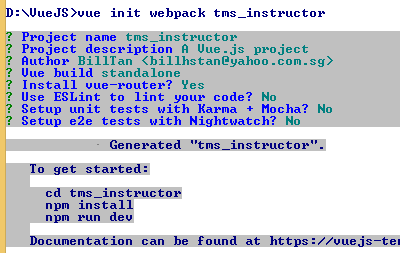
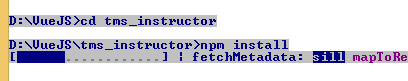
The following command, **vue init webpack tms\_instructor** was used to create a Vue SPA project, **tms\_instructor**. This project was not integrated into the Asp.NET core project. It was created for practice purposes.

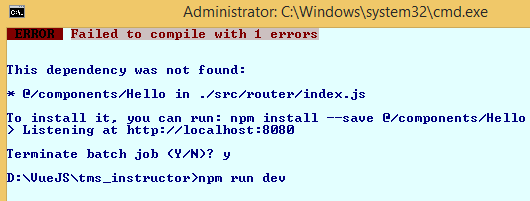


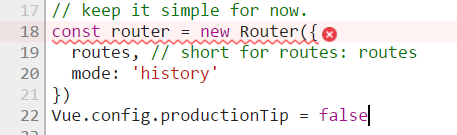


**<project name>\src\main.js is** where you configure the router

You use the **<router-vue>** tag inside the **App.vue** file's **<template>** section to define a *placeholder* which helps you to render the correct component.

When I was working on the **Main.js**, I did not notice that I have to make changes on the **Index.js**. The online tutorial did not remind me about the **index.js**.





The concept of **<router-link>** tag

The aim of a single page application (SPA) is such that the page does not *reload* again. To accomplish this, you need to use the **<router-link></router-link>** tag.

The tutorial at [**https://medium.com/@softwarecf/getting-started-with-vue-router-37cd7d783245**](https://medium.com/@softwarecf/getting-started-with-vue-router-37cd7d783245) discusses on **router-view** and **router-link** by using team and team member detail example (parent-child example).

OSchool colors:

#a39160

163 145 96

#222222

#343434

npm install moment

npm install vuetify --save-dev

I need to install the style-loader so that I can easily load CSS.

npm install style-loader --save

npm run build

npm install jquery --save

npm install webpack -g

npm install moment --save

I usually use npm run build to make the build files inside the dist directory. Then I use the gulp to copy the files into the wwwroot directory.

Using moment inside the Vue JS Single Page Application project.

Reference: <https://stackoverflow.com/questions/34308004/moment-js-with-vuejs>

npm install v-flex --save

**v-for** is a directive. **v-model** is also a directive.

Creating a View-Model

In **Vue.js**, view-models are created using the **Vue** class. If you are wondering what a view-model is, you can think view-model as:

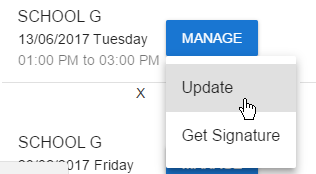
*An object that makes it very easy for you to display your model's data inside a view.*

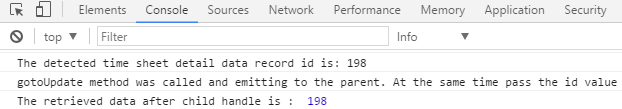
You can treat any *object literal* as a model, and any *HTML UI element* as a view.

The reference at [**https://www.sitepoint.com/getting-started-with-vue-js/**](https://www.sitepoint.com/getting-started-with-vue-js/) uses the object literal, view-model and other useful terms in the tutorial content.

You use a function definition to define a method. (Reference: [**https://www.w3schools.com/js/js\_objects.asp**](https://www.w3schools.com/js/js_objects.asp) )

https://medium.com/@sky790312/about-vue-2-parent-to-child-props-af3b5bb59829





The reference: [**https://codepen.io/shershen08/pen/xROOxw**](https://codepen.io/shershen08/pen/xROOxw) has working code on how the Vue JS code work together to provide Create-Read-Update-Delete functionality.

