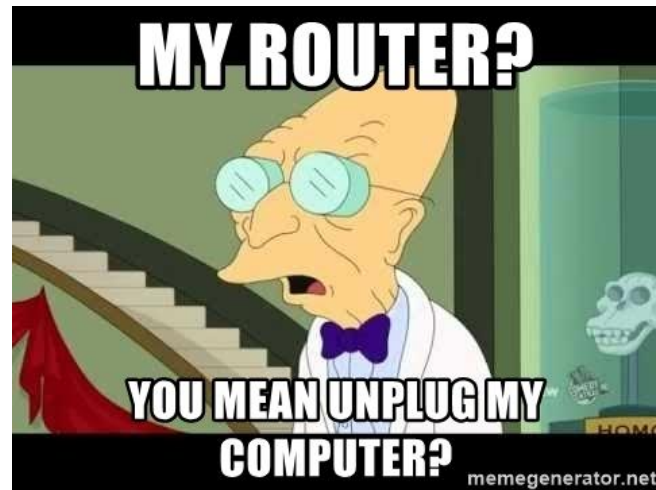




**JUST PROGRAMMING THE NEW  
ROUTER**



memegenerator.net





**Vue Router**

The official router for Vue.js.

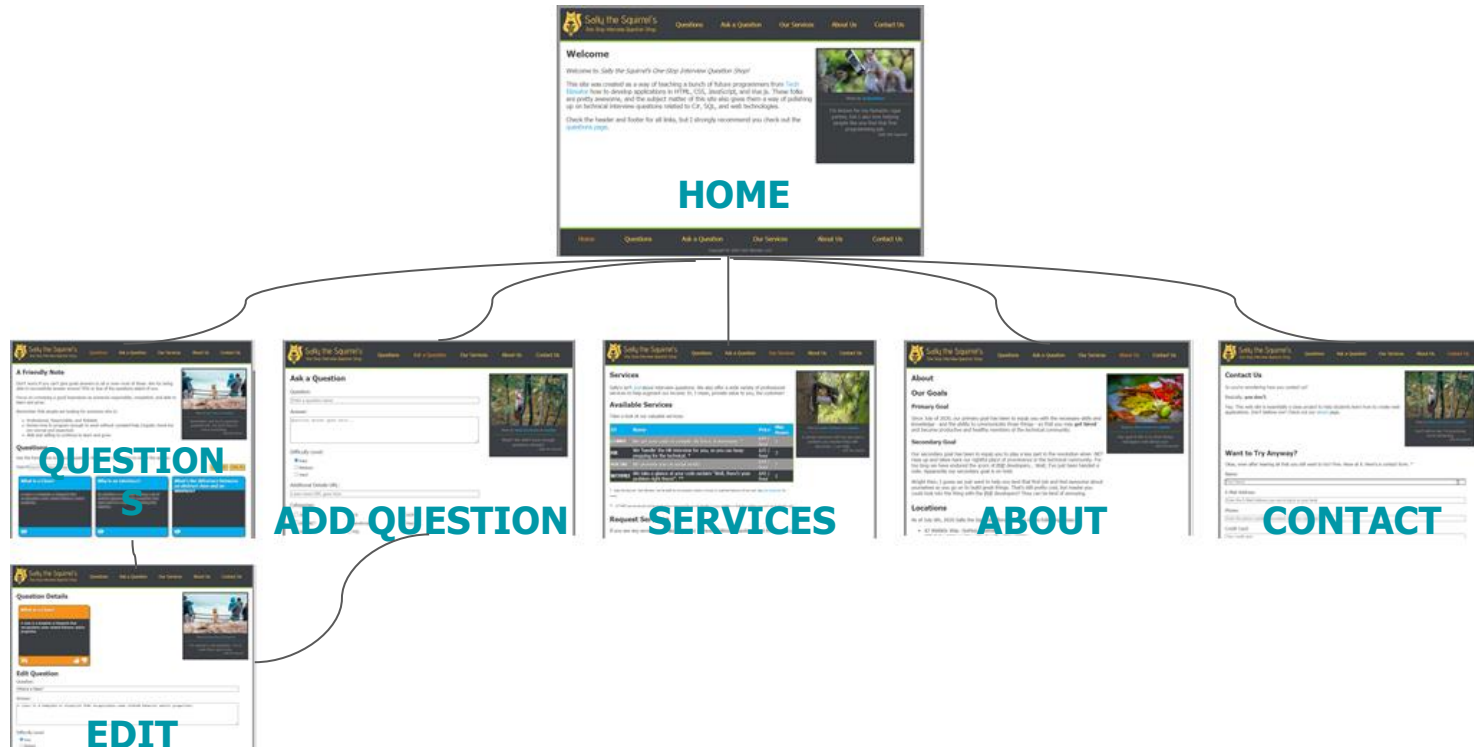
# Module 4-12

VUE Routing

# Objectives

- Why we need a “router” in a single page application (SPA)
- Define a new route
- Differences and uses of views and components directories
- Utilize router-link component
- Utilize router-view component
- Define dynamic routes
- `this.$route` object to access data from router
- `this.$router` object for programmatic navigation
- Lifecycle events

# Web Navigation



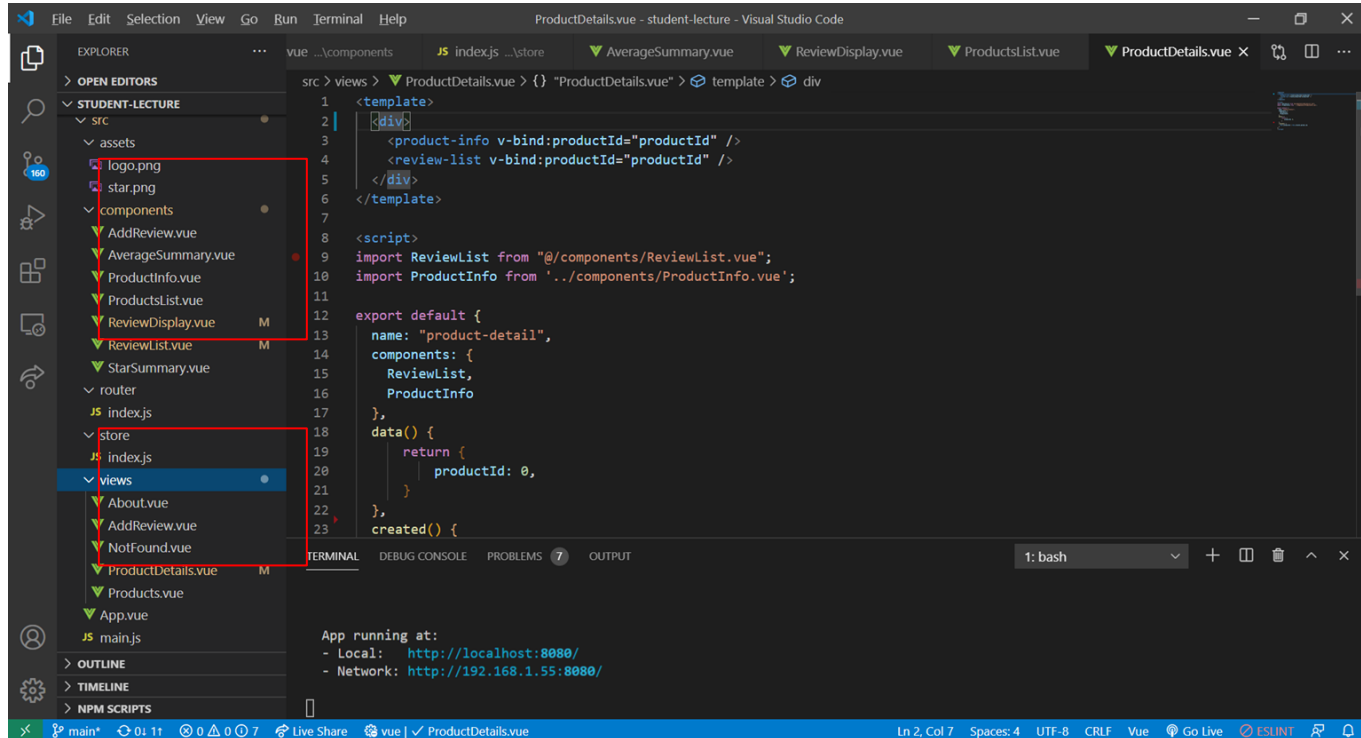
# Web Navigation

examplecom 1.0 > ca >				Search ca
Name	Type	Size		
assets	File folder			
personal-artwork	File folder			
portfolio-samples	File folder			
work-history	File folder			
biography.html	Chrome HTML Document	2 KB		
contact.html	Chrome HTML Document	2 KB		
index.html	Chrome HTML Document	2 KB		
personal-artwork.html	Chrome HTML Document	2 KB		
portfolio-samples.html	Chrome HTML Document	2 KB		
sitemap.html	Chrome HTML Document	2 KB		
social-media-links.html	Chrome HTML Document	2 KB		
work-history.html	Chrome HTML Document	2 KB		

# Vue.js Navigation



# Vue.js Navigation



# View vs. Components

## VIEWS

### Ask a Question

At Sally's we realize we can't have every question ever asked, but we sure can try! If you find something we missed, use this form to add a new entry so others can learn from your experience.

