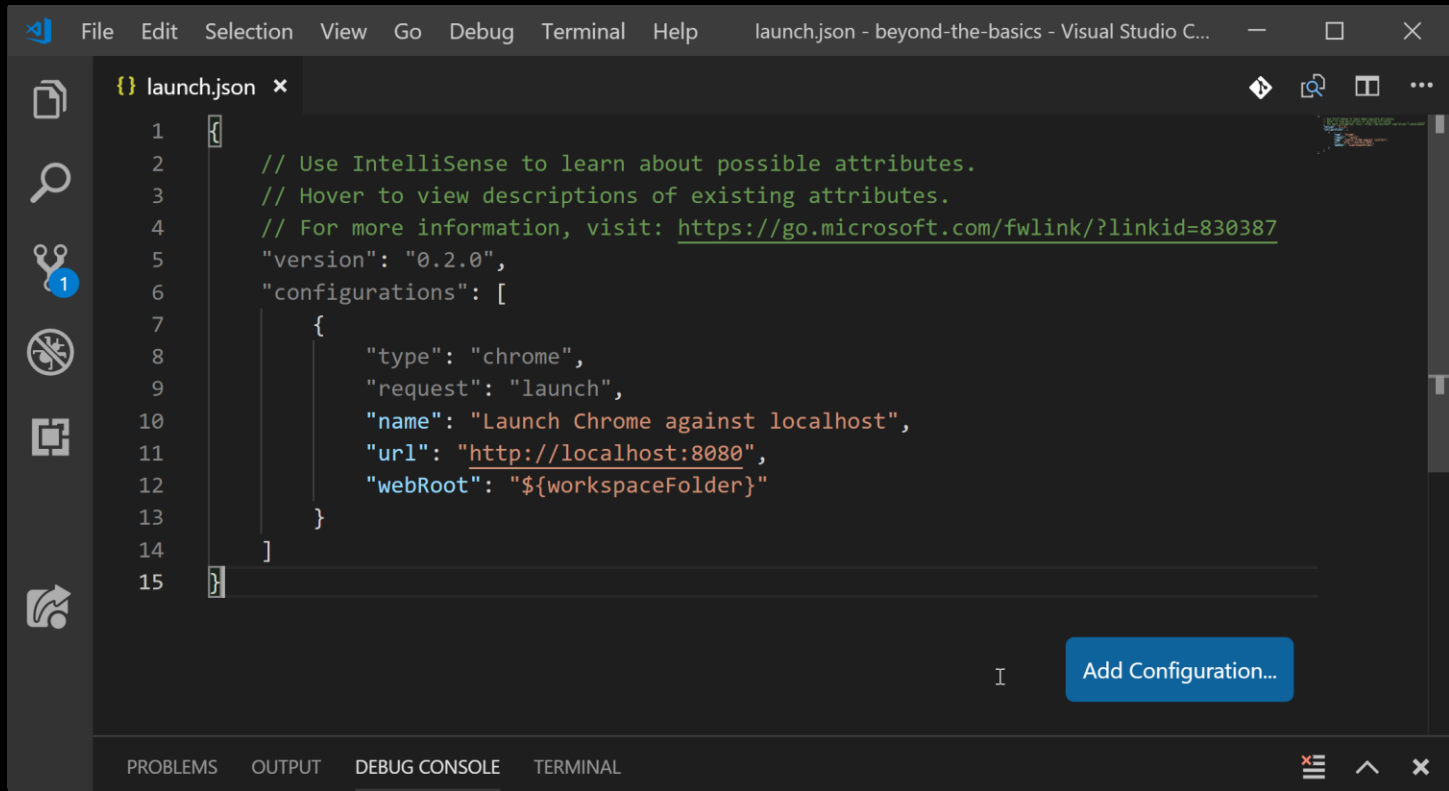
```
module.exports = {  
  configureWebpack: {  
    devtool: 'source-map'  
  }  
}
```

master\* 0 0 0 Live Share Prettier 2







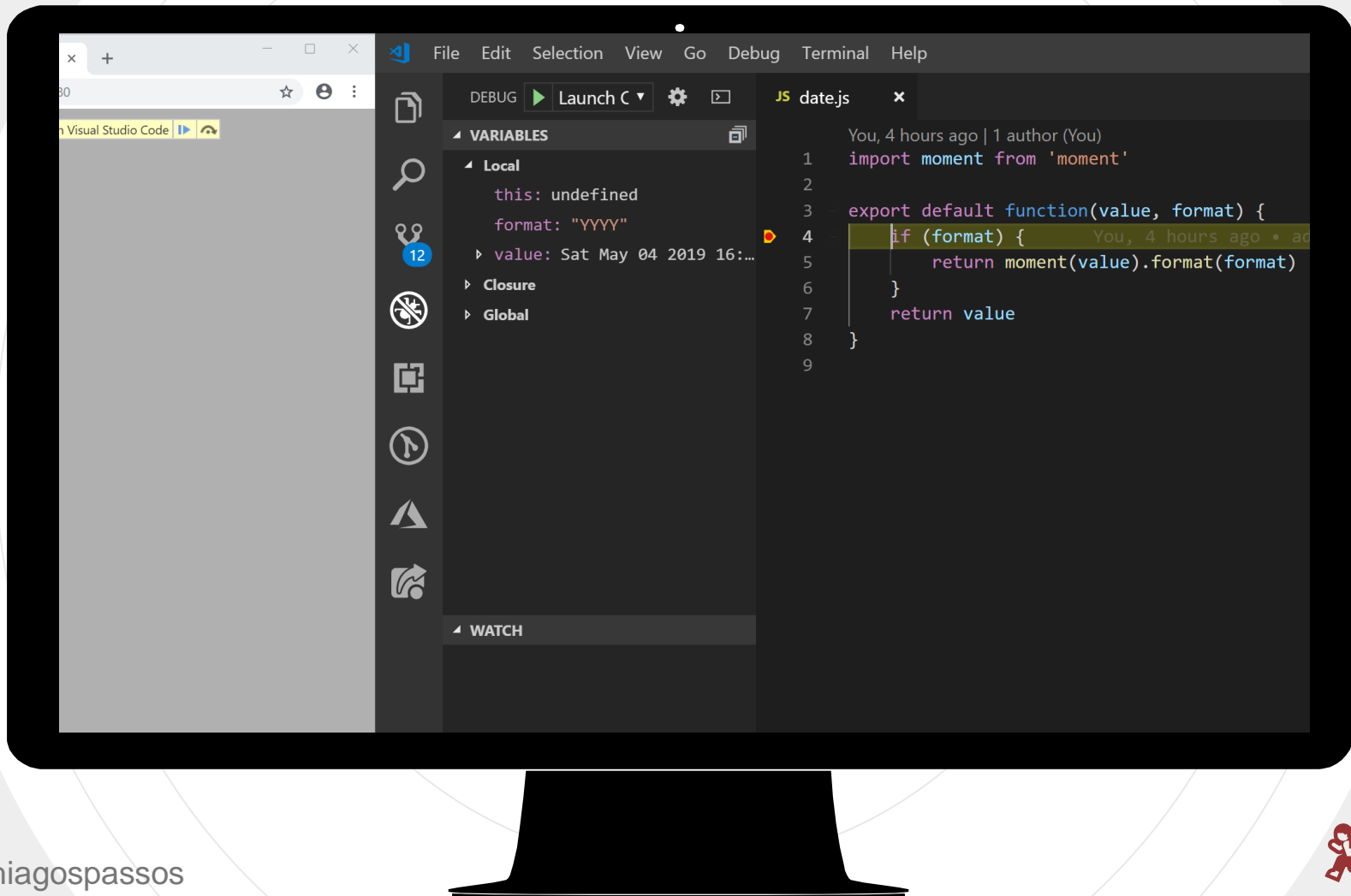


```
1 {
2     // Use IntelliSense to learn about possible attributes.
3     // Hover to view descriptions of existing attributes.
4     // For more information, visit: https://go.microsoft.com/fwlink/?linkid=830387
5     "version": "0.2.0",
6     "configurations": [
7         {
8             "type": "chrome",
9             "request": "launch",
10            "name": "Launch Chrome against localhost",
11            "url": "http://localhost:8080",
12            "webRoot": "${workspaceFolder}"
13        }
14    ]
15 }
```

Add Configuration...







@thiagospassos



# Style Guide

[bit.ly/vue-style-guide](https://bit.ly/vue-style-guide)



@thiagospassos



```
Vue.component('todo-item', {  
  // ...  
})  
  
export default {  
  name: 'TodoItem',  
  // ...  
}
```



Multi-word component names

```
Vue.component('some-comp', {  
  data: function () {  
    return {  
      foo: 'bar'  
    }  
  }  
})
```

```
export default {  
  data () {  
    return {  
      foo: 'bar'  
    }  
  }  
}
```

Component data must be a function

```
props : {  
  status: String  
}
```

```
props: {  
  status: {  
    type: String,  
    required: true,  
    validator: function(value){  
      // ...  
    }  
  }  
}
```

Props should be detailed

```
<ul>
  <li
    v-for="todo in todos"
    :key="todo.id">
    {{ todo.text }}
  </li>
</ul>
```

Always use key for v-for

```
<ul v-if="shouldShowUsers">
  <li
    v-for="user in users"
    :key="user.id">
    {{ user.name }}
  </li>
</ul>
```

Avoid v-if with v-for in the same element

```
<style scoped>
.button {
  border: none;
  border-radius: 2px;
}

.button-close {
  background-color: red;
}
</style>
```

Component style scoping

# Your Turn!

Follow the instructions to complete the code exercises and challenges



Workshop Guide  
[bit.ly/masteringvuejs](https://bit.ly/masteringvuejs)



Workshop Code  
[bit.ly/vuejs-oslo-2019](https://bit.ly/vuejs-oslo-2019)



@thiagospassos



# Mastering Vue.js Dynamic Forms & Validation



# Agenda

Best Practices

Inline Forms & Validation

Complex Forms & Validation

# Agenda

Best Practices

Inline Forms & Validation

Complex Forms & Validation


# Best Practices

- Forms and validation are arguably the most important interaction between your application and your users
- Let's examine some best practices



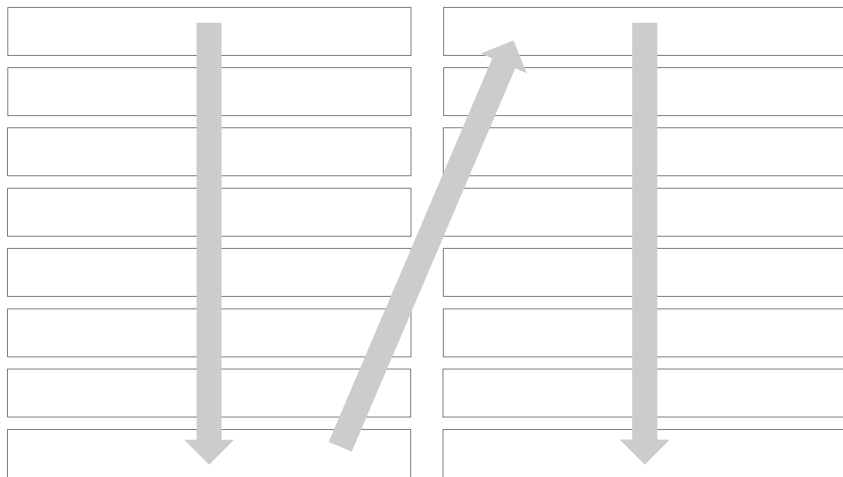


# 1. Single Column Layout

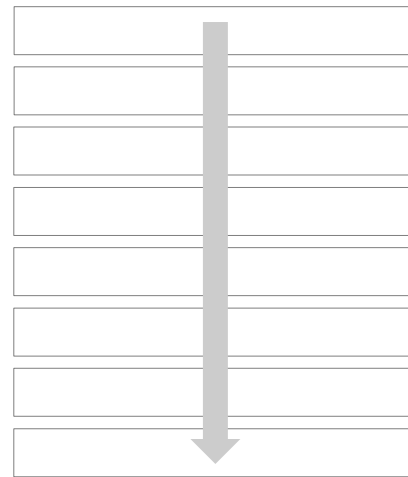
- Prefer single column layouts over multi column layouts
  - Single column layouts have a clear path to completion
  - Multi column layouts require the user to scan the form to determine the path to completion
- 



Bad Example 👎



Good Example 👍



## 2. Organise Fields Logically

- Order fields logically from the user's perspective, not the application or database perspective
- Order fields from easiest to hardest. Don't make it hard just to get started!



## Bad Example

Address

Full Name

Email

## Good Example

Full Name

Email

Address



## Bad Example

Credit Card Number

Expiry

CCV

Address

Full Name

## Good Example

Full Name

Address

Credit Card Number

Expiry

CCV







# 3. Group Related Fields

- Grouping related fields into logical sets will improve layout and user comprehension



## Bad Example

Full Name

Address

Credit Card Number

Expiry

CCV

## Good Example

### Personal

Full Name

### Address

Line 1

### Payment

Credit Card Number

Expiry

CCV



## 4. Inline Validation

- Use inline validation to provide fast and effective feedback
- For complex forms, avoid waiting until the user submits the form
- Ideally validate after input, not on each keystroke



## Bad Example

Full Name

Email

Credit Card Number

Expiry

CCV

You really messed up!

- Full Name is required.
- Email is invalid.
- Credit card number is required.
- Expiry is required.
- CCV is required.

## Good Example

Full Name

Required field

Email

Credit Card Number

Expiry

CCV





## 5. Clear Actions

- Ensure that the form actions are clearly visible and well-defined
- Emphasis should be placed on the primary action



## Bad Example

Email

Password

Register

Cancel

*(evil example)*

Email

Password

Cancel

Register

## Good Example

Email

Password

Register

Cancel

*(or)*

Register

Cancel



# Agenda

Best Practices

Inline Forms & Validation

Complex Forms & Validation

# Demo

- Create a form for inline editing of categories
- Add validation support to the new categories form





# Agenda

Best Practices

Inline Forms & Validation

Complex Forms & Validation

# Demo

- Create a form for editing products
- Implement complex validation rules with Vuelidate



# Your Turn!

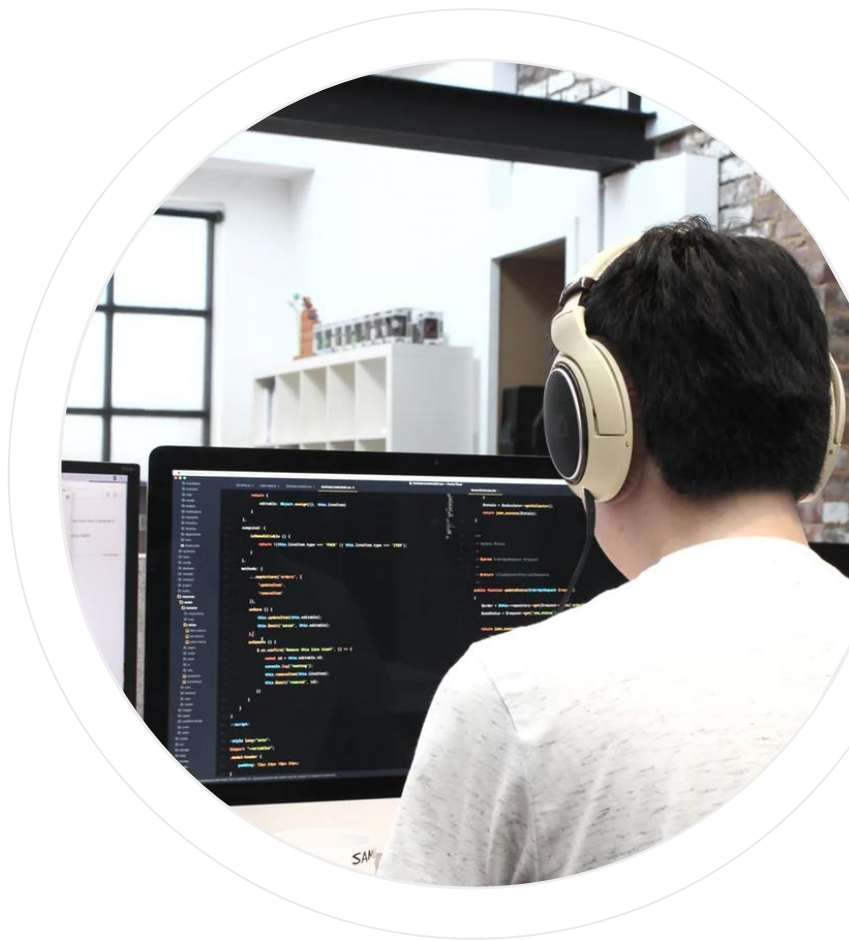
Follow the instructions to complete the code exercises and challenges



Workshop Guide  
[bit.ly/masteringvuejs](https://bit.ly/masteringvuejs)



Workshop Code  
[bit.ly/vuejs-oslo-2019](https://bit.ly/vuejs-oslo-2019)



@thiagospassos



# Mastering Vue.js **Automated Testing**

# Agenda

Why automated testing?