<template id="current-month-timesheet-template">

<div>

    <h2>View current timesheet</h2>

    <div><h3>Manage Timesheet for {{timeSheetMonth}} / {{timeSheetYear}}</h3></div>

  <v-layout row>

   <v-list three-line>

     <v-list-item is="timesheetdetail-item"

      v-for="(oneTimeSheetDetail, index) in timeSheetDetailData"

      v-bind:key="oneTimeSheetDetail.timeSheetDetailId"

      v-bind:timeSheetDetailId = "oneTimeSheetDetail.timeSheetDetailId"

      v-bind:customerAccountName="oneTimeSheetDetail.customerAccountName"

      v-bind:dateOfLesson="oneTimeSheetDetail.dateOfLesson"

      v-bind:officialTimeInHHMM="oneTimeSheetDetail.officialTimeInHHMM"

      v-bind:officialTimeOutHHMM="oneTimeSheetDetail.officialTimeOutHHMM"

      v-bind:status="oneTimeSheetDetail.status"

      v-on:remove="timeSheetDetailData.splice(index, 1)"

      @interface="handleUpdateTimeSheetDataDetail" >

   </v-list-item>

</v-list>

</v-layout>

</div>

</template>

<script>

import moment from 'moment'

import Vue from 'vue'

export default {

  name: 'current-month-timesheet',

  data: function() {

    return{

    currentView: 'current-month-timesheet'  ,

    timeSheetMonth: getCurrentMonthName(),

    timeSheetYear: new Date().getFullYear(),

    timeSheetDetailData: [],

    moment:moment

    }

  },

        mounted: function () {

            this.loadTimeSheetDetailData();

        },

        methods: {

       handleUpdateTimeSheetDataDetail (event) {

             console.log('The retrieved data after child handle is : ', event) // get the data after child dealing

            var selectedTimeSheetDetailId = event;

  const selectedOneTimeSheetDetail = this.timeSheetDetailData.find(d => d.timeSheetDetailId === selectedTimeSheetDetailId)

            console.dir(selectedOneTimeSheetDetail);

            //Had router not defined problem.

            //https://stackoverflow.com/questions/41860578/vue-route-is-not-defined

            this.$router.push({ path: '/update\_timesheet\_detail', params: { oneTimeSheetDetailData: selectedOneTimeSheetDetail}})

        },

            getWeekDayName : function(inWeedDayNumber) {

            var dayNameList = ['Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday',

                'Saturday'];

            console.log(dayNameList[inWeedDayNumber]);

            return dayNameList[inWeedDayNumber];

            },

            getTimeFromMins : function (mins) {

                //https://stackoverflow.com/questions/36035598/how-to-convert-minutes-to-hours-using-moment-js

                // Do not include the first validation check if you want, for example,

                // getTimeFromMins(1530) to equal getTimeFromMins(90) (i.e. mins rollover)

                if (mins >= 24 \* 60 || mins < 0) {

                    throw new RangeError("Valid input should be greater than or equal to 0 and less than 1440.");

                }

                var h = mins / 60 | 0,

                    m = mins % 60 | 0;

                return moment.utc().hours(h).minutes(m).format("hh:mm A");

            },

            updateTimeSheetDetail: function (id) {

                this.$router.push('UpdateTimeSheetDetail')

                //return '/TimeInTimeOutData/UpdateTimeSheetDetail/' + id;

            },

            loadTimeSheetDetailData: function () {

                var that = this;

                var $requestTracker = null;

                $requestTracker = jQuery.ajax({

                    type: 'GET',

                    beforeSend: function (xhr) {

                        xhr.setRequestHeader('RequestVerificationToken', $('input[name=\_\_RequestVerificationToken]').val());

                    },

                    url: '/API/TimeInTimeOutData/GetTimeSheetAndTimeSheetDetails',

                    dataType: 'json',

                    contentType: 'application/x-www-form-urlencoded'

                });

                $requestTracker.done(function (data) {

                    console.dir(data);

       that.timeSheetDetailData = data.timeSheetDetails;

                });