Questions submitted will be reviewed in a timely manner. Or not. We don't know. I'm mostly just interested in acorns. But still submit them!

### Add Question

Question

Answer

Difficulty

2

Add Question

Cancel

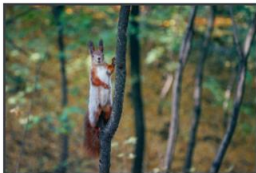


Photo by Alexey Savchenko on Unplash

What? We didn't have enough questions already?

- Sally the Squirrel

## COMPONENTS

Question

Answer

Difficulty

2

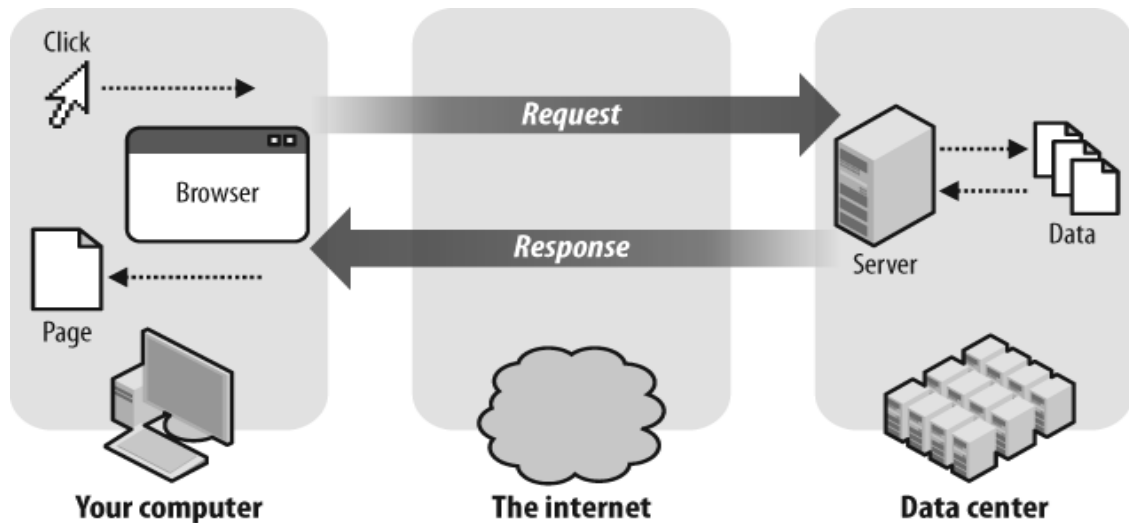
Add Question

Cancel



# What is Routing?

- Routing allows users to be redirected to a certain component via a URL.
- Remember MVC Spring RequestMappings? Similar idea.



# Installing Vuex Router

- During project creation, add Router.
- Add router to already created project

```
Vue CLI v3.7.0
Update available: 4.5.10

? Please pick a preset: Manually select features
? Check the features needed for your project: (Press <space> to select, <a> to toggle all, <i> to invert selection)
> (*) Babel
  ( ) TypeScript
  ( ) Progressive Web App (PWA) Support
  ( ) Router
  ( ) Vuex
  ( ) CSS Pre-processors
  (*) Linter / Formatter
  ( ) Unit Testing
  ( ) E2E Testing
```

```
C:\Users\Margaret Green\Workspace\first-project>vue add router
? Use history mode for router? (Requires proper server setup for index fallback in production) No

[ ] [ ] Invoking generator for core:router...
[ ] [ ] Installing additional dependencies...

added 1 package from 1 contributor and audited 1295 packages in 7.226s
found 0 vulnerabilities

[ ] Successfully invoked generator for plugin: core:router
The following files have been updated / added:

  src/router/index.js
  src/store/index.js
  src/views/About.vue
  src/views/Home.vue
  package-lock.json
  package.json
  src/App.vue
  src/main.js

You should review these changes with git diff and commit them.

C:\Users\Margaret Green\Workspace\first-project>
```

# The src/router/index.js file overview

To define a route, an index.js file is needed.