Test types and frameworks

Testing our components

Testing our solution

# Agenda

Why automated testing?

Test types and frameworks

Testing our components

Testing our solution



**1**

**Improves  
confidence and  
removes fear of  
changing code**



2

# Code Quality

“Test a single concept in each test  
function” (Uncle Bob)



The image features a large white circle centered on a black background. To the left of the white circle, there is a series of overlapping circles in shades of gray, with the number '3' in white. To the right of the white circle, there is a series of concentric white circles. The text 'Act as documentation' is centered within the white circle.

**3**

**Act as  
documentation**

# Agenda

Why automated testing?

Test types and frameworks

Testing our components

Testing our solution

# Unit Tests

```
describe('HelloWorld.vue', () => {  
  it('renders props.msg when passed',  
    const msg = 'Hello World'  
    const wrapper = shallowMount(HelloWorld,  
      { propsData: { msg } }  
    )  
    expect(wrapper.text()).toMatch(msg)  
  })  
})  
  
describe('AddressForm.vue', () => {  
  it('it should fail if no address'  
    // ...  
  })  
})
```



# Integration Tests



@thiagospassos

# Tools



# Agenda

Why automated testing?

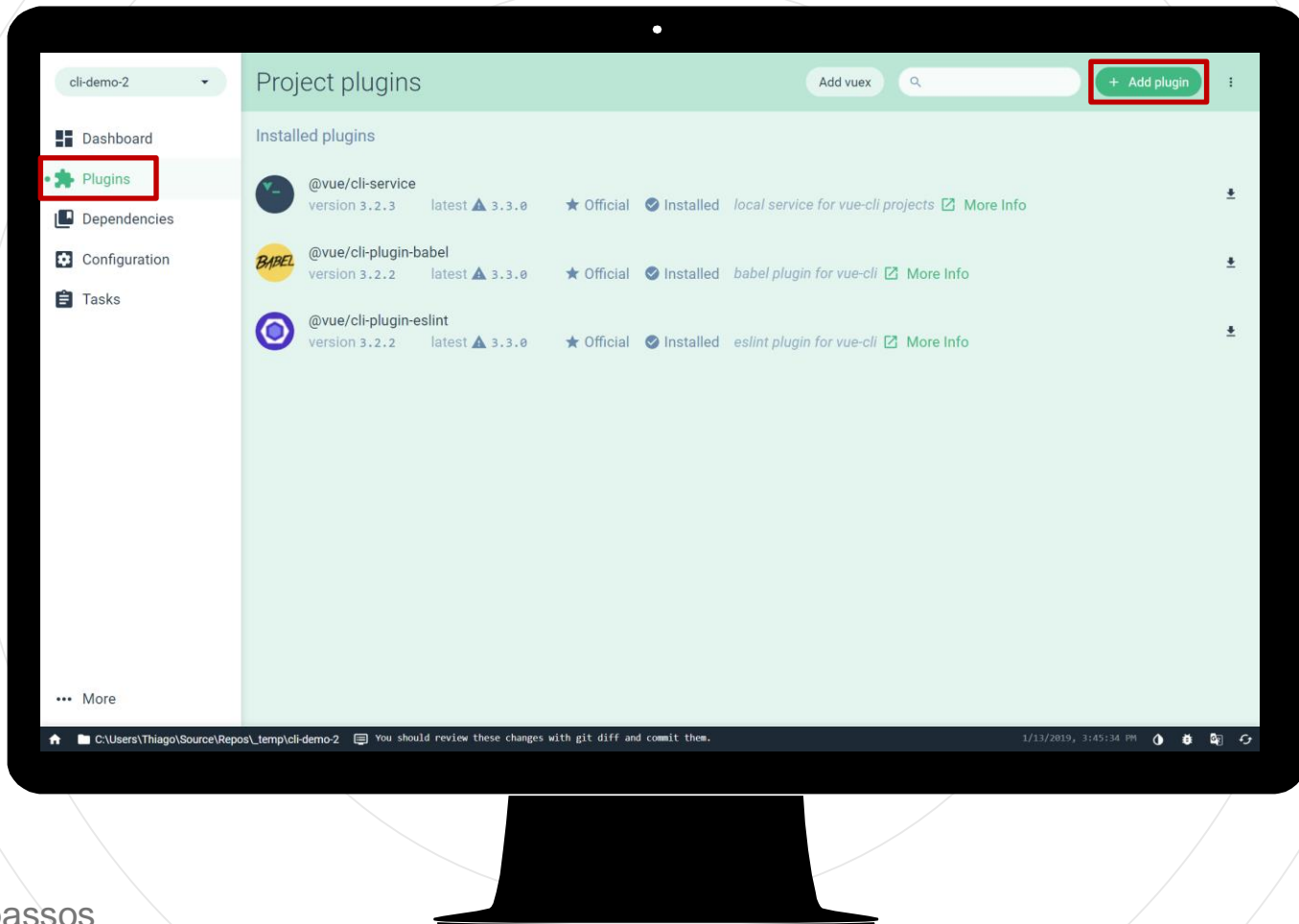
Test types and frameworks

Testing our components

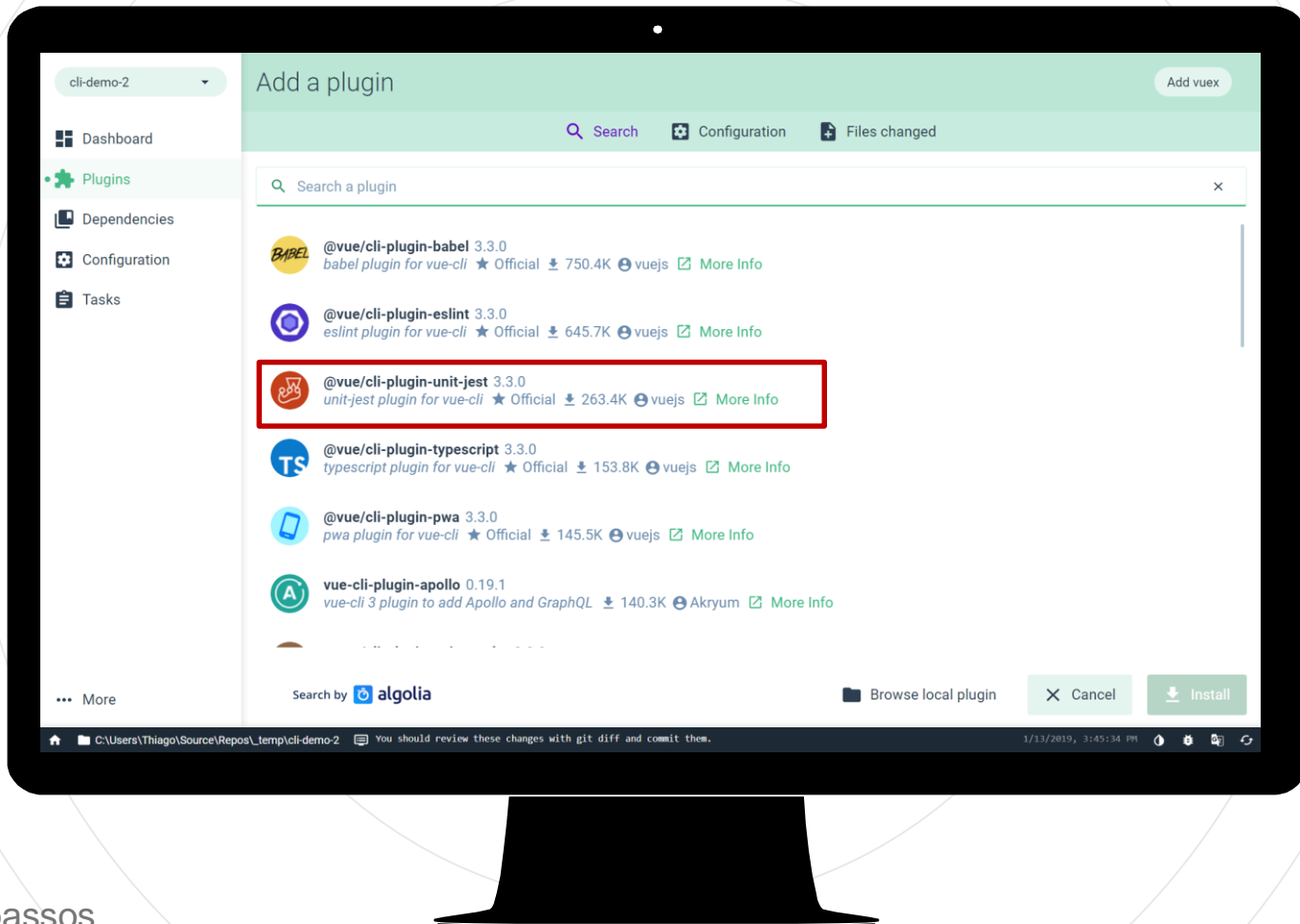
Testing our solution

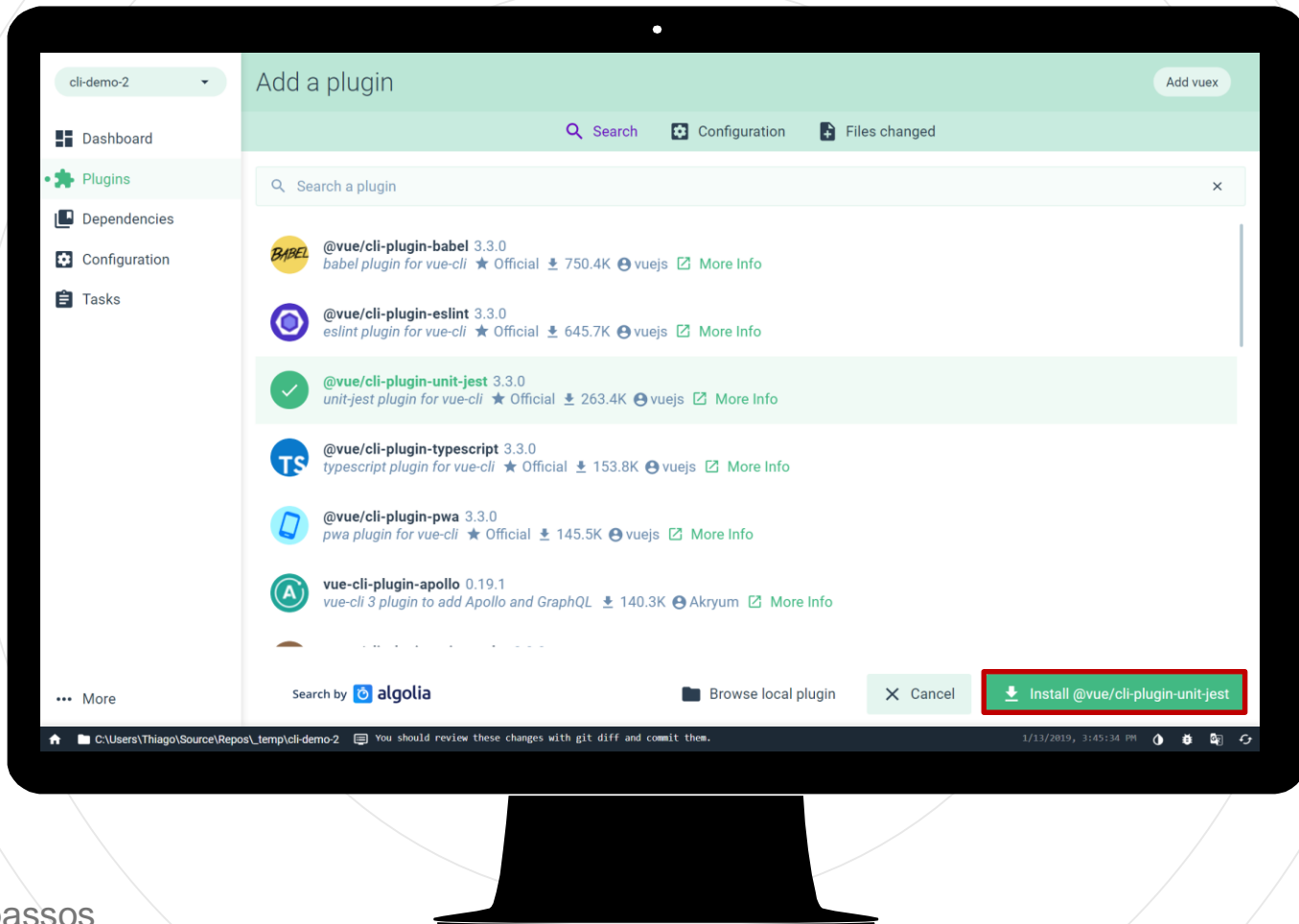
# Unit Tests

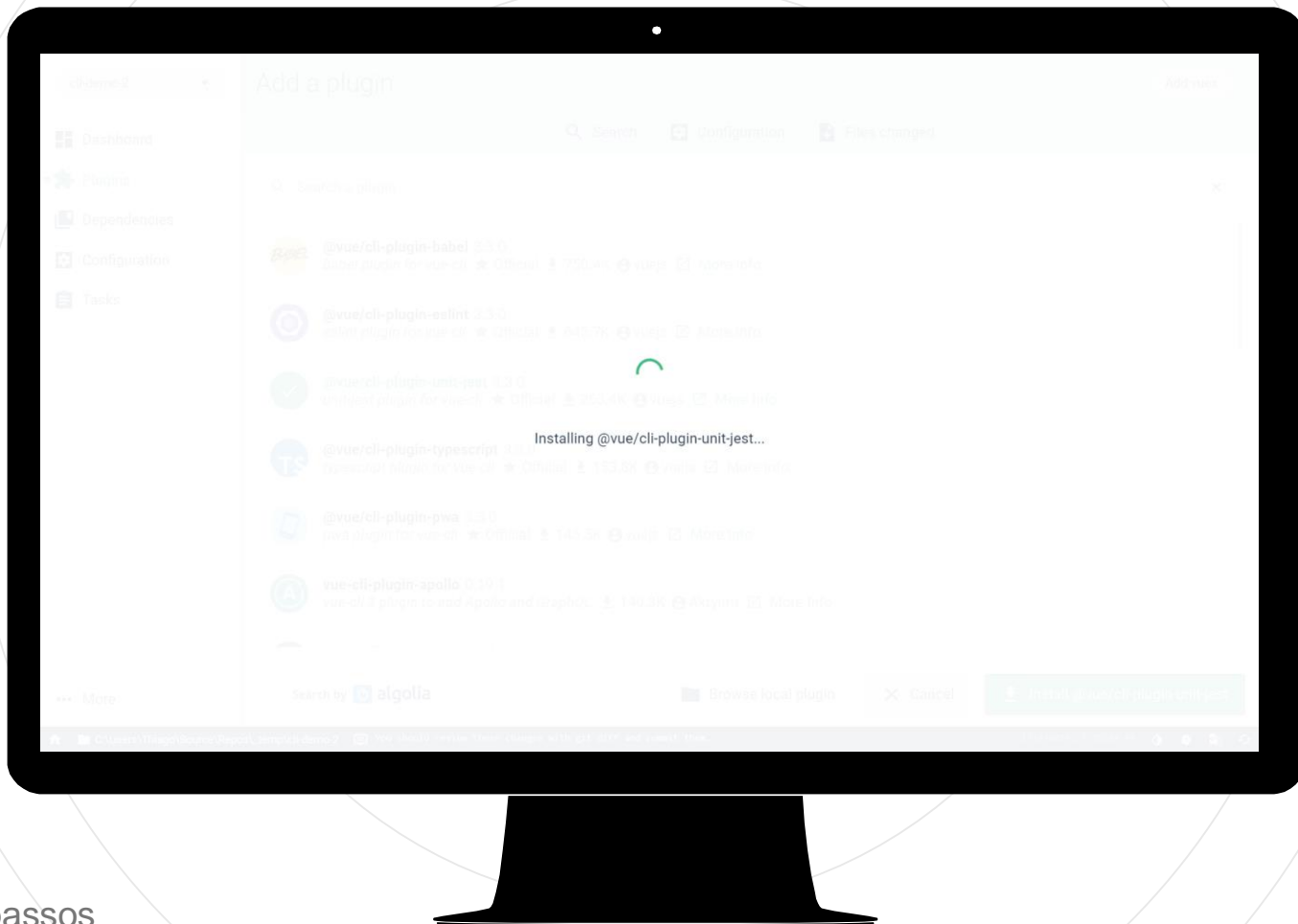




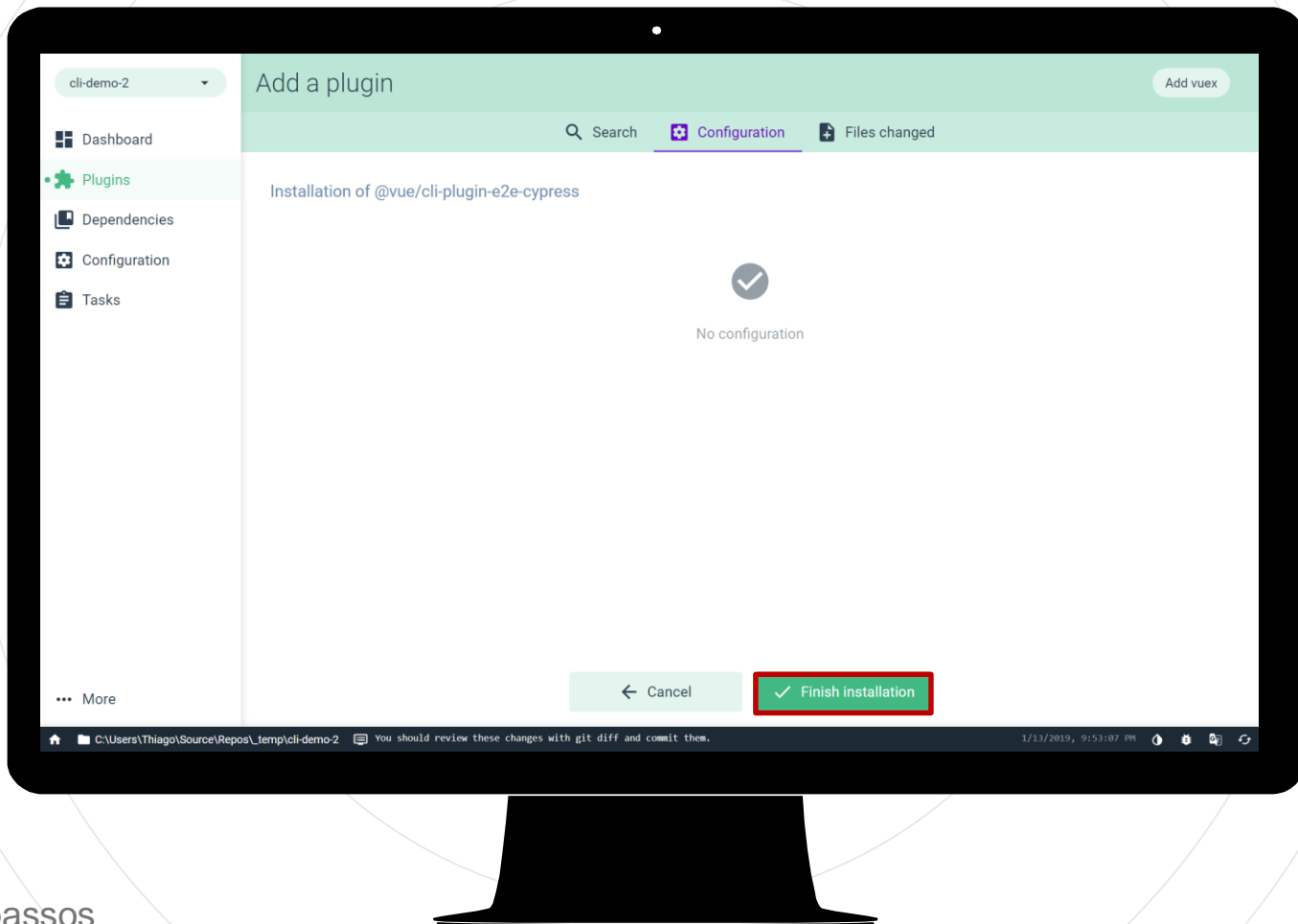


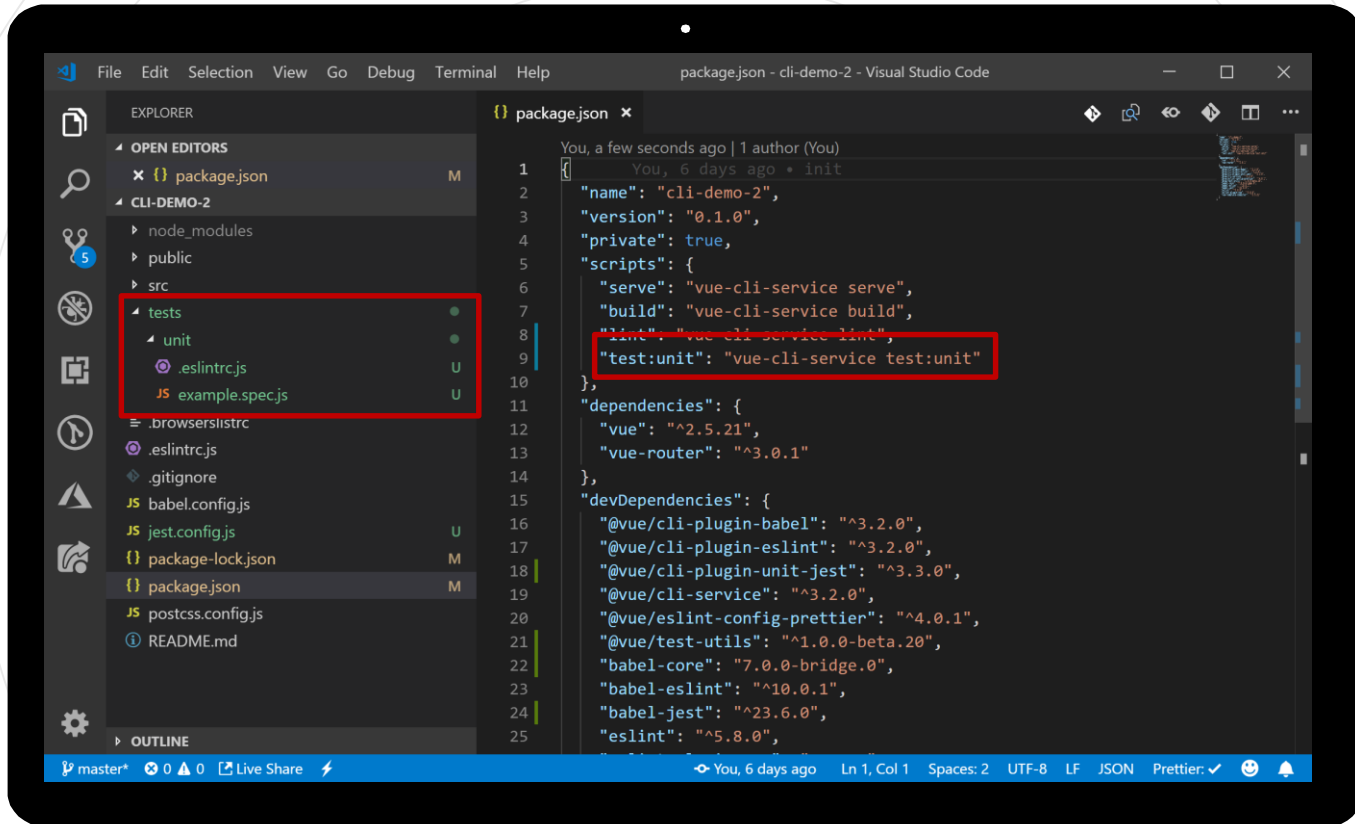






@thiagospassos





# Agenda

Why automated testing?