        }

}

}

Vue.component('timesheetdetail-item', {

    data: function() {

    return{

    moment:moment

    }

  },

  template: `

    <li>

       <v-list-tile avatar ripple>

           <v-list-tile-content>

                <v-list-tile-title>{{customerAccountName}}</v-list-tile-title>

                <v-list-tile-sub-title class="grey--text text--darken-4">{{moment(dateOfLesson).format("DD/MM/YYYY dddd")}}</v-list-tile-sub-title>

                <v-list-tile-sub-title>{{officialTimeInHHMM}} to {{officialTimeOutHHMM}} </v-list-tile-sub-title>

                <v-list-tile-sub-title>{{status}} </v-list-tile-sub-title>

              </v-list-tile-content>

    <v-list-tile-action>

    <v-menu offset-y >

      <v-btn primary light slot="activator">Manage</v-btn>

      <v-list>

        <v-list-item>

          <v-list-tile @click.native="gotoUpdate" >

            <v-list-tile-title >Update</v-list-tile-title>

          </v-list-tile>

          <v-list-tile>

            <v-list-tile-title>Get Signature</v-list-tile-title>

          </v-list-tile>

        </v-list-item>

      </v-list>

    </v-menu>

   </v-list-tile-action>

   </v-list-tile>

   <v-divider></v-divider>

      <button v-on:click="$emit('remove')">X</button>

</li>

  `,

  props: ['customerAccountName','dateOfLesson','officialTimeInHHMM','officialTimeOutHHMM','status','timeSheetDetailId'],

  methods: {

     gotoUpdate : function (){

      console.log('The detected time sheet detail data record id is: ' + this.timeSheetDetailId);

      console.log('gotoUpdate method was called and emitting to the parent. At the same time pass the id value');

      // handle data and give it back to parent by interface

      this.$emit('interface', this.timeSheetDetailId);

     }

  }

});

    function getCurrentMonthName(){

        var monthNameList = ["January", "February", "March", "April", "May", "June",

            "July", "August", "September", "October", "November", "December"

        ];

        var currentDate = new Date();

        return monthNameList[currentDate.getMonth()];

    }

</script>

<!-- Add "scoped" attribute to limit CSS to this component only -->

<style scoped>

h1, h2 {

  font-weight: normal;

}

ul {

  list-style-type: none;

  padding: 0;

}

li {

  display: inline-block;

  margin: 0 10px;

}

a {

  color: #42b983;

}

</style>

<template id="update-timesheet-detail-template" >

<h2>Update Timesheet Detail Data</h2>

    <v-card class="grey lighten-4 elevation-0">

    <v-card-text>

      <v-container fluid>

   <v-layout row wrap>

      <v-flex xs12 sm6>

        <v-menu

          lazy

          :close-on-content-click="false"

          v-model="menu2"

          transition="v-scale-transition"

          offset-y

          :nudge-left="40"

        >

          <v-text-field

            slot="activator"

            label="time-in"

            v-model="oneTimeSheetDetailData.actualTimeIn | timeDisplay"

            prepend-icon="access\_time"

            readonly

          ></v-text-field>

          <v-time-picker v-model="oneTimeSheetDetailData.actualTimeIn | timeDisplay"></v-time-picker>

        </v-menu>

      </v-flex>

      <v-flex xs12 sm6>

        <v-dialog

          persistent

          v-model="modal2"

          lazy

        >

          <v-text-field

            slot="activator"

            label="time-out"

            v-model="oneTimeSheetDetailData.actualTimeOut | timeDisplay"

            prepend-icon="access\_time"

            readonly

          ></v-text-field>

          <v-time-picker v-model="oneTimeSheetDetailData.actualTimeOut | timeDisplay" actions>

            <template scope="{ save, cancel }">

              <v-card-row actions>

                <v-btn flat primary @click.native="cancel()">Cancel</v-btn>

                <v-btn flat primary @click.native="save()">Save</v-btn>

              </v-card-row>

            </template>

          </v-time-picker>

        </v-dialog>

      </v-flex>

    </v-layout>

         <v-layout row>

          <v-flex xs4>

            <v-subheader>Comments</v-subheader>

          </v-flex>

          <v-flex xs8>

            <v-text-field

              name="input-7-1"

              label="Provide comments if late or exceeded the official end time period."