- There is some boiler plate code beyond the scope of this class, you can carry those over for now, they are highlighted in blue.
- Our focus will be on the sections in red, which we will define.

```
import Vue from 'vue'
import Router from 'vue-router'
import Home from './views/Home.vue'
import About from './views/About.vue'
import NotFound from './views/NotFound.vue'

Vue.use(Router)
const routes = [
  {
    path: '/',           // Required
    name: 'home',        // Recommended, but not required
    component: Home      // Required
  },
  {
    path: '/about',
    name: 'about',
    component: About
  },
  {
    path: '*',
    name: 'notFound',
    component: NotFound
  }
]
const router = new VueRouter({
  routes
})
export default router
```

# The router.js file: importing views

We need to first define the components a user can potentially be routed to, this can be achieved through imports:

```
import Home from '../views/Home.vue'  
import About from '../views/About.vue'  
import Books from '@views/Books.vue'
```

Note that in this example the components are in a folder called views, this should make no difference, Home, About, and Books are VUE components.

# The router/index.js file: importing components.

Next, we define the routes, these physically map the view components we imported.

```
import Home from './views/Home.vue'  
import About from './views/About.vue'  
import Books from '@views/Books.vue'
```

```
routes: [  
  {  
    path: '/',  
    name: 'home',  
    component: Home  
  },  
  {  
    path: '/about',  
    name: 'about',  
    component: About  
  },  
  {  
    path: '/books',  
    name: 'books',  
    component: Books  
  }  
]
```

Each route is a JSON object, comprised of three key value pairs:

- **path:** what the user types into the URL
- **name:** This is how we will refer to the route within App.vue
- **component:** The component that was imported in, that the user will be redirected to

# Adding router links to App.vue

We are almost done! The last step is want to define the routes within App.vue.

```
<template>
  <div id="app">
    <header>
      <ul class="nav">
        <router-link v-bind:to="{ name: 'home' }" tag="li">Home</router-link>
        <router-link to="/about" tag="li">About the Author</router-link>
        <router-link :to="{ name: 'books' }" tag="li">Related Books</router-link>
      </ul>
    </header>
    <router-view class="content"/>
  </div>
</template>
```

First we will talk about the red code – router-link, and then we will discuss later the blue code.

# Adding the router links to App.vue

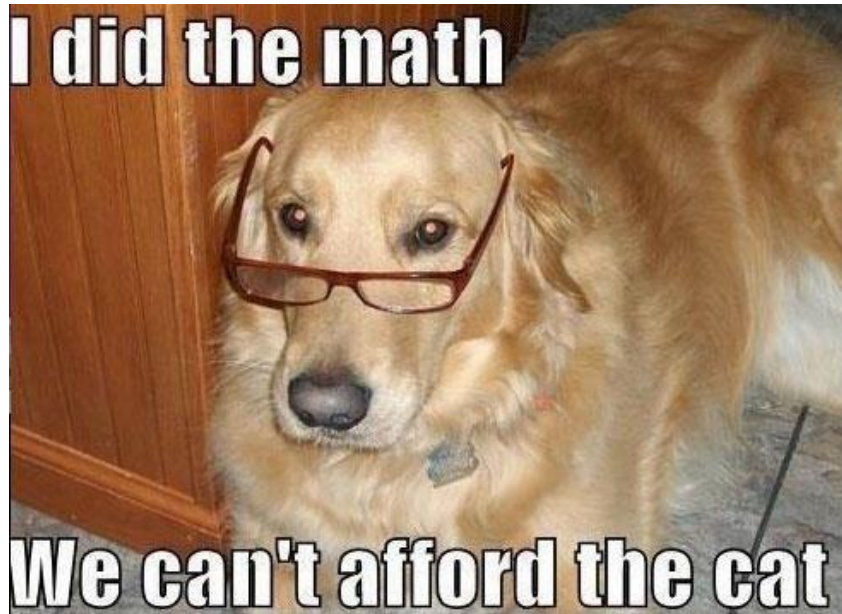
This is the basic structure of a router-link

```
<router-link v-bind:to="{ name: 'home' }" tag="li">Home</router-link>
```

This should match up to the name of the route specified in router/index.js

You can define the type of html element you want rendered as the link here.

Let's Implement These Routes!