Test types and frameworks

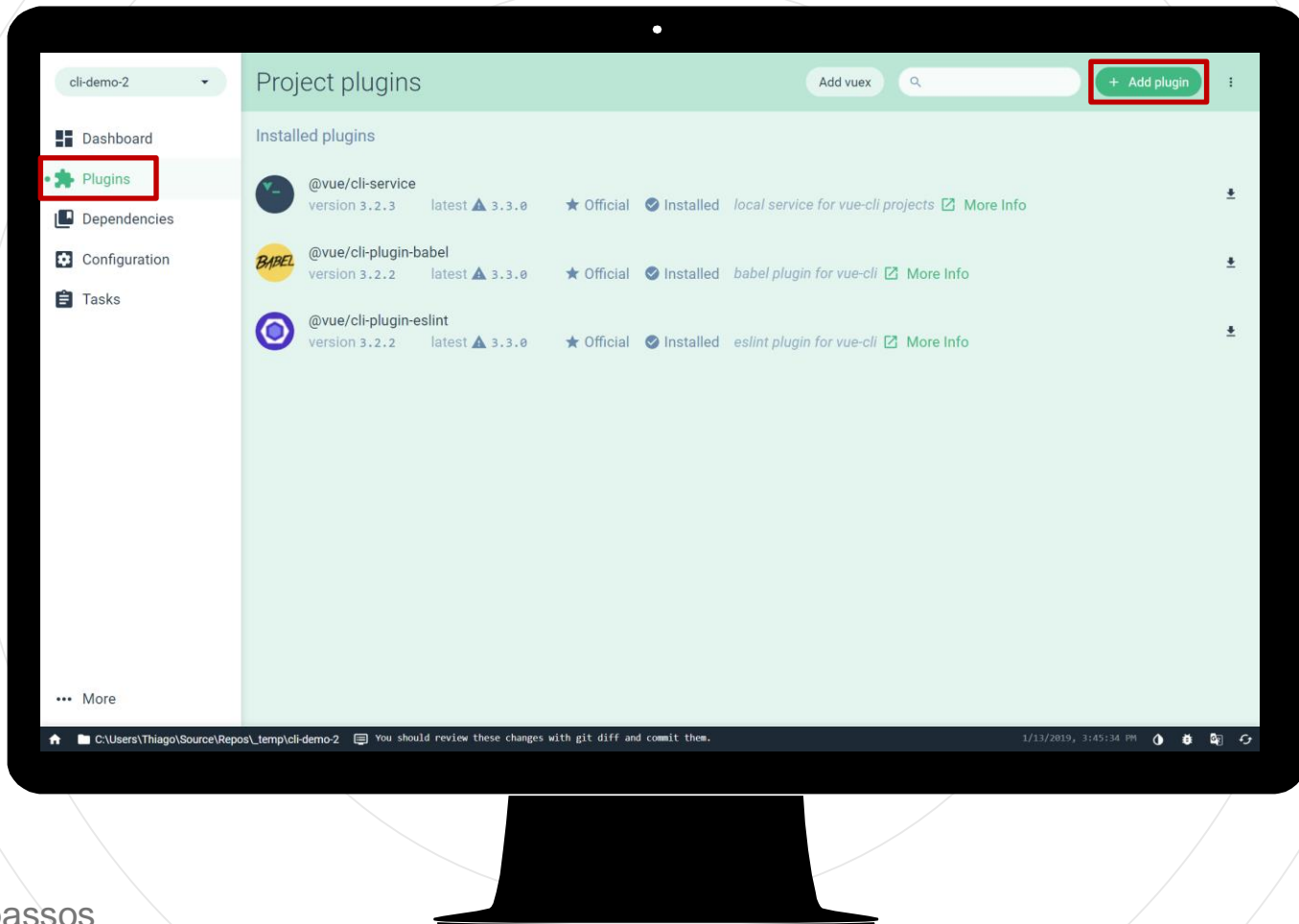
Testing our components

Testing our solution

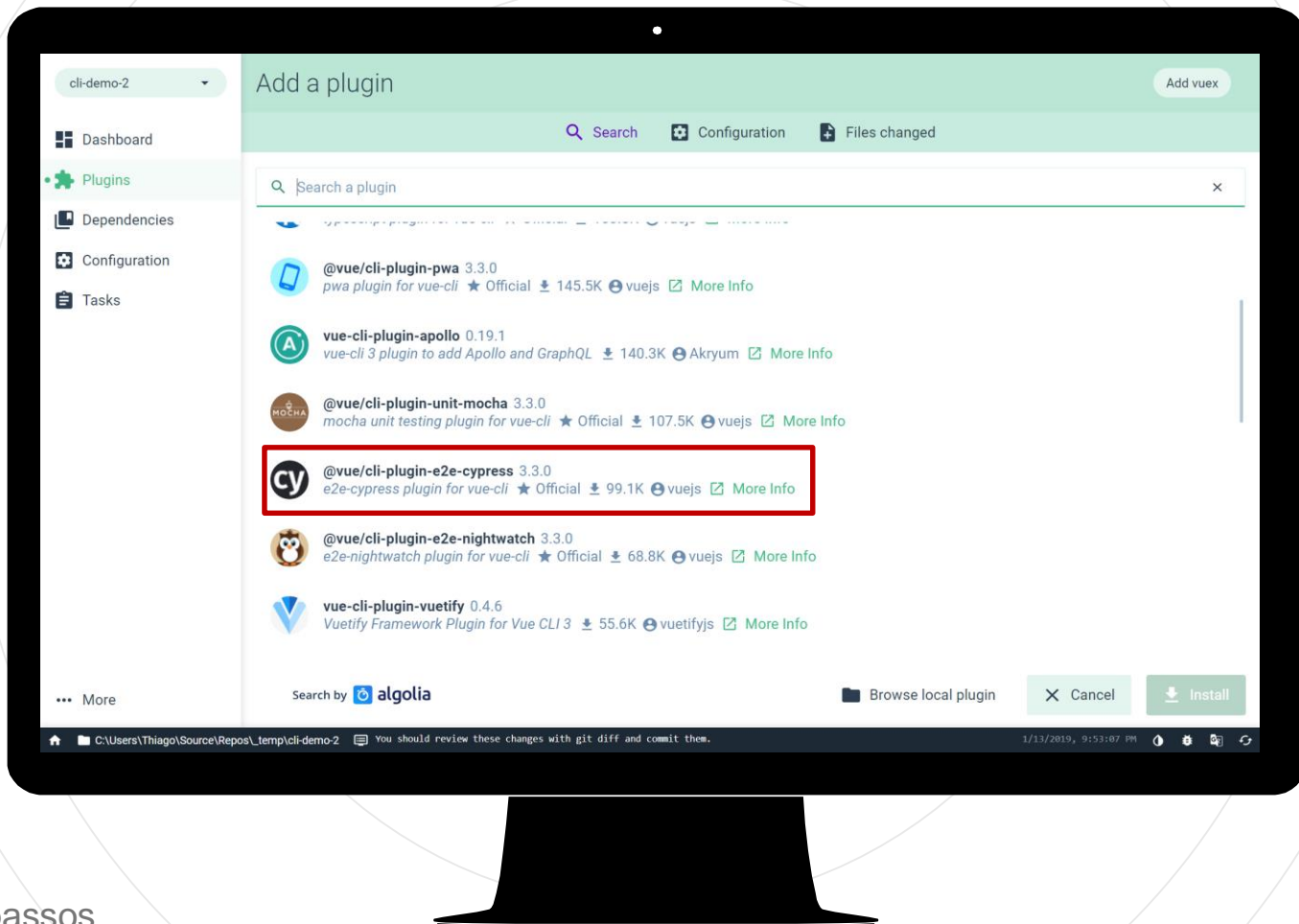
# Integration Tests

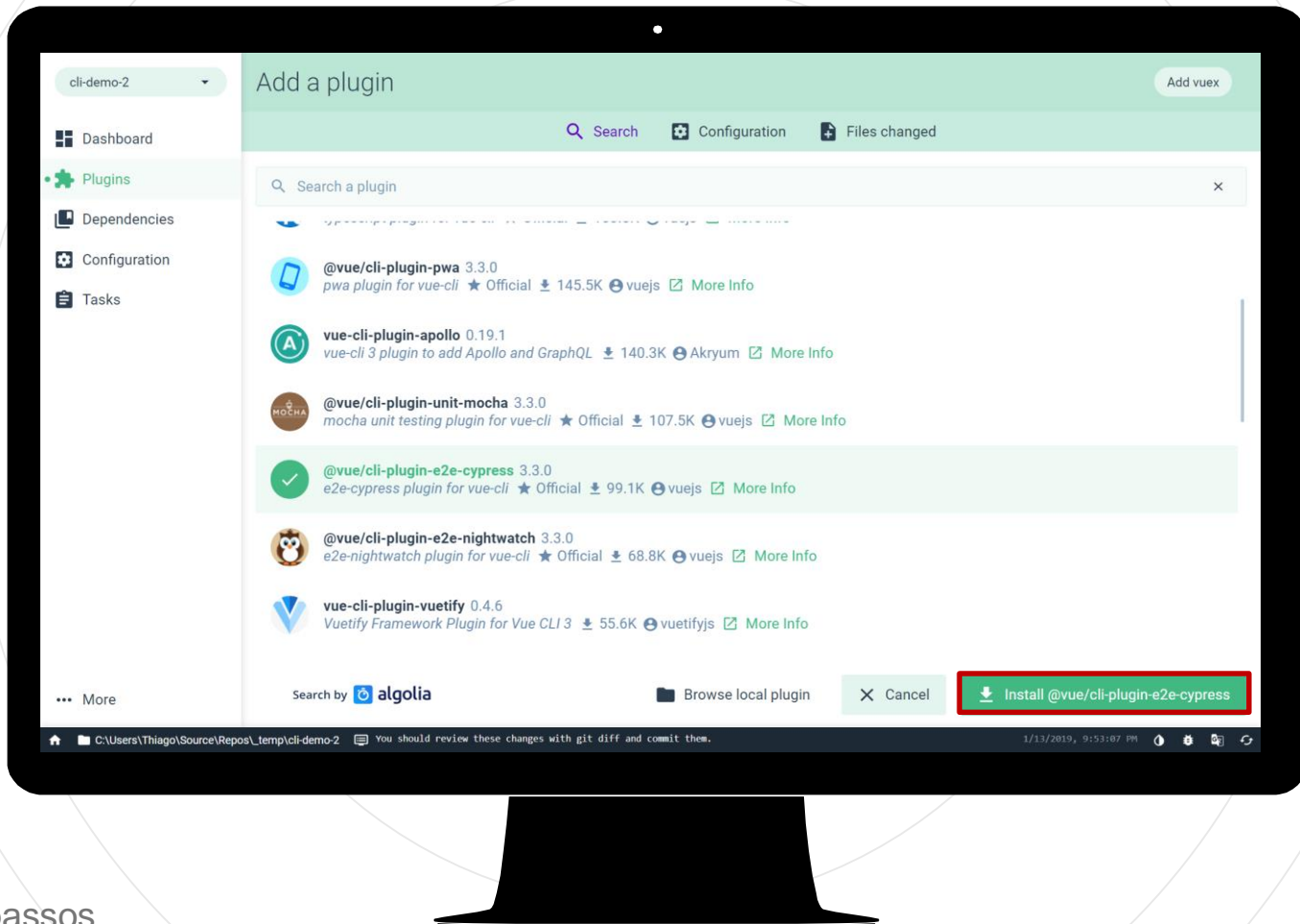


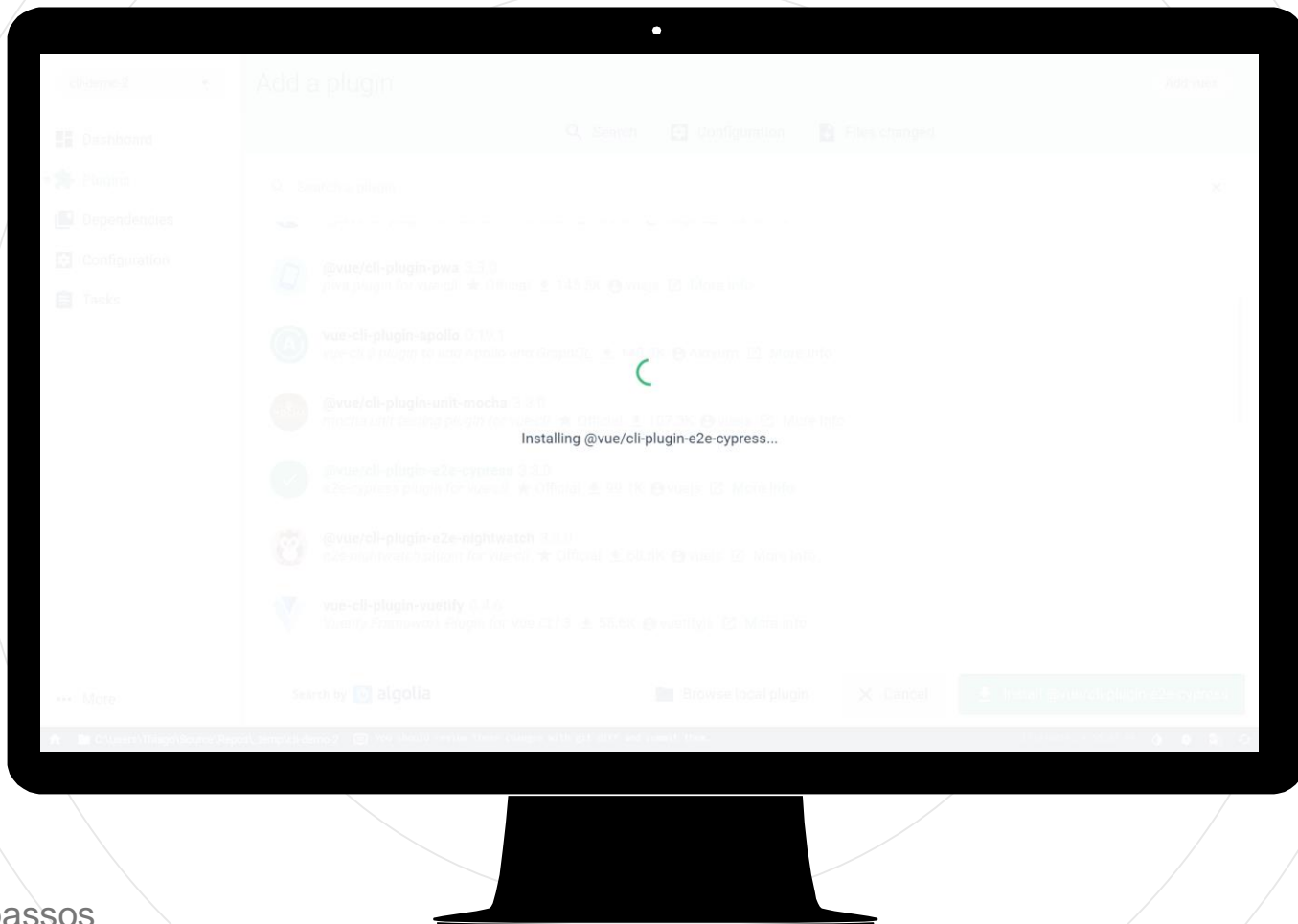
@thiagospassos

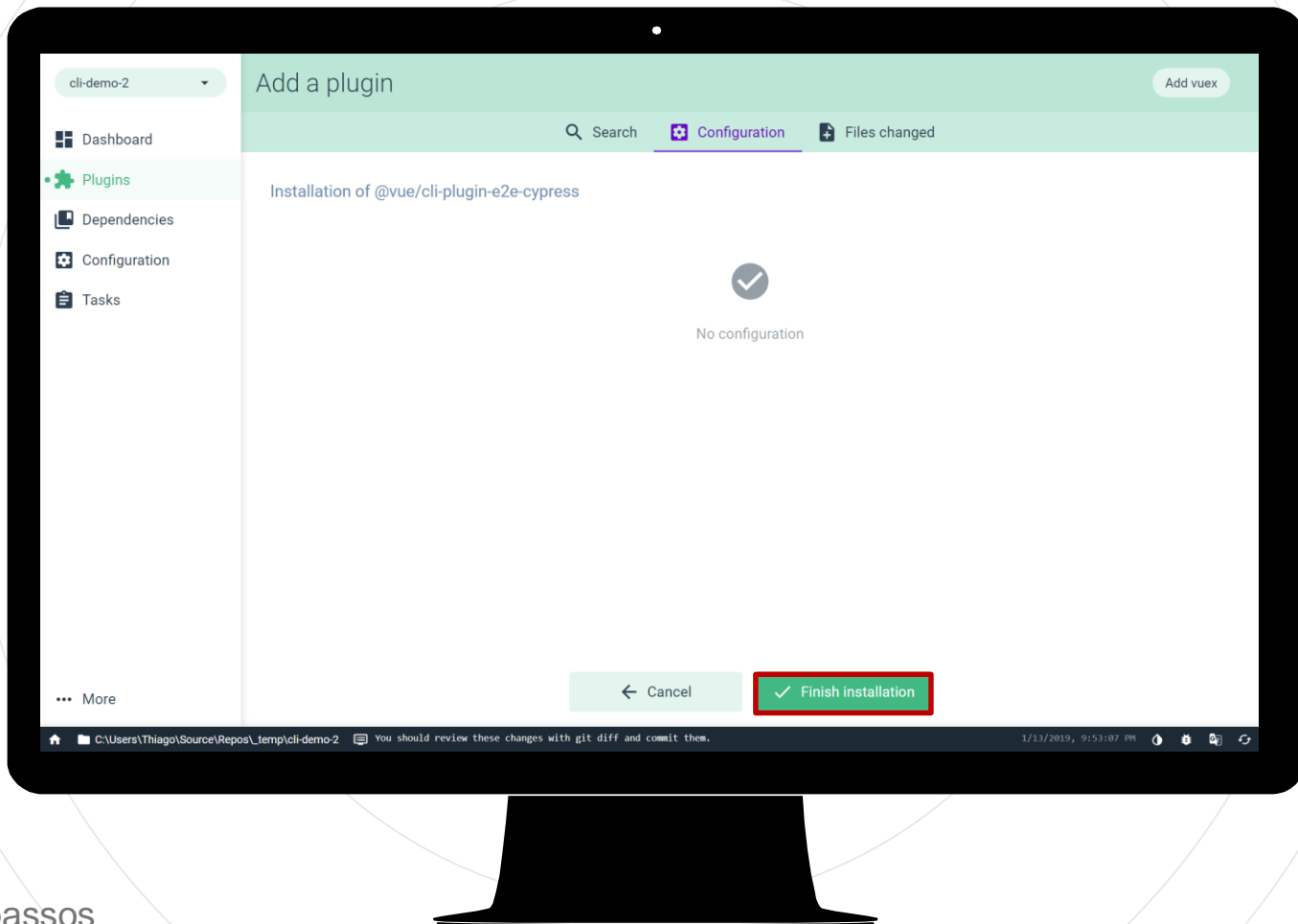