              multi-line

            ></v-text-field>

          </v-flex>

        </v-layout>

              <div>

                <v-btn small primary light>Small Button</v-btn>

              </div>

      </v-container>

    </v-card-text>

  </v-card>`

</template>

<script>

import moment from 'moment'

export default {

  name: 'update-timesheet-detail-template',

  data: function() {

    return{

    moment:moment

    }

  },

   props:   {oneTimeSheetDetailData: {

      type: Object,

      default: function () {

        return {actualTimeIn:0, actualTimeOut:0};

      }

    }

    },

   methods: {

   },

   mounted: {

     function(){

       console.log('current information inside the update interface is : ');

     }

   }

}

</script>

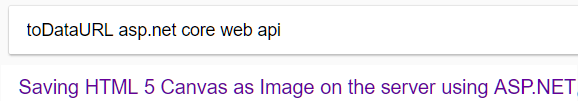
https://github.com/hootlex/vuejs-form-validation-example/blob/master/src/App.vue



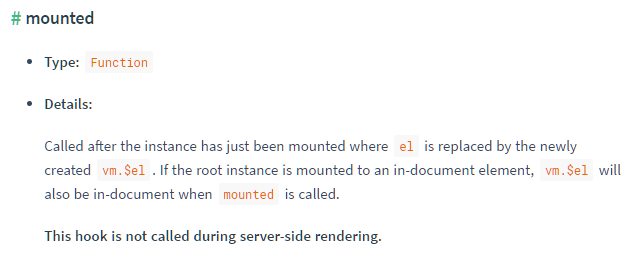
I kept assuming that this **toDataURL()** method is some kind of custom method which is defined inside the **signature\_pad** library. This assumption is wrong.

**https://developer.mozilla.org/en-US/docs/Web/API/HTMLCanvasElement/toDataURL**

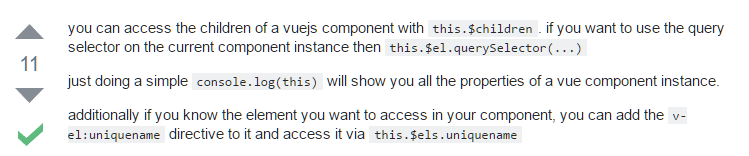
I have been searching for any online content which discusses on leveraging the **signature\_pad** to submit data to server-side API. I could not find any due to lack of concept on **toDataURL** method which is built-in in HTML 5. Finally, when I got my concepts right, I was able to use the following search keywords to do a search for some clues.



Reference: [**http://www.dotnetfunda.com/articles/show/1662/saving-html-5-canvas-as-image-on-the-server-using-aspnet**](http://www.dotnetfunda.com/articles/show/1662/saving-html-5-canvas-as-image-on-the-server-using-aspnet)



https://stackoverflow.com/questions/37104304/vuejs-getting-an-element-within-a-component

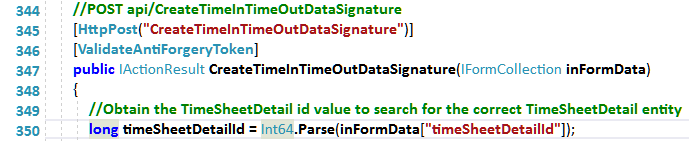


How to reference DOM elements in Vue JS 2.0

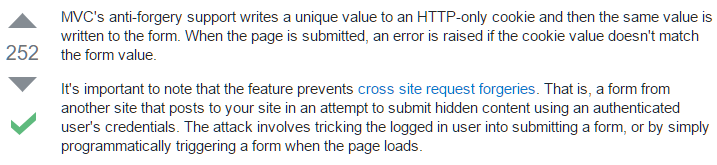
https://stackoverflow.com/questions/36557841/how-to-get-dom-element-of-current-vue-component

**Coping with Anti Cross Site Request Forgery with Asp.Net Core Web API and Vue 2 SPA**

I have programmed the CreateTimeInTimeOutDataSignature Web API method. This Web API method has [ValidateAntiForgeryToken] placed at before the Web API definition.