# Dynamic Routing

- Sometimes data is encapsulated within a URL path:
  - Consider the following URL: (account/**135**)
  - The value 135 could very well be an ID that is associated with a “row” of data.
- The purpose of dynamic routing is to implement these URL parameters.
- Remember MVC Spring PathParams? Similar idea.

# Dynamic Routing : router/index.js

We must first implement a route , where the path parameter contains a placeholder for the path variable.

```
{  
  path: '/users/:id',  
  name: 'user',  
  component: User  
}
```

# Dynamic Routing : Defining the Router Links

We can now define router links

```
<tr v-for="user in users" :key="user.id">
  <td><router-link :to="{name: 'users', params: {id: user.id}}">{{user.id}}</router-link></td>
  <td>{{user.name}}</td>
</tr>
```

Template section

```
data() {
  return {
    users: [
      {
        id: 1,
        name: "Leanne Graham"
      },
      {
        id: 2,
        name: "Ervin Howell"
      }
    ]
  }
}
```

Script section

Here we have a v-for that will iterate through every object in the users array, each time it does so it generates a new router-link with its respective id value.

Two links are generated:

- /users/1
- /users/2

# Router-View

```
1 <template>
2   <div id="app">
3     <AppHeader />
4     <router-view /> <!-- The currently active view will be presented here -->
5     <app-footer />
6   </div>
7 </template>
```

# Let's Implement Dynamic Routes!



# Router-link with parameters

```
1 <router-link v-bind:to="{name:'product-details', params: {id: product.id}}">  
2   View Product Details  
3 </router-link>
```

# This \$route

```
1 created() {  
2   const id = this.$route.params.id; // Grabs the route parameter named id, if it was present  
3   this.question = this.$store.state.questions.find(q => q.id === id);  
4  
5   if (!this.question) {  
6     this.$router.push({name: 'NotFound'});  
7   }  
8 }
```

# Routing

SomeComponent.vue

```
1 <router-link v-bind:to="{name: 'product-details', params: {id: product.id}}">
2   View Product Details
3 </router-link>
```

router/index.js

```
1 const routes = [
2   {
3     path: '/',
4     name: 'products',
5     component: Products,
6   },
7   {
8     path: '/products/:id',
9     name: 'product-details',
10    component: ProductDetails,
11  },
12  {
13    path: '*',
14    name: 'not-found',
15    component: NotFound,
16  }
17 ]
```

ProductDetails.vue

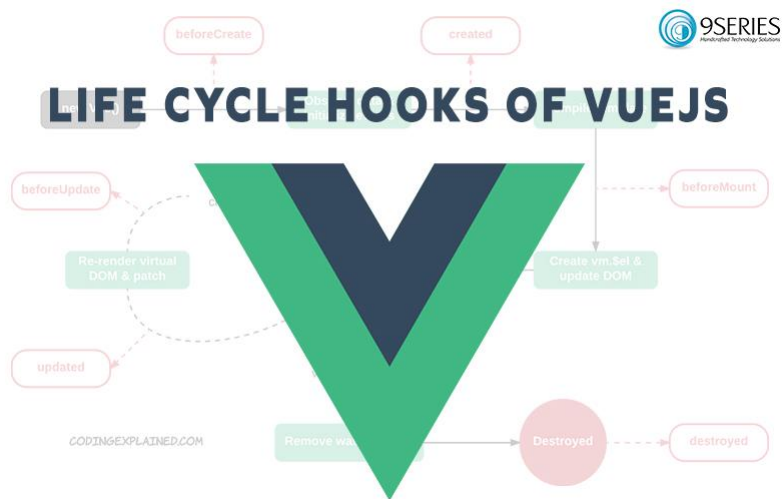
```
1 <script>
2 export default {
3   name: "product-detail",
4   data() {
5     return {
6       productId: 0,
7     }
8   },
9   created() {
10    this.productId = this.$route.params.id;
11  }
12 };
13 </script>
```