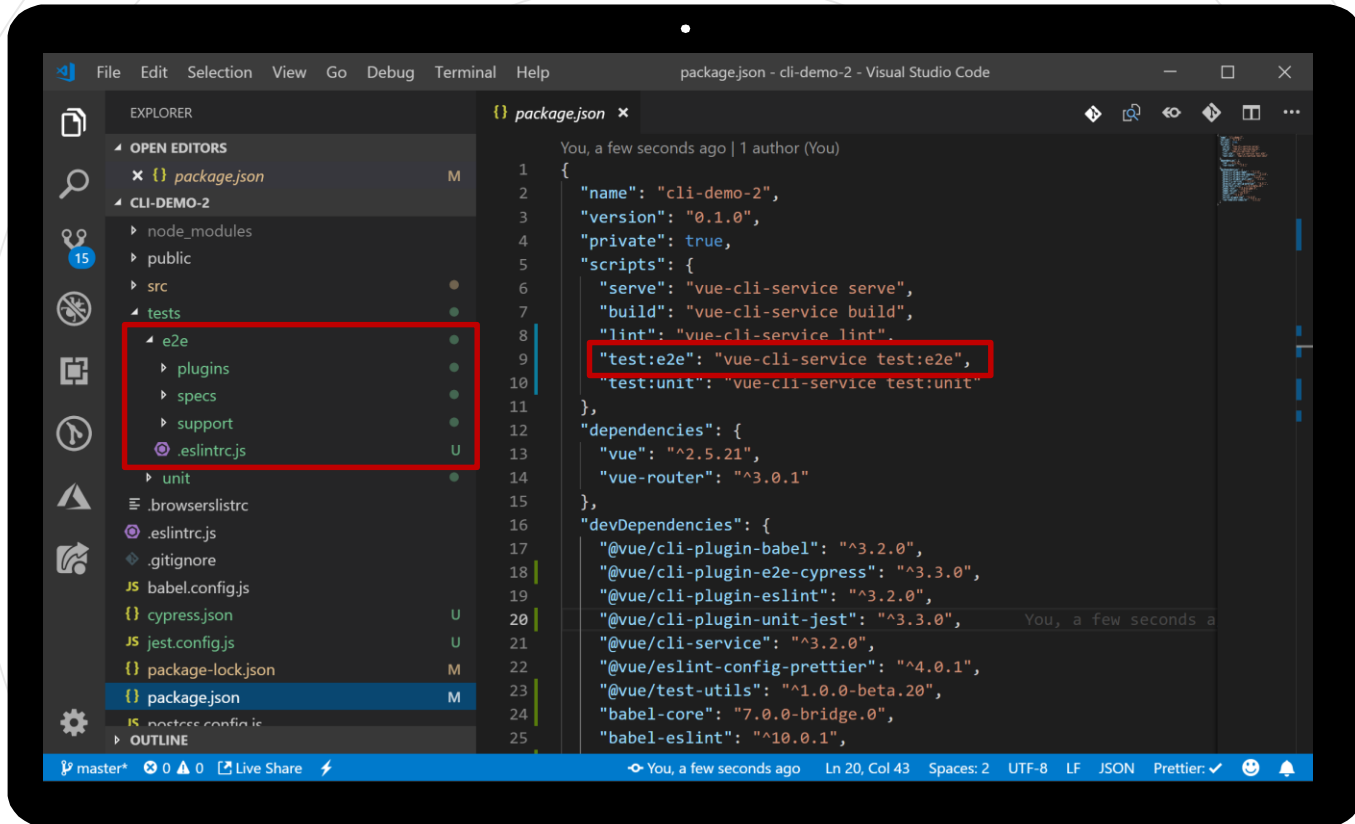












# Your Turn!

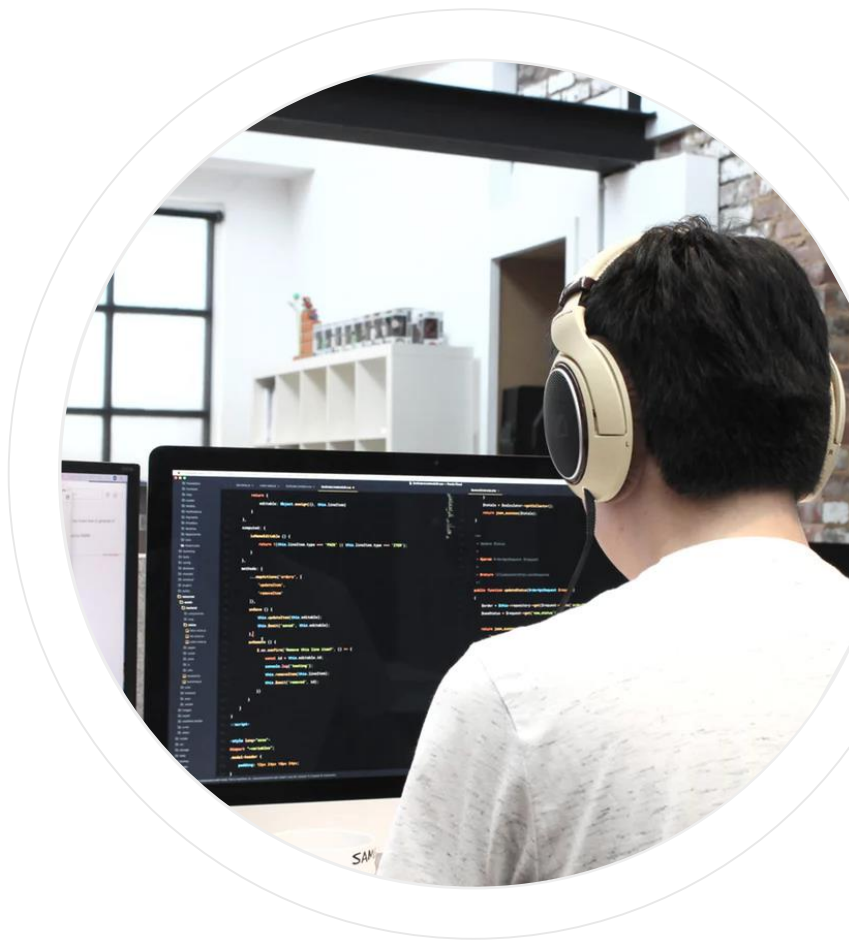
Follow the instructions to complete the code exercises and challenges



Workshop Guide  
[bit.ly/masteringvuejs](https://bit.ly/masteringvuejs)



Workshop Code  
[bit.ly/vuejs-oslo-2019](https://bit.ly/vuejs-oslo-2019)