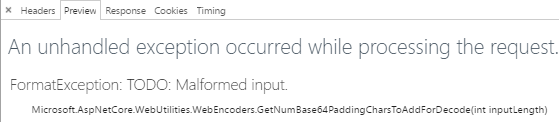
The purpose of **[ValidateAntiForgeryToken]** is explained clearly at [**https://stackoverflow.com/questions/13621934/validateantiforgerytoken-purpose-explanation-and-example**](https://stackoverflow.com/questions/13621934/validateantiforgerytoken-purpose-explanation-and-example).



When I test my Vue JS Single Page Application (SPA) to call the Web API method, I have observed the following error response from the server side.



I checked the server response and noticed that it has something to do with "The antiforgery token could not be decrypted."

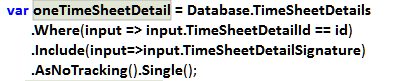




To solve the problem, I found the following solution at [**https://laracasts.com/discuss/channels/vue/vueify-templates-and-laravel-blade**](https://laracasts.com/discuss/channels/vue/vueify-templates-and-laravel-blade) which discusses the similar issue. Due to my past experience in PHP Laravel, I know that the PHP Laravel also implements the same anti forgery technique as the Asp.NET core web api. Which means the solution is relevant to my problem.



The following command is only stable if each TimeSheetDetail entity has one TimeSheetDetailSignature entity. Or else, a runtime error sequence has no elements will occur.



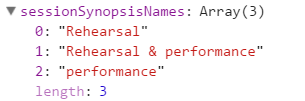
**Viewport Size**

The following article is excellent for view port size concepts.

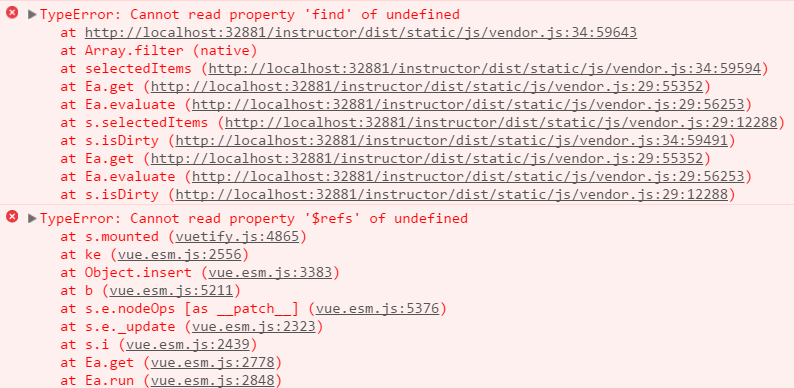
Reference: **https://ericportis.com/posts/2014/srcset-sizes/**

https://jsfiddle.net/mani04/bgzhw68m/

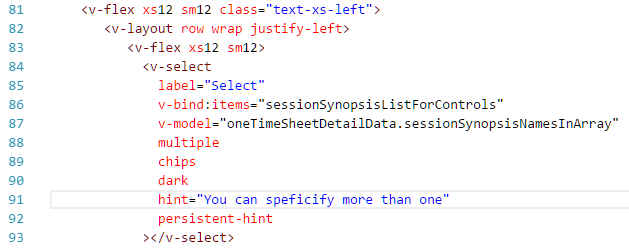
Data binding for the

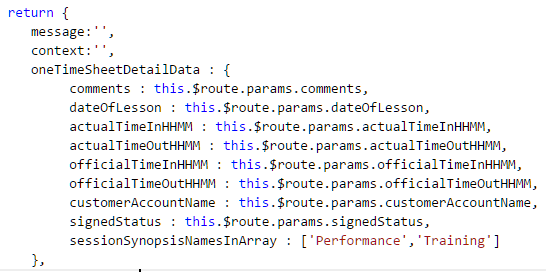


You need to get used to the following error messages which appear frequently within the Chrome DevTool's console output interface.