## Adding reviews with this.\$router

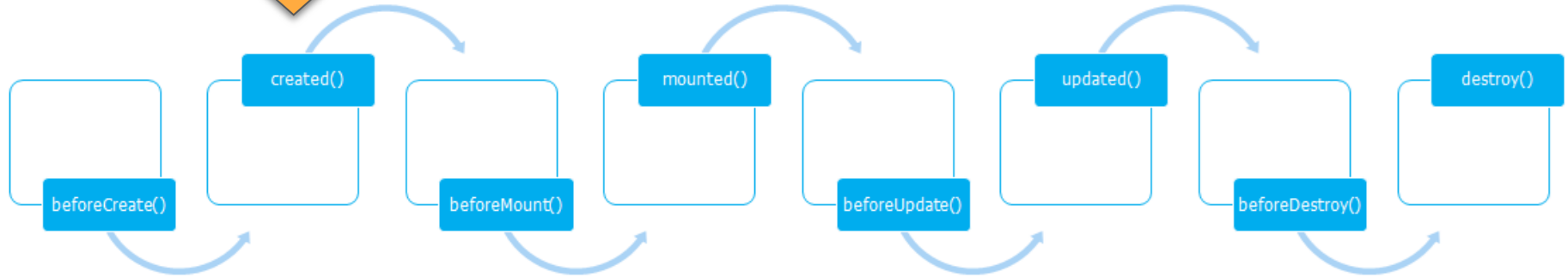
```
6
7 // named route
8 router.push({ name: 'user', params: { userId: '123' } })
9
10 // with query, resulting in /register?plan=private
11 router.push({ path: 'register', query: { plan: 'private' } })
```

# Life cycle hooks



# Life cycle events

You mostly just care  
about this one



# created()

```
1 // This fires after the component is created but before it renders.
2 // You can access any data, props, and $route info you need to here.
3 // This is also a good place to kick off requests for data your component will eventually need
4 created() {
5     const id = this.$route.params.id; // Grabs the route parameter named id, if it was present
6     this.question = this.$store.state.questions.find(q => q.id === id);
7
8     if (!this.question) {
9         this.$router.push({name: 'NotFound'});
10    }
11 }
```

# Let's Code! But first – RECAP!



# Objectives

- Differences and uses of views and components directories

## View

**Ask a Question**

At Sally's we realize we can't have every question ever asked, but we sure can try! If you find something we missed, use this form to add a new entry so others can learn from your experience. Questions submitted will be reviewed in a timely manner. Or not. We don't know. I'm mostly just interested in acorns. But still submit them!

**Add Question**

Question


Answer

Difficulty

2

Add Question

Cancel



## Components



Question

Answer

Difficulty

2

Add Question

Cancel

# Objectives

- Differences and uses of views and components directories
- Utilize router-link component

```
<router-link v-bind:to="{ name: 'home' }" tag="li">Home</router-link>
```

# Objectives

- Differences and uses of views and components directories
- Utilize router-link component
- Utilize router-view component

```
<template>
  <div id="app">
    <header>
      <ul class="nav">
        <router-link :to="{ name: 'home' }" tag="li">Home</router-link>
        <router-link :to="{ name: 'about' }" tag="li">About the Author</router-link>
        <router-link :to="{ name: 'books' }" tag="li">Related Books</router-link>
      </ul>
    </header>
    <router-view class="content"/>
  </div>
</template>
```



• Home	• About the Author	• Related Books
<b>Products</b>		
ID Name	# Reviews	
1 Cigar Parties for Dummies	4	
2 Personal Finance for Dummies	0	
3 Project Management for Dummies	0	
4 Critical Thinking Skills for Dummies	0	
5 Home Maintenance for Dummies	0	

Where component  
will be displayed



# Objectives

- Differences and uses of views and components directories
- Utilize router-link component
- Utilize router-view component
- Define dynamic routes

```
{  
  path: '/users/:id',  
  name: 'user',  
  component: User  
}
```

```
<tr v-for="user in users" :key="user.id">  
  <td><router-link :to="{name: 'users', params: {id: user.id}}">{{user.id}}</router-link></td>  
  <td>{{user.name}}</td>  
</tr>
```

# Objectives

- Differences and uses of views and components directories
- Utilize router-link component
- Utilize router-view component
- Define dynamic routes
- this.\$route object to access data from router

```
1 <script>
2 export default {
3   name: "product-detail",
4   data() {
5     return {
6       productId: 0,
7     }
8   },
9   created() {
10    this.productId = this.$route.params.id;
11  }
12 };
13 </script>
```

# Objectives

- Differences and uses of views and components directories
- Utilize router-link component
- Utilize router-view component
- Define dynamic routes
- this.\$route object to access data from router
- this.\$router object for programmatic navigation

```
8     if (!this.question) {  
9         this.$router.push({name: 'NotFound'});  
10    }  
11 }
```

# Objectives

- Differences and uses of views and components directories
- Utilize router-link component
- Utilize router-view component
- Define dynamic routes
- `this.$route` object to access data from router
- `this.$router` object for programmatic nav
- Lifecycle events