@thiagospassos



# Vue.js

## Consuming APIs

# Agenda

Getting Started

Configuration

Interceptors

Put all together



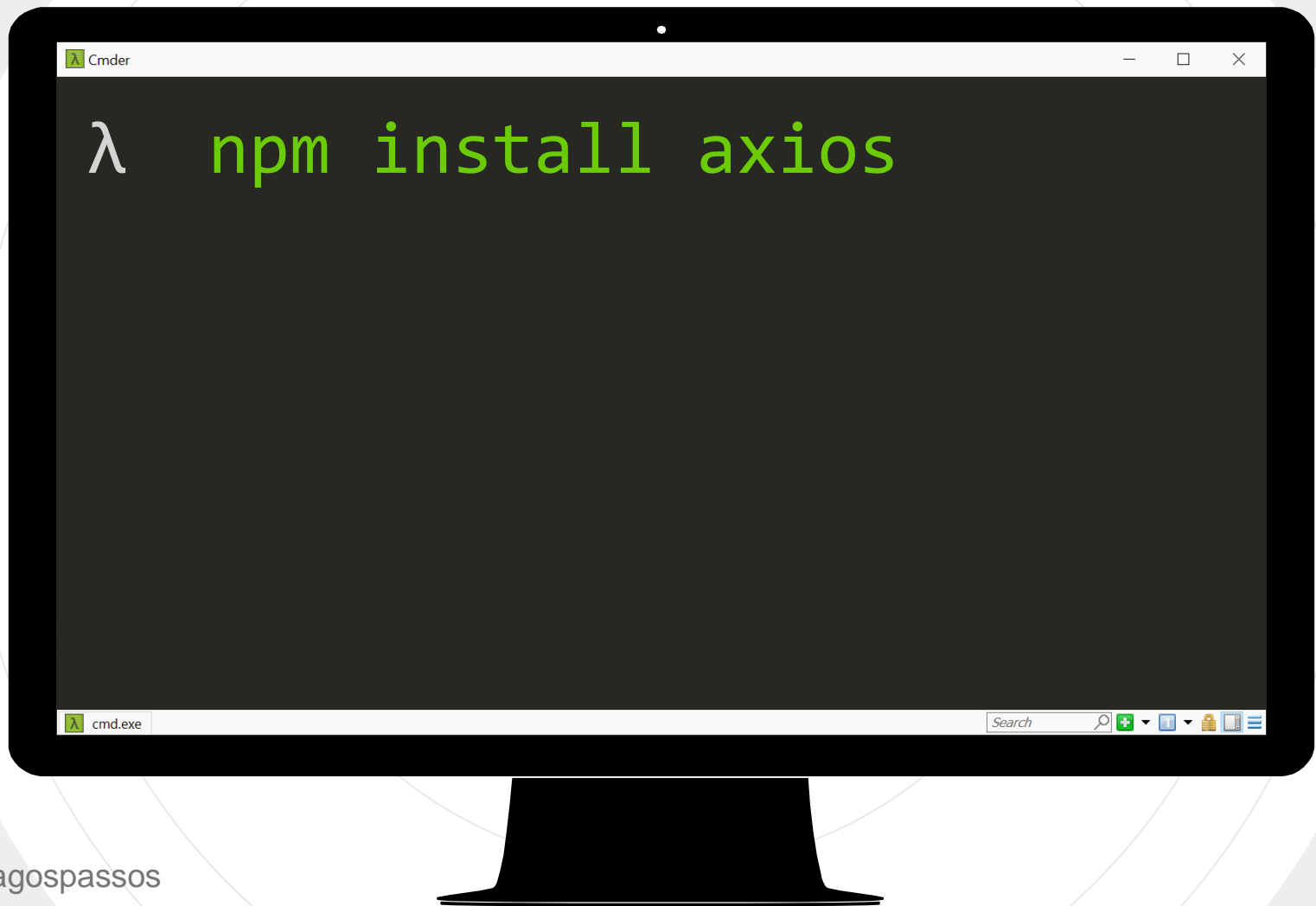
# Agenda

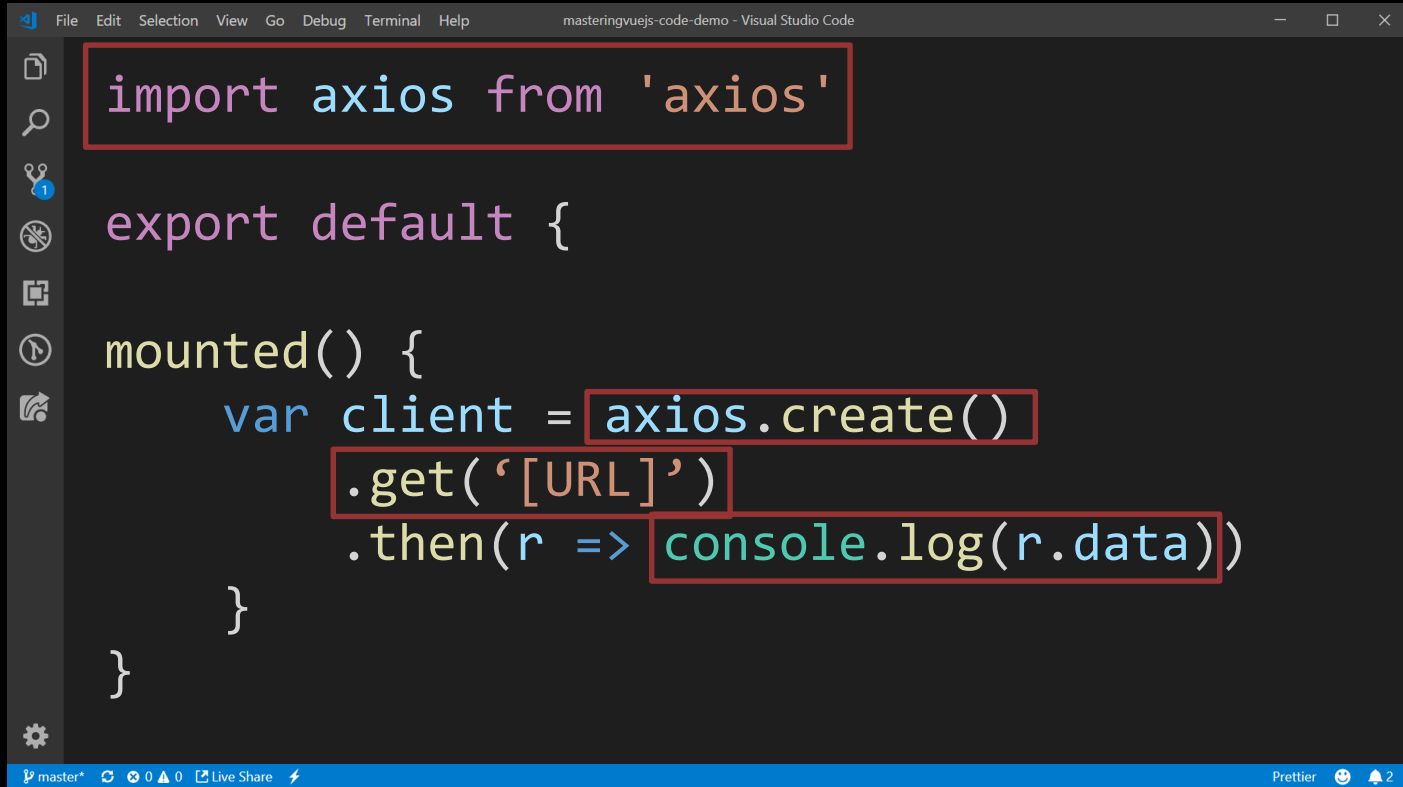
Getting Started

Configuration

Interceptors

Put all together





```
File Edit Selection View Go Debug Terminal Help masteringvuejs-code-demo - Visual Studio Code

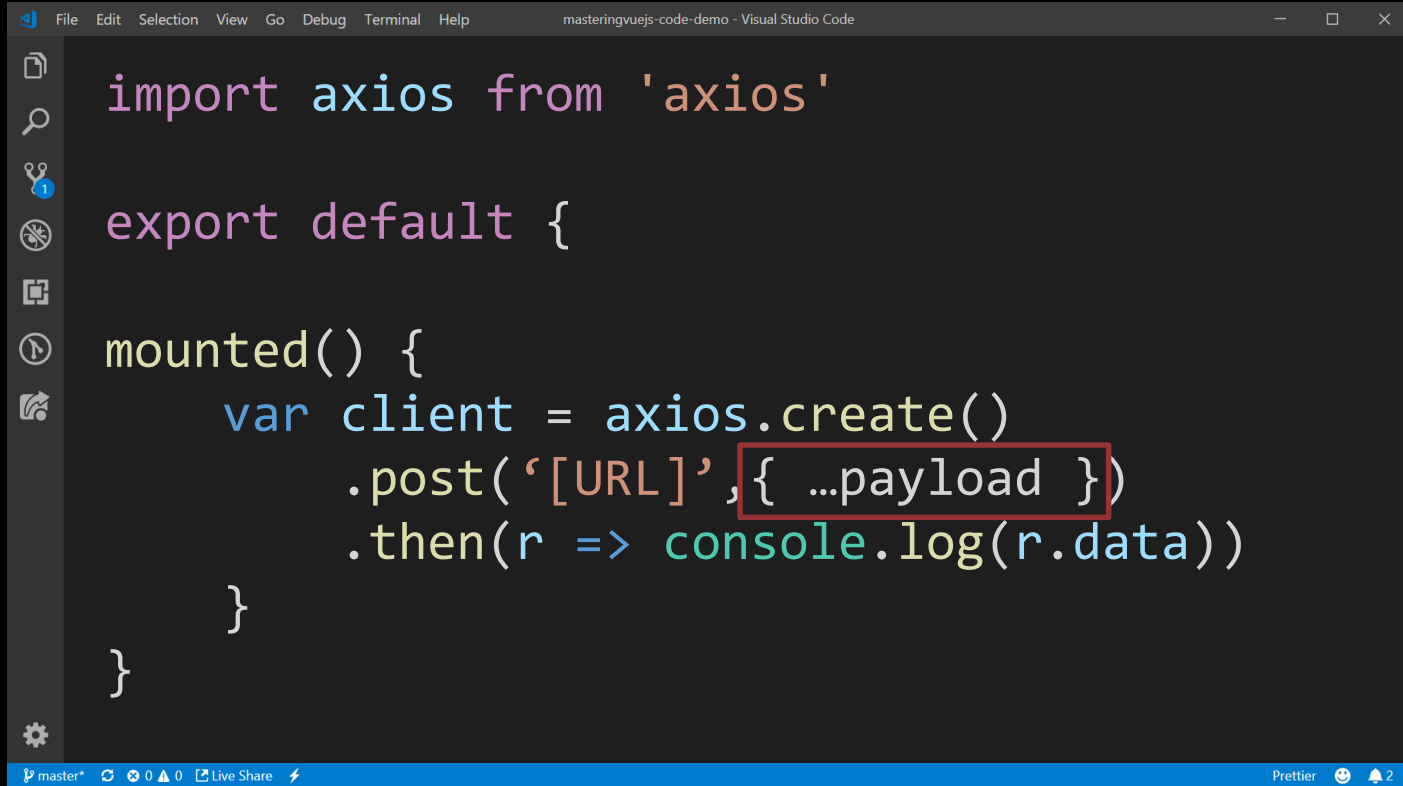
import axios from 'axios'

export default {

  mounted() {
    var client = axios.create()
      .get('[URL]')
      .then(r => console.log(r.data))
  }
}
```

master\* 0 0 0 Live Share Prettier 2





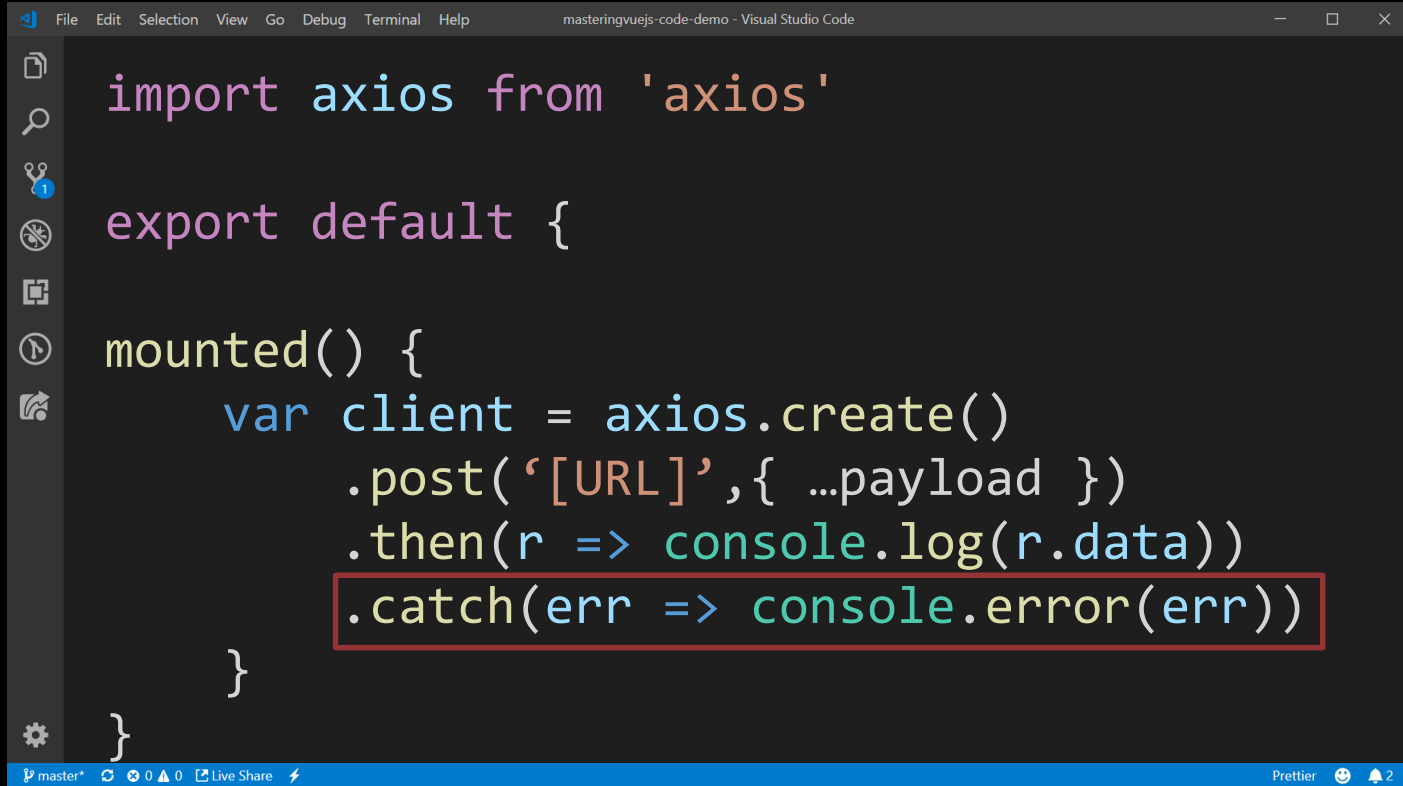
```
import axios from 'axios'

export default {

  mounted() {
    var client = axios.create()
    .post('[URL]', { ...payload })
    .then(r => console.log(r.data))
  }
}
```

Visual Studio Code interface details: The title bar reads 'masteringvuejs-code-demo - Visual Studio Code'. The left sidebar contains icons for Explorer, Search, Source Control, Run and Debug, Extensions, and Settings. The status bar at the bottom left shows 'master\*' and icons for error, warning, and info counts (0 each), along with a 'Live Share' icon. The status bar at the bottom right shows 'Prettier' and notification icons.





```
import axios from 'axios'

export default {

  mounted() {
    var client = axios.create()
      .post('[URL]', { ...payload })
      .then(r => console.log(r.data))
      .catch(err => console.error(err))
  }
}
```

Visual Studio Code interface details: The title bar reads 'masteringvuejs-code-demo - Visual Studio Code'. The left sidebar contains icons for Explorer, Search, Source Control, Run and Debug, Extensions, and Live Share. The status bar at the bottom left shows 'master\*' and icons for error, warning, and Live Share. The status bar at the bottom right shows 'Prettier', a smiley face icon, and a notification bell with the number '2'.