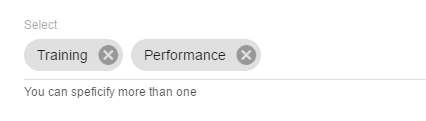


Refer to the code snippet below. I should have used **v-model=" oneTimeSheetDetailData .sessionSynopsisNamesInArray"** for the data binding**.** I did not use the **oneTimeSheetDetailData** inside the v-model. I used **v-model="sessionSynopsisNamesInArray"**. The Vue engine could not find it and raised the above error messages.





I hardcode the array to initialize the Vuetify's Select component so that I can have a feel how the Vuetify Select component should be initialized.





When you code the logic to submit data to the server-side Web API, you will most likely trying to provide code to access the Vue's data object. The following two commands work.

console.dir(this.$data.oneTimeSheetDetailData.sessionSynopsisNamesInArray);

console.dir(this.oneTimeSheetDetailData.sessionSynopsisNamesInArray);

The Vuetify library's DataTable requires the following JavaScript fundamentals:

JavaScript Promise concept

I had problems trying to bind data to the DataTable Vue component. The databinding works, if I hardcode an Array of objects to bind. The databinding breaks when I attempt to perform AJAX request to grab data from the server.

https://www.twilio.com/blog/2015/10/asyncawait-the-hero-javascript-deserved.html

Testing Update Timesheet detail and Signature functionality on 20 June 2017

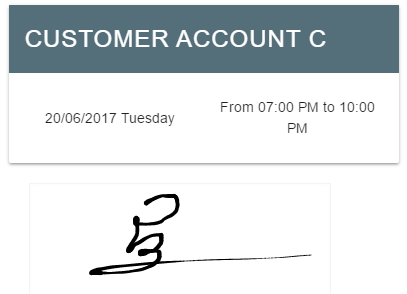
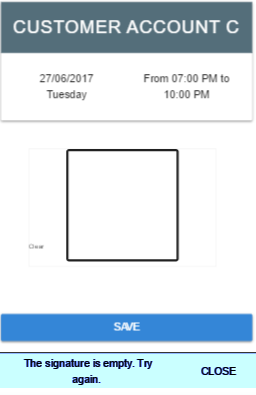
1. When the Save button of the time picker Vue component is clicked, somehow the event can propagate up and triggered the form submission logic. As a result, the data is saved. (Tested again to reproduce the problem, but unable to reproduce the problem)

2. Once the timesheet detail data is signed, the update of time-in and time-out functionality should be disabled or changed to view mode. (Done)

3. After giving the time-in and time-out data, the toast message only appear once when the Save button is clicked to submit form data. The toast message does not appear for the second time. (Fixed)

4. There is a possibility that the user can accidentally provide AM value instead of the PM value. Need to think of a way to reduce the human error.

5. Refer to the figure below, the canvas will lose the digital signature if a resize action occur. This is likely to happen in the actual use case. At first, I thought that the digital data is still there. The digital data is *lost* when *resize* occurs. I have tested it by resizing to make the signature disappear first, and then save the form. When I called an API to check the signature, the signature is a white image.



Canvas losing the digital data due to resize is difficult to fix. But, I was able to reduce the human error, I have referred to <http://jsfiddle.net/amaan/rX572/> to do validation on the canvas. (Done)

6. Need to show the actual time-in and time-out data above the signature canvas. During development, I have provided code to display only the official time-in and time-out because the update timesheet detail functionality was not developed yet. (Fixed)

The following will not work.

<img src="**{{imageUrl}}**" />

<img **:src**="imageUrl" />

<canvas id='editor' style='border:solid'></canvas>

<canvas id='blank' style='display:none'></canvas>

<button id='save'>Save</button>

<script>

canvas = document.getElementById('editor');

ctx = canvas.getContext('2d');

canvas.addEventListener('mousemove',function(e){

ctx.lineTo(e.pageX,e.pageY);

ctx.stroke();

});

document.getElementById('save').addEventListener('click',function(){

if(canvas.toDataURL() == document.getElementById('blank').toDataURL())

alert('It is blank');

else

alert('Save it!');

});

</script>