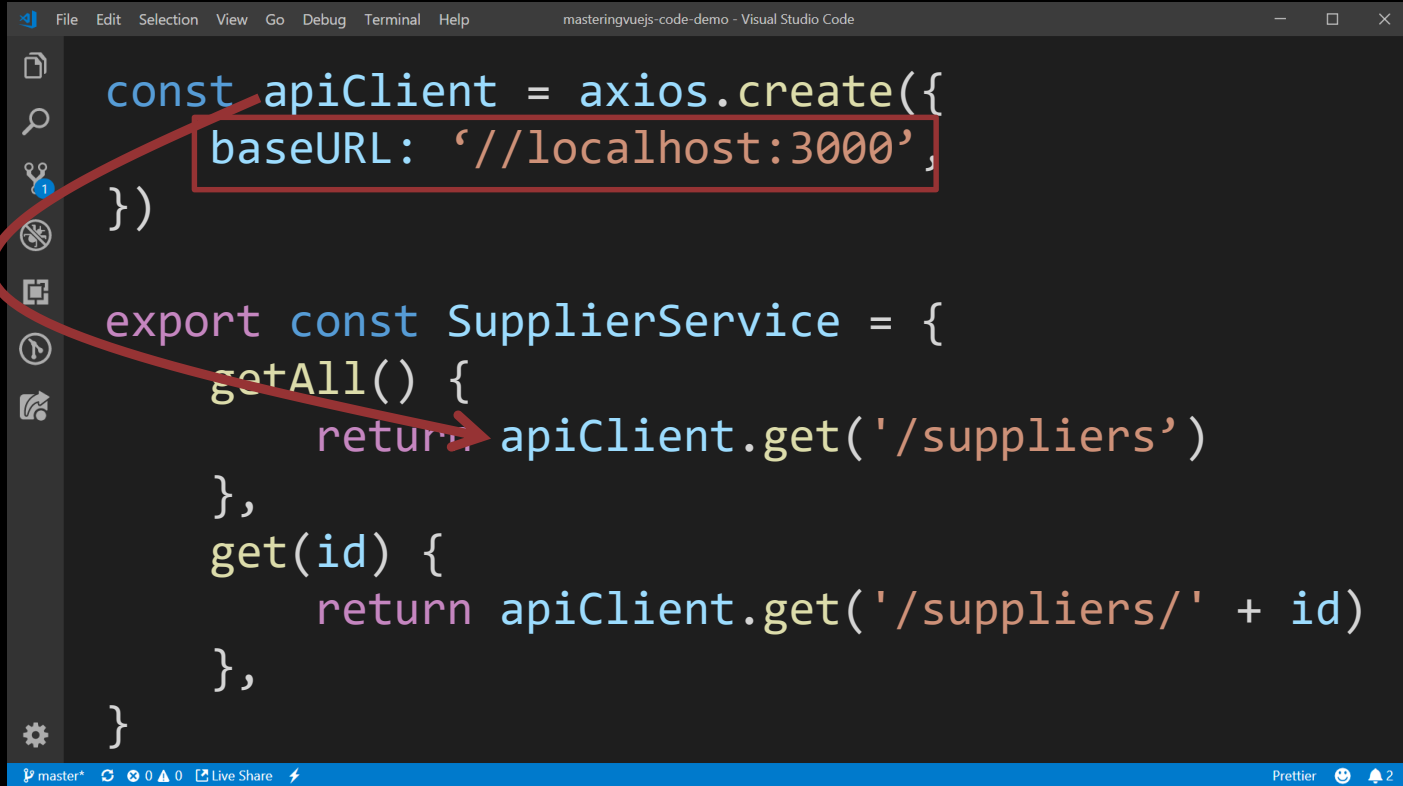
# Agenda

Getting Started

Configuration

Interceptors

Put all together



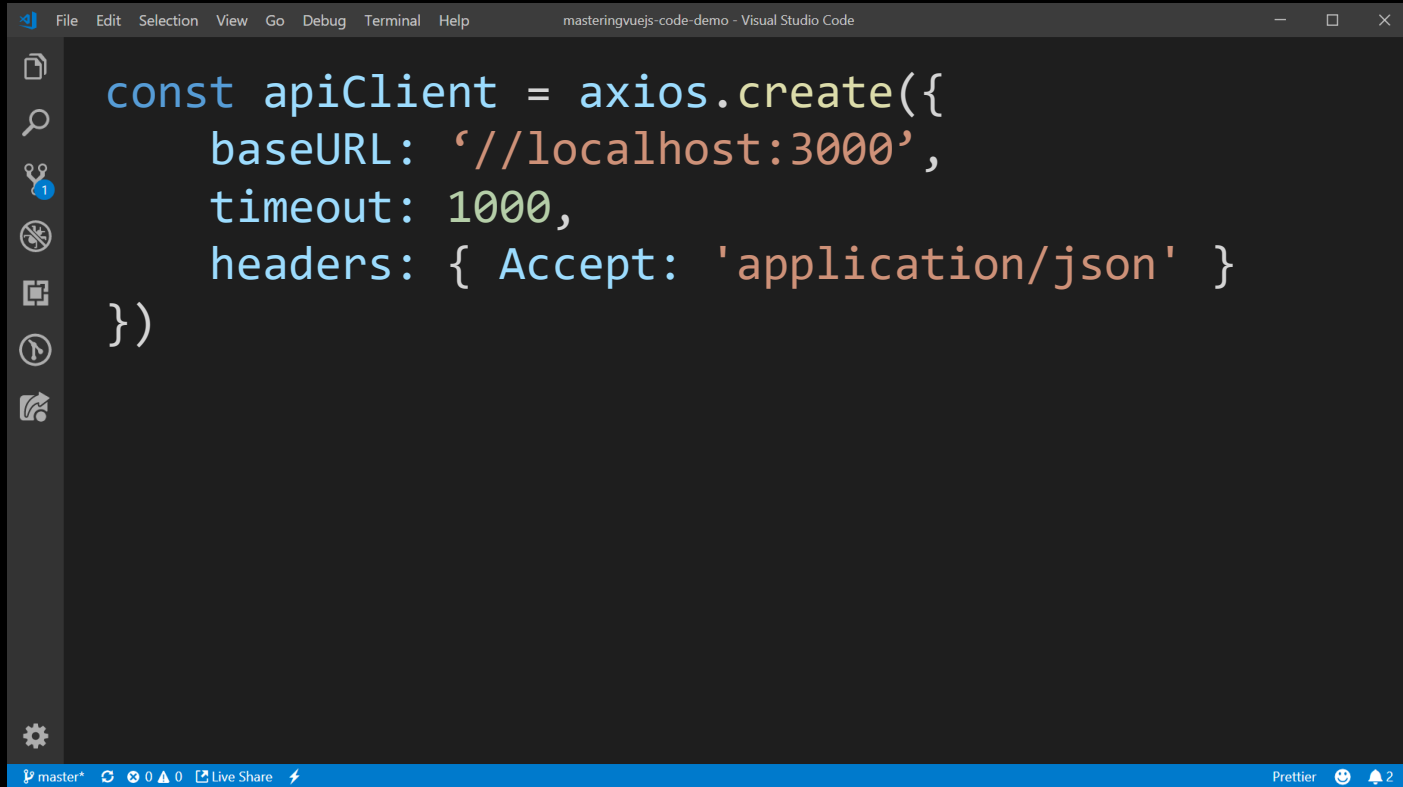
```
File Edit Selection View Go Debug Terminal Help masteringvuejs-code-demo - Visual Studio Code

const apiClient = axios.create({
  baseURL: 'localhost:3000',
})

export const SupplierService = {
  getAll() {
    return apiClient.get('/suppliers')
  },
  get(id) {
    return apiClient.get('/suppliers/' + id)
  },
}
```

master\* 0 0 0 Live Share Prettier 2

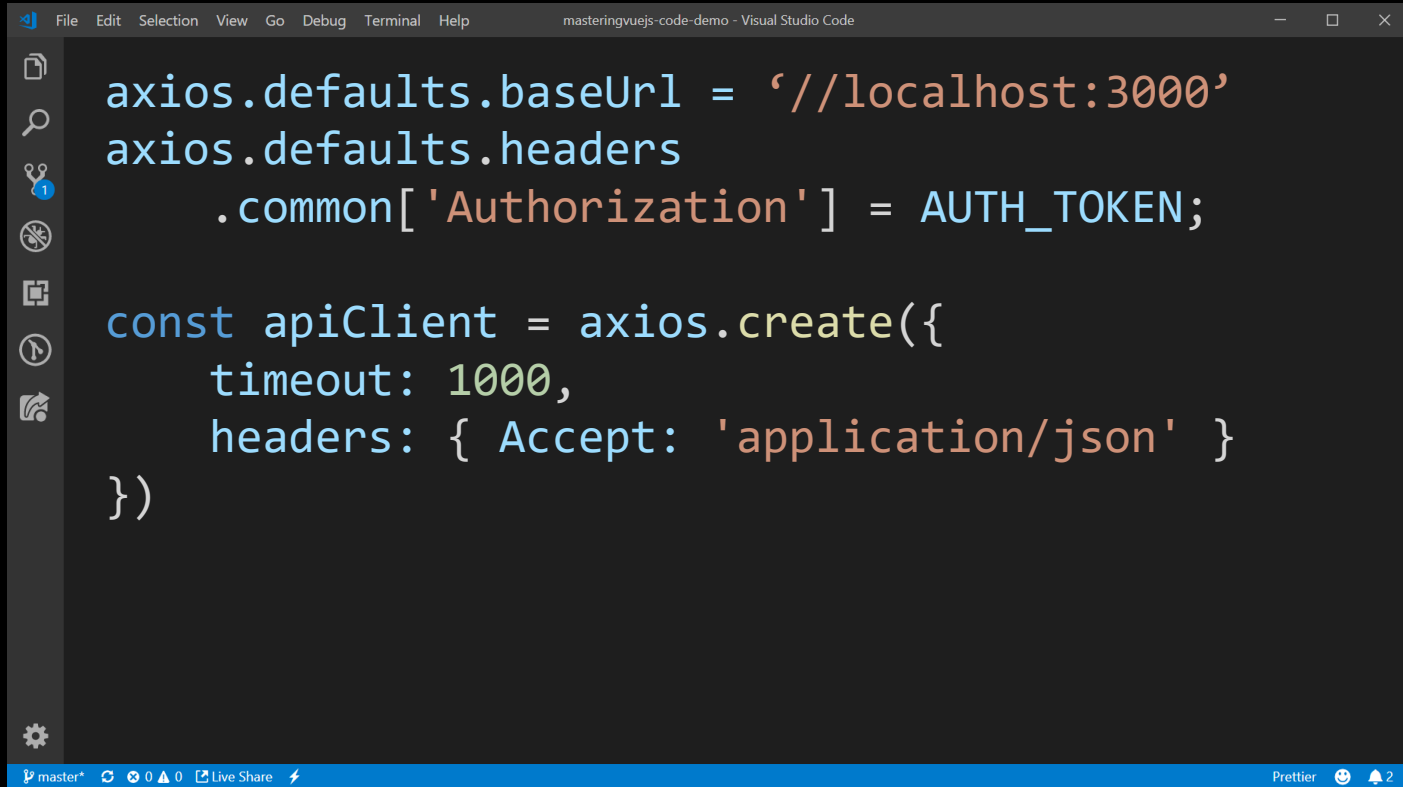




```
const apiClient = axios.create({
  baseUrl: '//localhost:3000',
  timeout: 1000,
  headers: { Accept: 'application/json' }
})
```





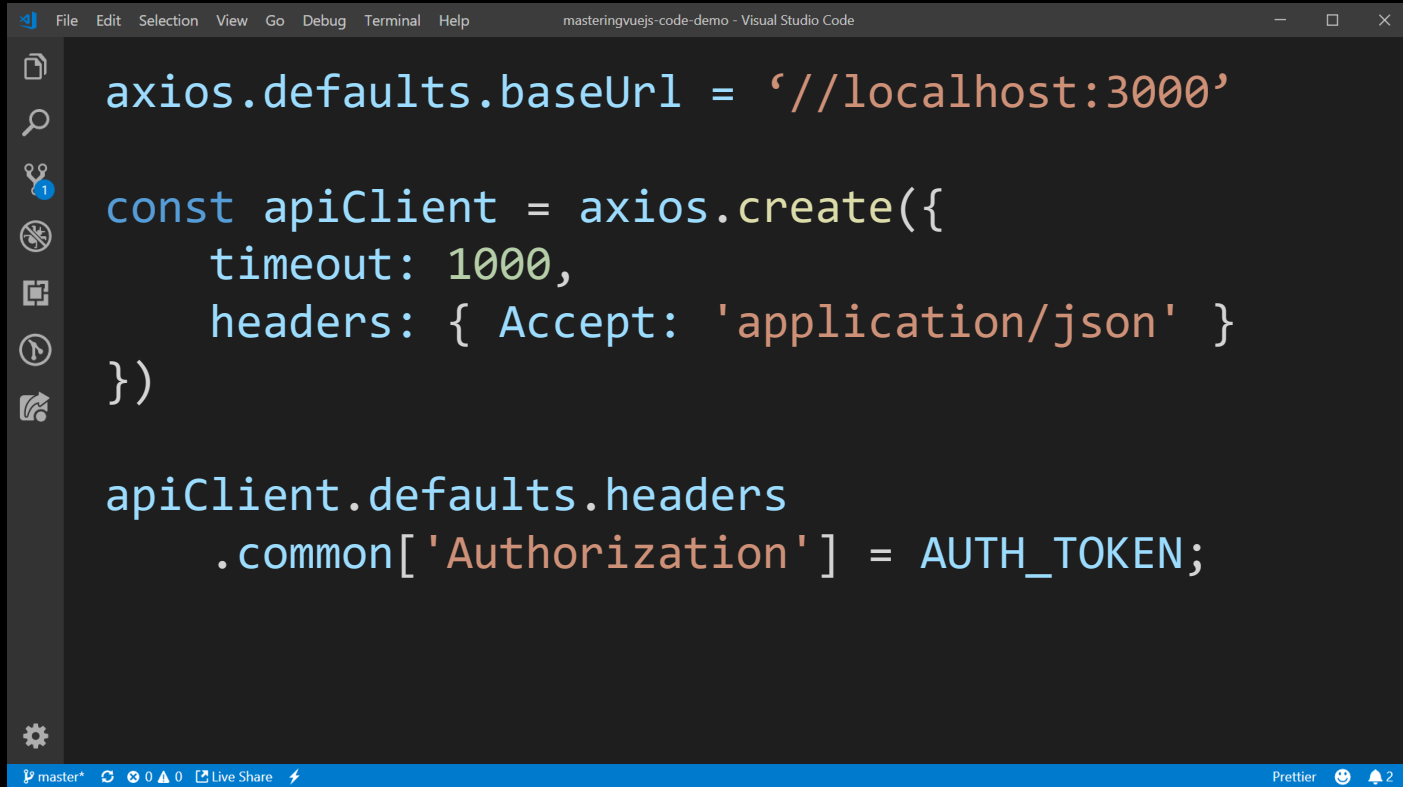


```
File Edit Selection View Go Debug Terminal Help masteringvuejs-code-demo - Visual Studio Code
axios.defaults.baseUrl = '//localhost:3000'
axios.defaults.headers
    .common['Authorization'] = AUTH_TOKEN;

const apiClient = axios.create({
  timeout: 1000,
  headers: { Accept: 'application/json' }
})
```

master\* 0 0 0 Live Share Prettier 2





```
File Edit Selection View Go Debug Terminal Help  masteringvuejs-code-demo - Visual Studio Code

axios.defaults.baseUrl = 'http://localhost:3000'

const apiClient = axios.create({
  timeout: 1000,
  headers: { Accept: 'application/json' }
})

apiClient.defaults.headers
  .common['Authorization'] = AUTH_TOKEN;

master* 0 0 Live Share Prettier 2
```



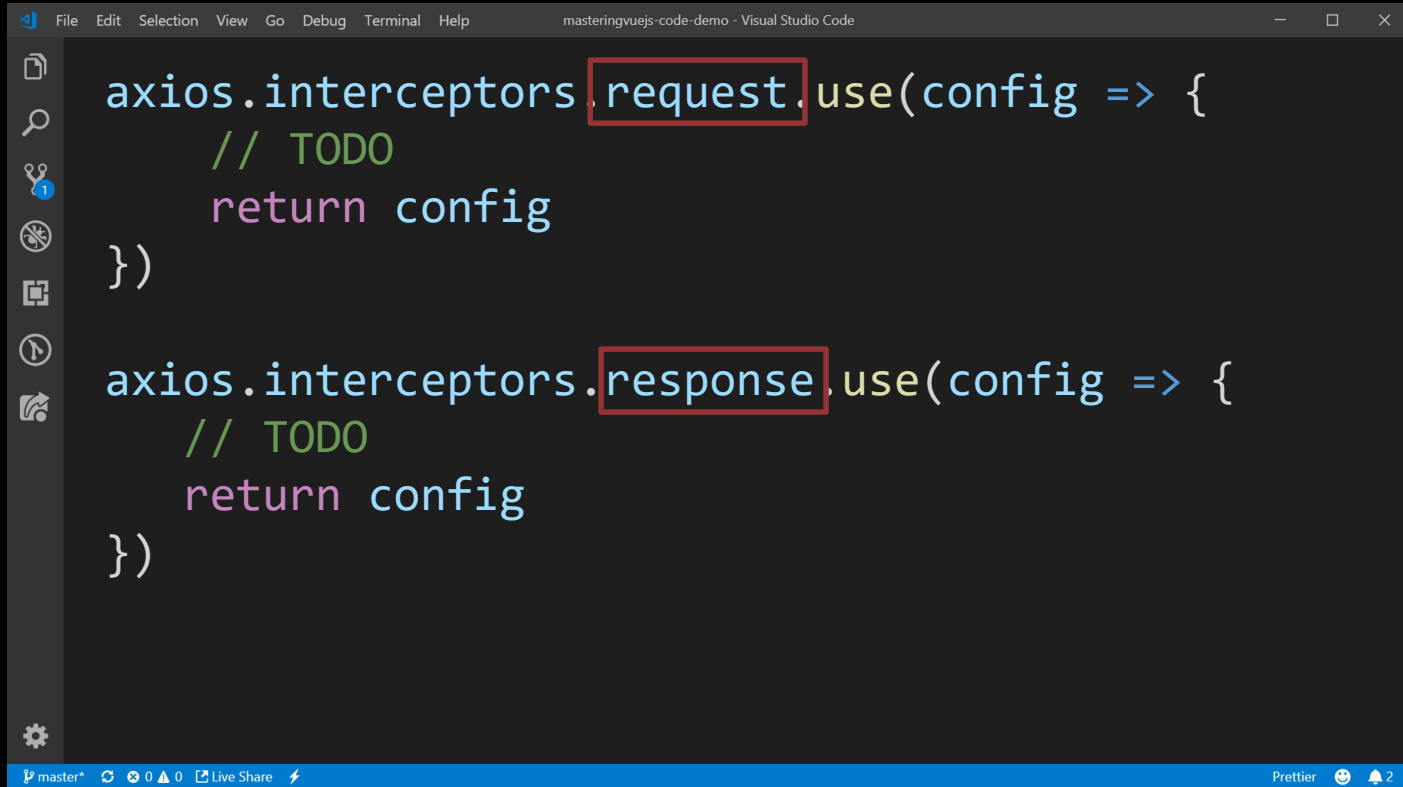
# Agenda

Getting Started

Configuration

Interceptors

Put all together



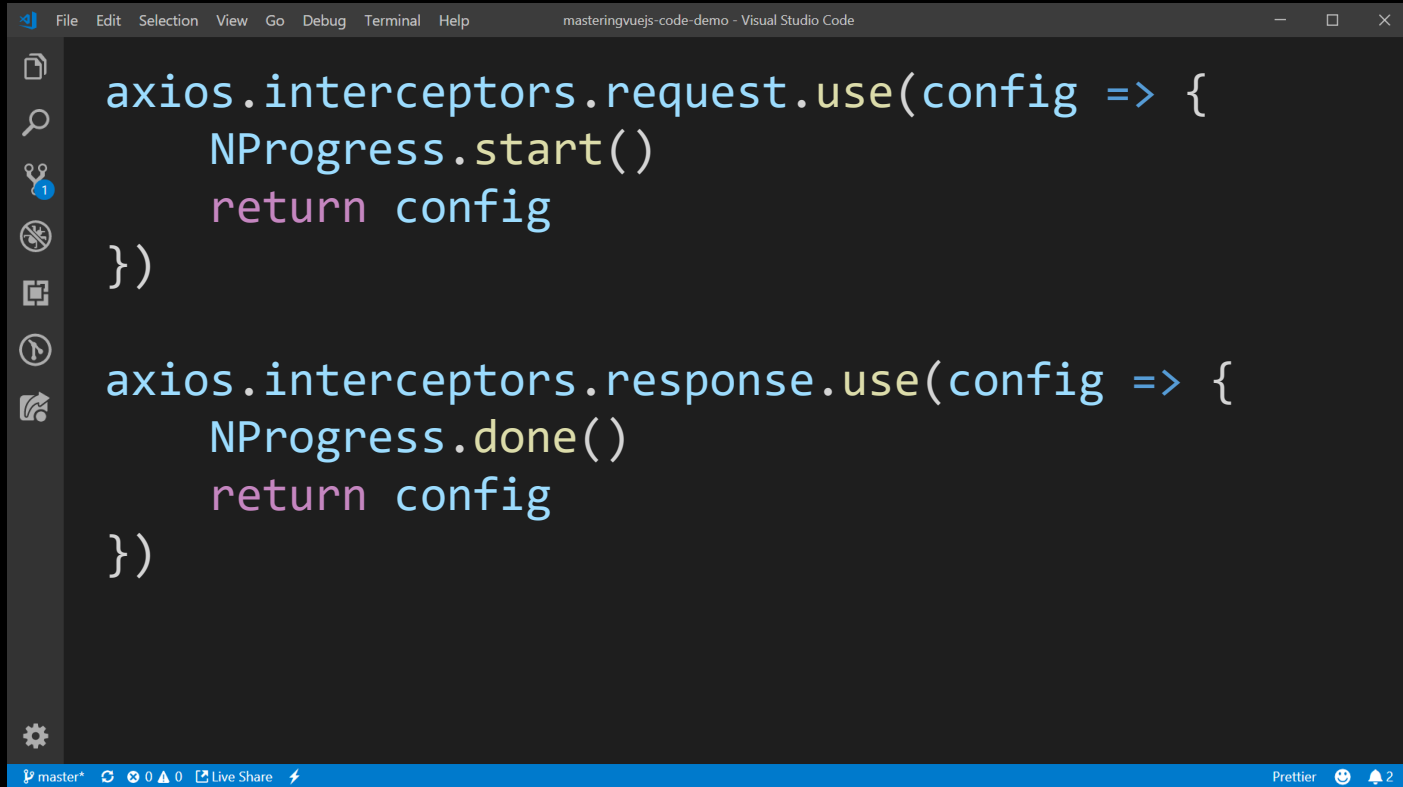
```
File Edit Selection View Go Debug Terminal Help masteringvuejs-code-demo - Visual Studio Code

axios.interceptors.request.use(config => {
  // TODO
  return config
})

axios.interceptors.response.use(config => {
  // TODO
  return config
})

master* 0 0 0 Live Share Prettier 2
```





```
File Edit Selection View Go Debug Terminal Help masteringvuejs-code-demo - Visual Studio Code
```

```
axios.interceptors.request.use(config => {  
  NProgress.start()  
  return config  
})  
  
axios.interceptors.response.use(config => {  
  NProgress.done()  
  return config  
})
```

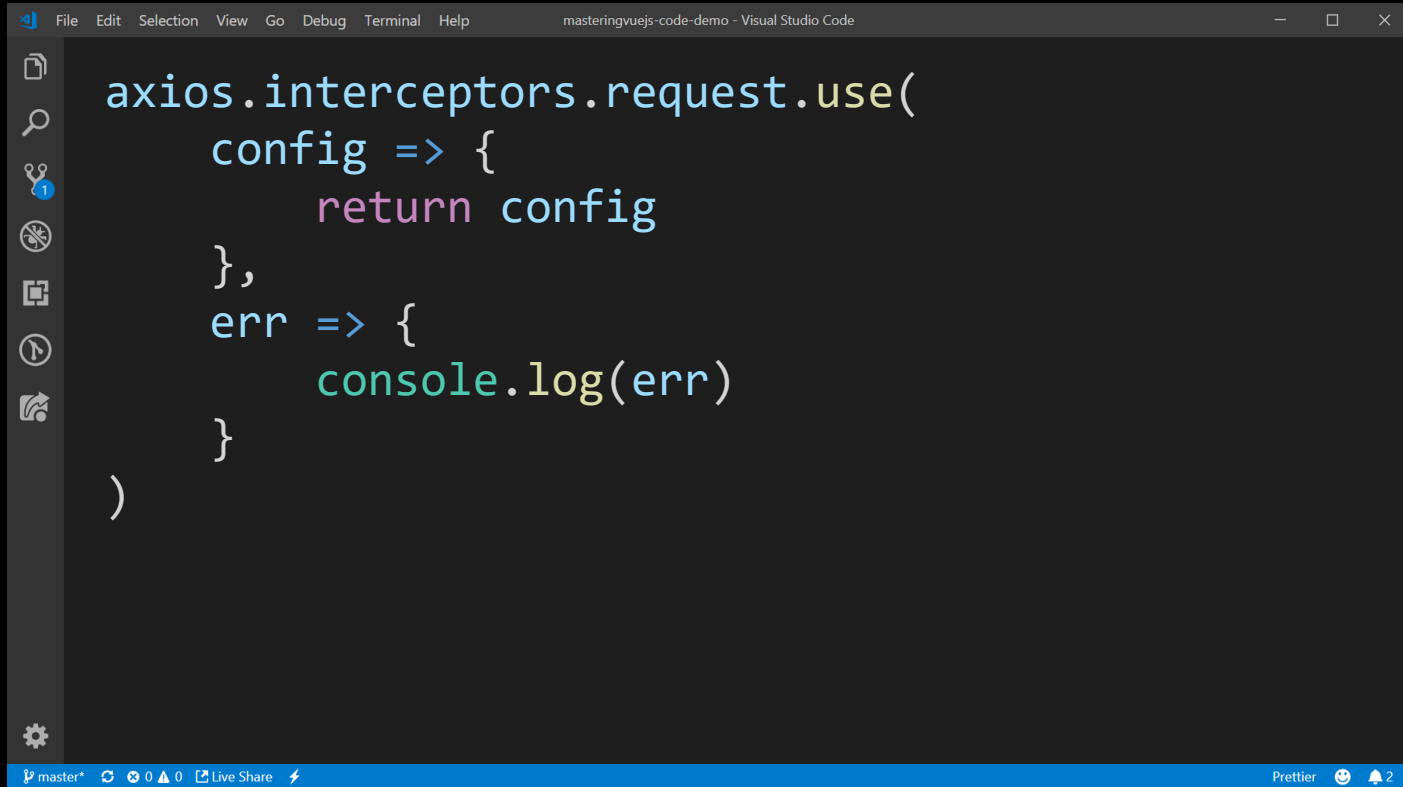
master\* 0 0 0 Live Share Prettier 2





```
axios.interceptors.request.use(config => {  
  if (AuthService.token()) {  
    config.headers.authorization =  
      'Bearer ' + AuthService.token()  
  }  
  return config  
})
```



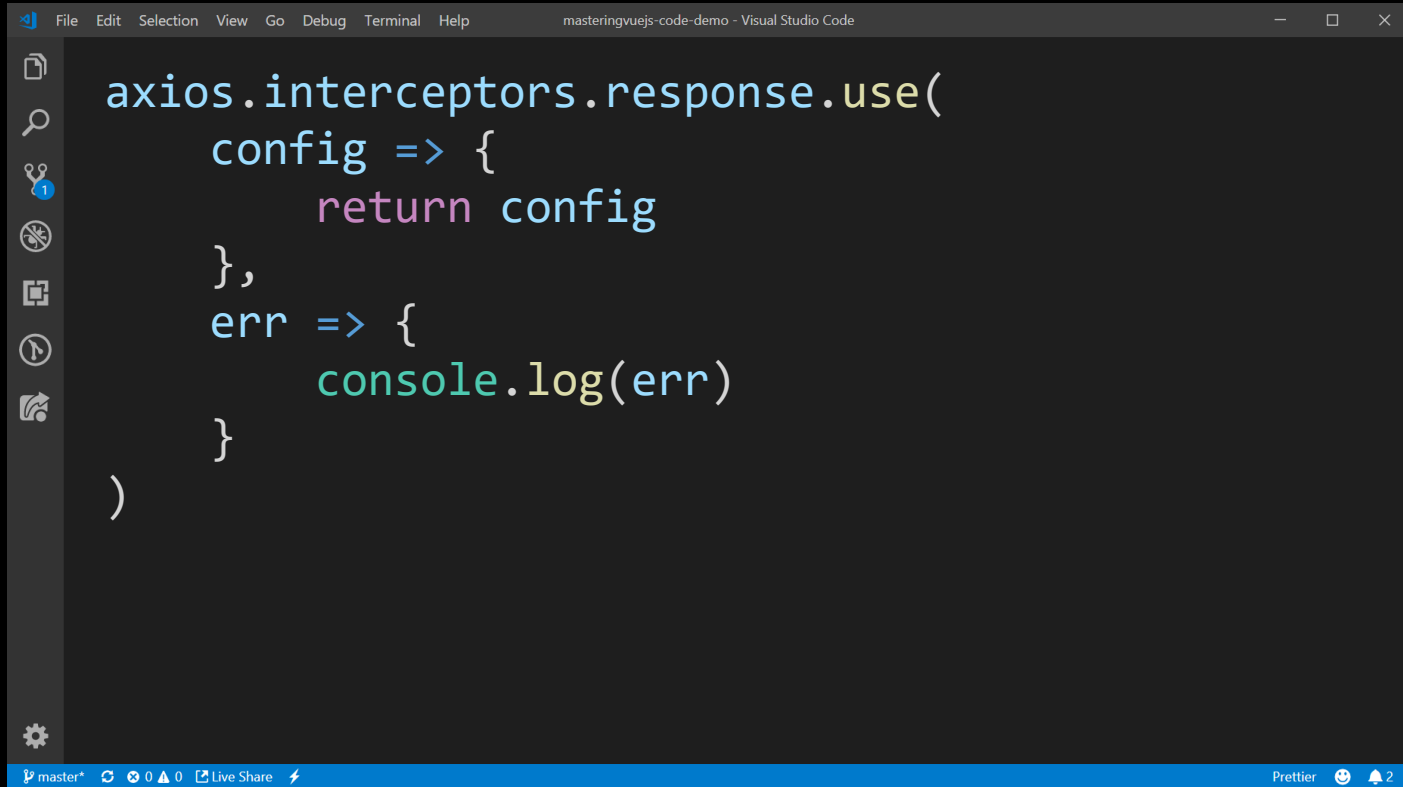


```
File Edit Selection View Go Debug Terminal Help masteringvuejs-code-demo - Visual Studio Code
```

```
axios.interceptors.request.use(  
  config => {  
    return config  
  },  
  err => {  
    console.log(err)  
  }  
)
```

master\* 0 0 0 Live Share Prettier 2





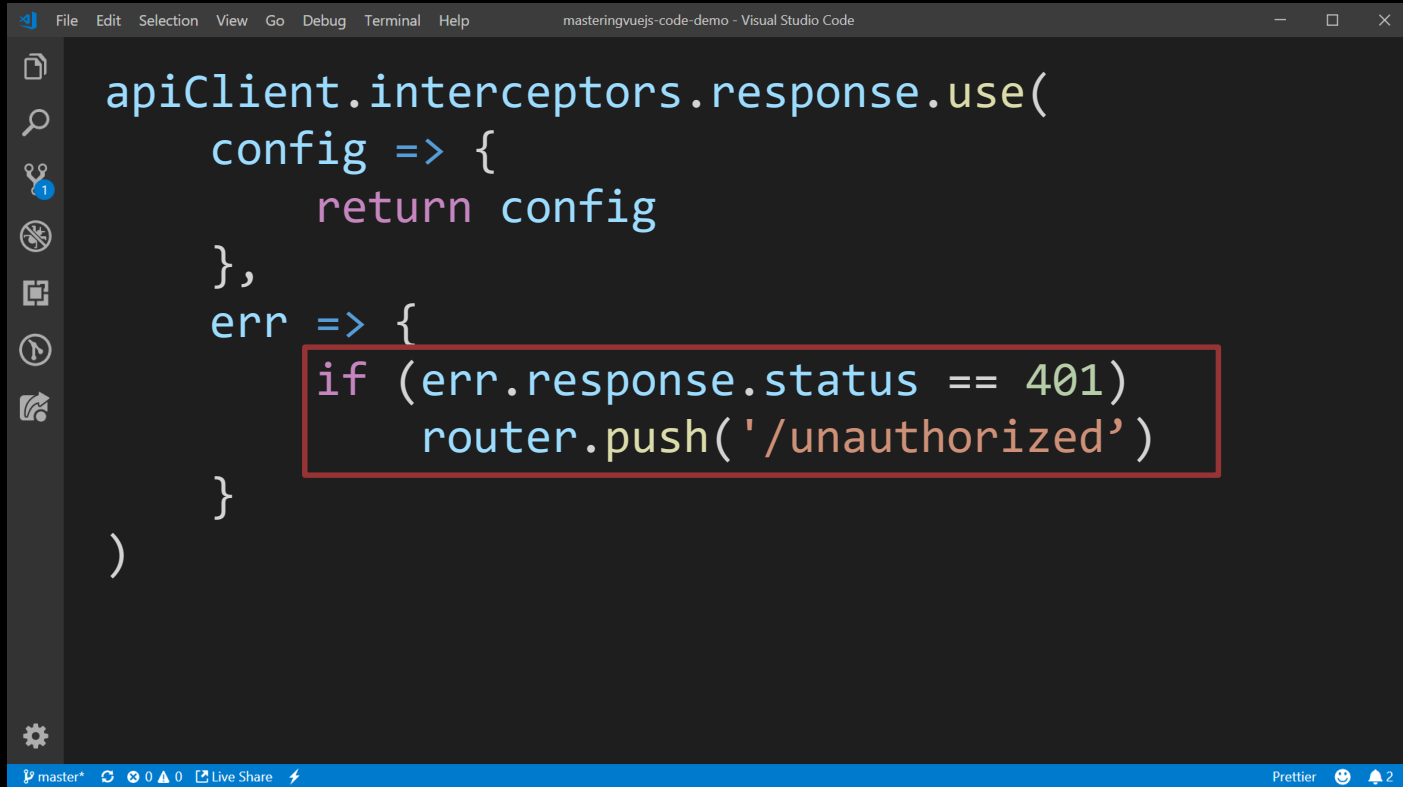
```
File Edit Selection View Go Debug Terminal Help masteringvuejs-code-demo - Visual Studio Code
```

```
axios.interceptors.response.use(  
  config => {  
    return config  
  },  
  err => {  
    console.log(err)  
  }  
)
```

master\* 0 0 0 Live Share Prettier 2







```
ApiClient.interceptors.response.use(  
  config => {  
    return config  
  },  
  err => {  
    if (err.response.status == 401)  
      router.push('/unauthorized')  
  }  
)
```

Visual Studio Code interface details: The top menu bar includes File, Edit, Selection, View, Go, Debug, Terminal, and Help. The title bar reads 'masteringvuejs-code-demo - Visual Studio Code'. The left sidebar contains icons for Explorer, Search, Source Control, Run and Debug, Extensions, and Testing. The bottom status bar shows 'master\*' with 0 errors and 0 warnings, a 'Live Share' button, and the 'Prettier' formatter. A notification bell icon shows 2 alerts.



# Agenda

Getting Started

Configuration

Interceptors

Put all together



# Your Turn!

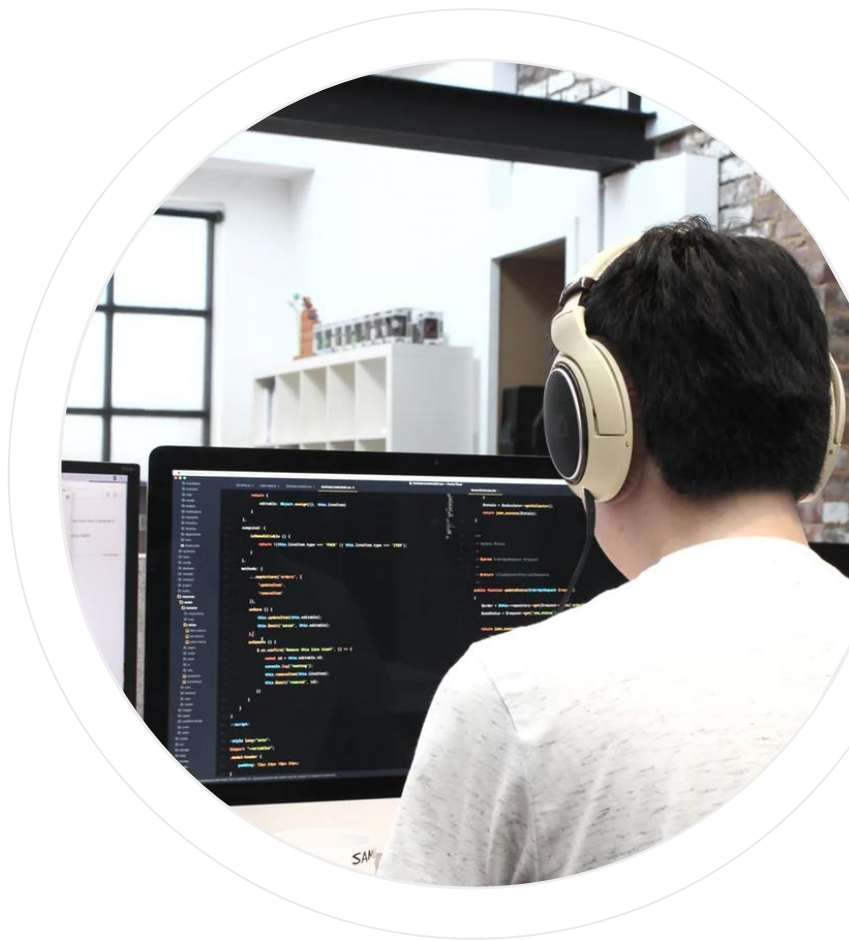
Follow the instructions to complete the code exercises and challenges



Workshop Guide  
[bit.ly/masteringvuejs](https://bit.ly/masteringvuejs)



Workshop Code  
[bit.ly/vuejs-oslo-2019](https://bit.ly/vuejs-oslo-2019)



@thiagospassos



# Mastering Vue.js State Management

# Agenda

What is Vuex?

When to use Vuex?

Accessing State

changing State

# Agenda

What is Vuex?

When to use Vuex?

Accessing State

Changing State

# What is Vuex?

- Reactive state management pattern and library
- Serves as a centralised store for all components
- State changes tracked and predictable
- Leverages reactivity system for simple and efficient updates
- Supports advanced features





# Agenda

What is Vuex?

When to use Vuex?

Accessing State

Changing State



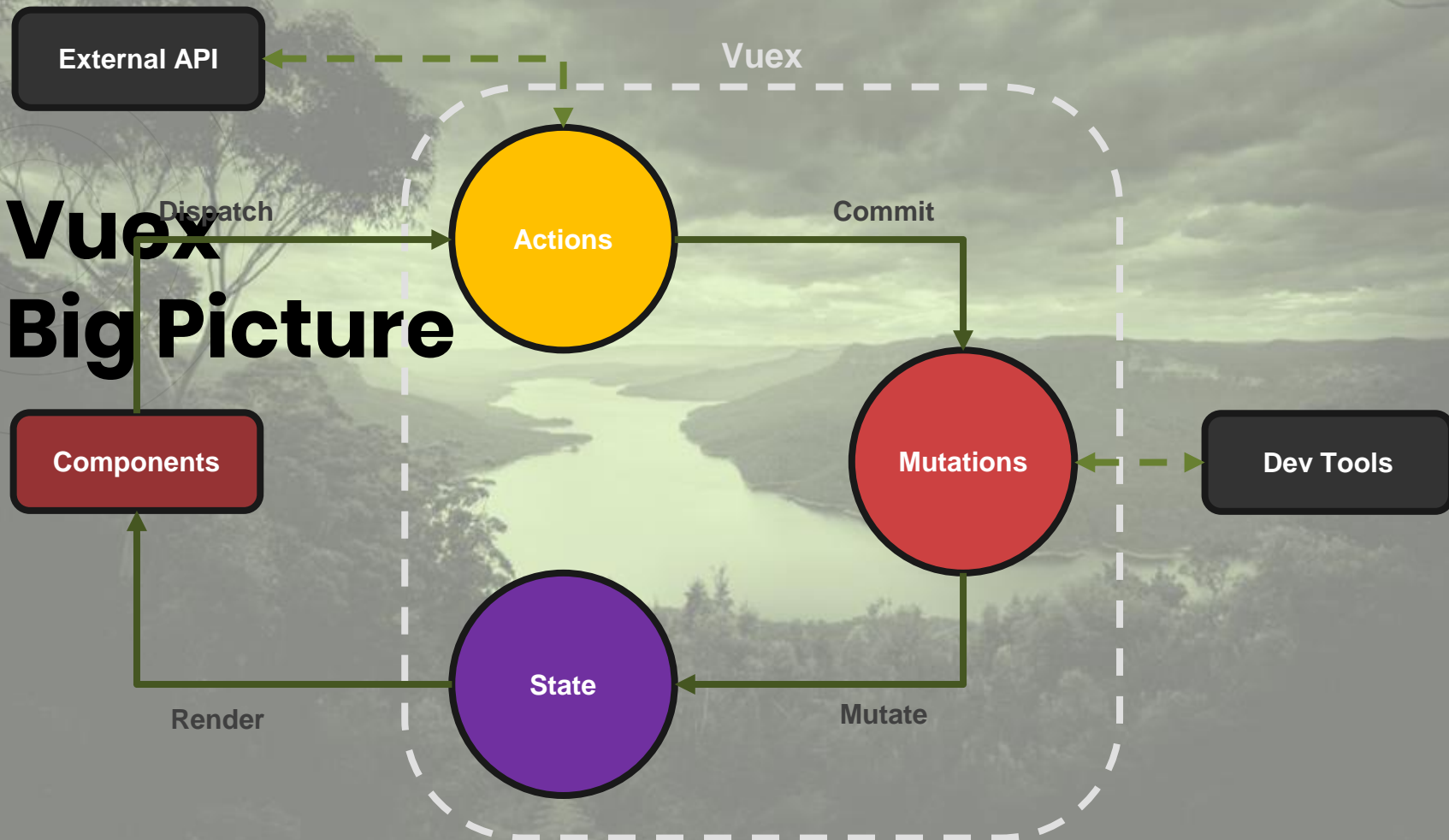


# When to use Vuex?

- Sharing data between more than two components
- Sharing data between components with no direct parent-child relationship
- Advanced debugging features such as time-travel
- Short-term vs. long-term productivity?



# Vuex Big Picture



# Agenda

What is Vuex?

When to use Vuex?

Accessing State

Changing State



# Demo: Accessing State

- Installing and configuring Vuex
- Working with Vuex state and getters
- Displaying shared state within components



# Core Concepts: State

- Single object containing all shared state
- Straight-forward to locate state

```
state: {  
  userName: 'Jason',  
  roles: [ 'Administrator', 'Contributor' ]  
}
```



# Core Concepts: Getters

- Computed properties for the store
- Result is cached based on dependencies

```
getters: {  
  isAdmin: (state) => {  
    return state.roles.contains(r => r === 'Administrator')  
  }  
}
```



# Agenda

What is Vuex?

When to use Vuex?

Accessing State

changing State



# Demo: Changing State

- Working with Vuex actions and mutations
- Tracking success and error notifications





# Core Concepts: Mutations

- Used to commit and track state changes
- Mutations are synchronous
- Mutations should only be called from actions

```
mutations: {  
  addUser: (state, payload) => {  
    return state.users.push(payload)  
  }  
}
```



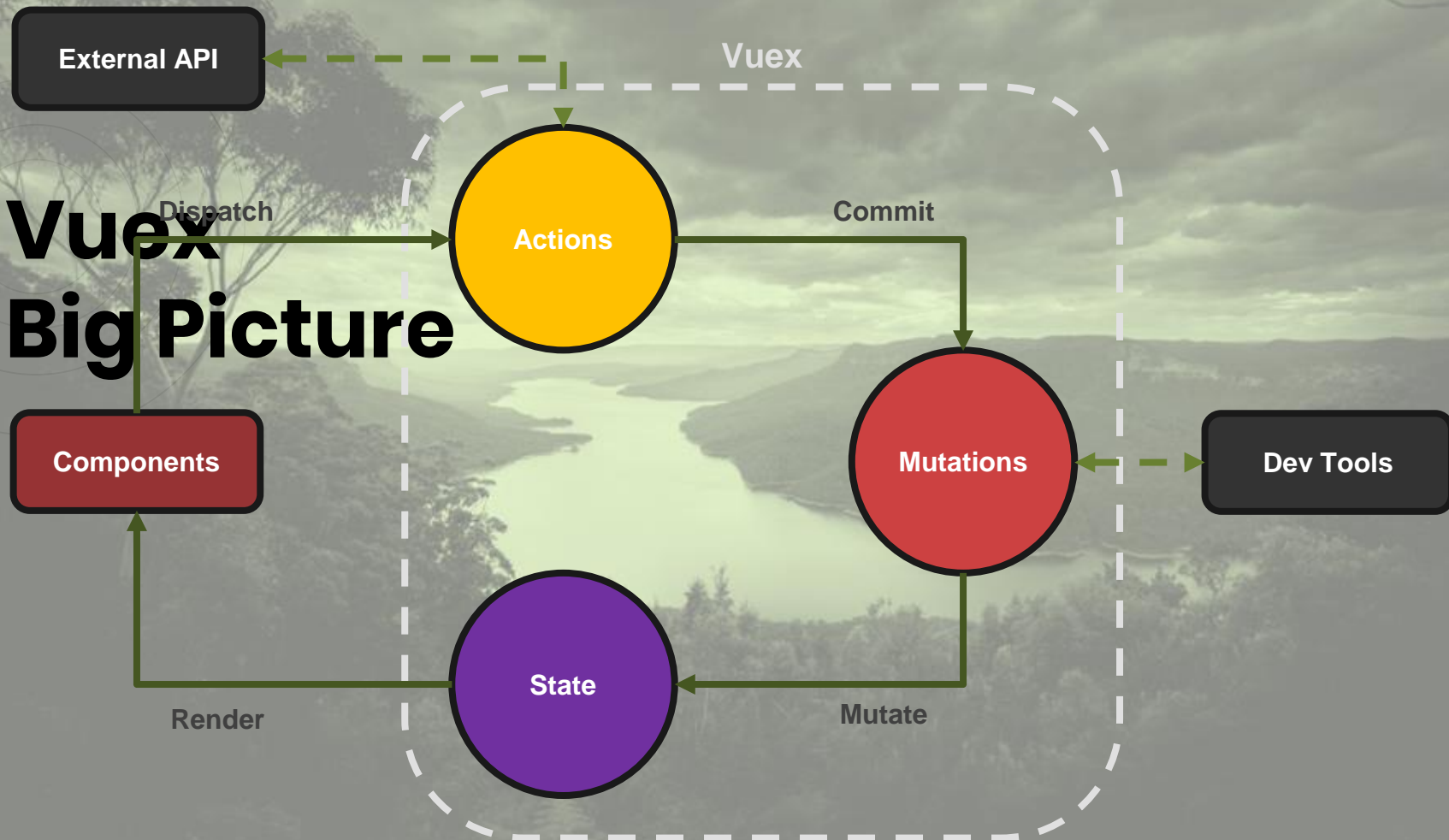
# Core Concepts: Actions

- Contain business logic and commit mutations
- Actions are asynchronous
- Always put mutations within actions

```
actions: {  
  addUser: ({ commit }, user) => {  
    return UserService.addUser(user).then(  
      () => commit('addUser', user)  
    )  
  }  
}
```



# Vuex Big Picture





# Key Points

- Serves as a centralised store for all components
- Share application level state across many components
- Efficiently access derived state from multiple components
- ...





# Key Points

- Effectively encapsulate logic to manage state changes
- Easily debug state changes with snapshots and time-travel



# Your Turn!

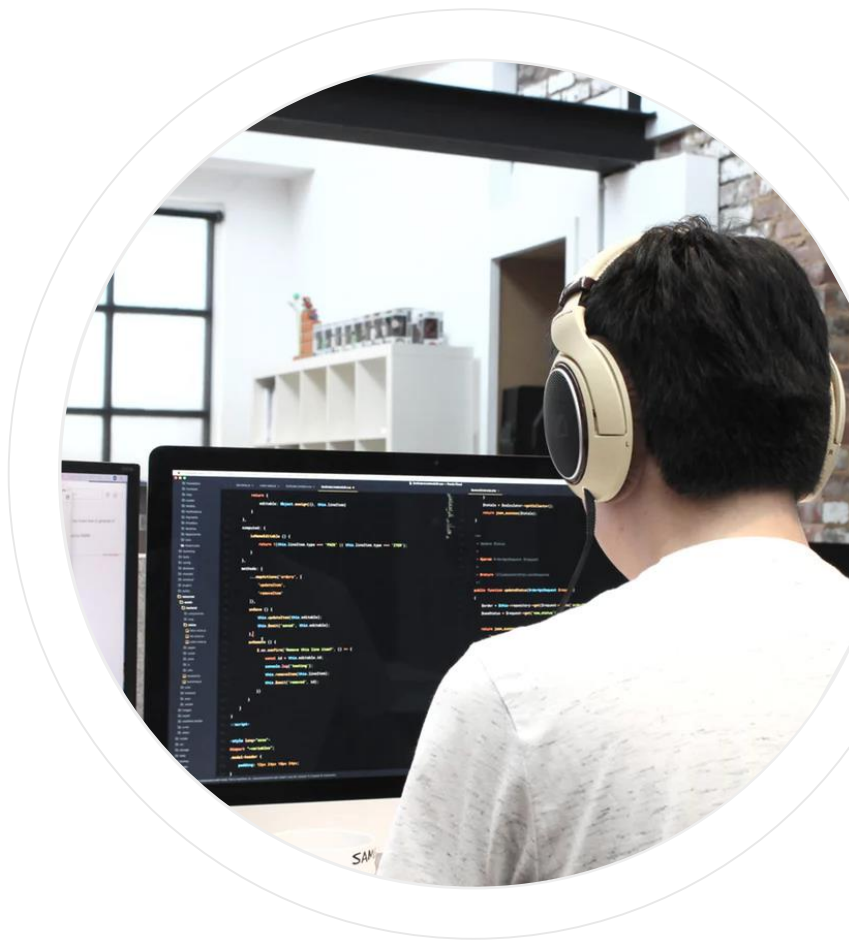
Follow the instructions to complete the code exercises and challenges



Workshop Guide  
[bit.ly/masteringvuejs](https://bit.ly/masteringvuejs)



Workshop Code  
[bit.ly/vuejs-oslo-2019](https://bit.ly/vuejs-oslo-2019)



@thiagospassos

# Thank you!

The resources for this workshop:



Workshop Guide

[bit.ly/masteringvuejs](https://bit.ly/masteringvuejs)



Workshop Code

[bit.ly/vuejs-oslo-2019](https://bit.ly/vuejs-oslo-2019)



@thiagospassos