# مستندات پروژه SPL – مراحل ساخت و تولید

## VueJs

مراحل ساخت و تولید پروژه SPL با استفاده از Vue js بوده و فرایند به صورت تولید صفحات در سمت کلاینت می باشد. Vue js به عنوان یک فریم ورک بر بستر JavaScript (همانند رقبای خود – React و Angular) می تواند کنترل کاملی روی صفحه HTML و نحوه ارسال/دریافت اطلاعات به/از سرور را در اختیار برنامه نویس قرار دهد. نکته مهم این است که این فریم ورک بهترین نکات مربوط به دو رقیب خود را درون خود گنجانده و مشکلات آنها را نیز بر طرف کرده است. سرعت رو به رشد استفاده از این فریم ورک خود دلیل موجهی بر قدرت و پایداری آن است. این فریم ورک نیاز به NodeJs به عنوان پلتفرم دارد.

## IntelliJ WebStorm

این IDE از آنجا که پشتیبانی برتری نسبت به زبان JavaScript دارد به عنوان محیط برنامه نویسی انتخاب شده است. البته این نکته قابل ذکر است که برای برنامه نویسی در پلتفرم NodeJs نیاز به هیجگونه IDE نمی باشد و برنامه نویس صرفا با فایل های txt و js نیز می تواند نرم افزار را تولید کند. از این رو این محیط برنامه نویسی صرفا به خاطر قابلیت های تسهیل دهنده کار برنامه نویس مانند intellisence, debug, search و ... استفاده می شود.

## مراحل تولید

در اینجا ساخت پروژه SPL قدم به قدم توضیح داده شده است. البته نکته مهم این است که سورس کد جاری ممکن است با این مستندات تفاوتهایی داشته باشد، از این رو حتما سورس کد را از مسیر زیر در github دانلود کرده و مطالعه نمایید. هدف این سند ارائه دانش نحوه تولید پروژه SPL در بستر وب و موبایل میباشد.

### نصب VueJs

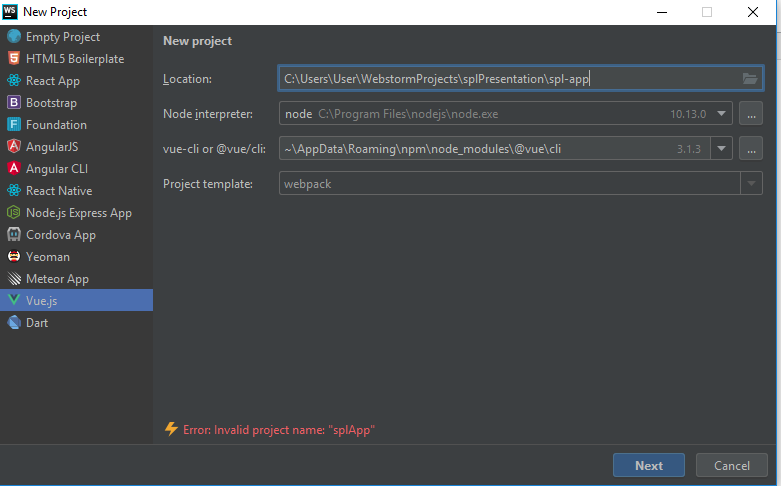
ابتدا دستور زیر را در cmd اجرا میکنیم:

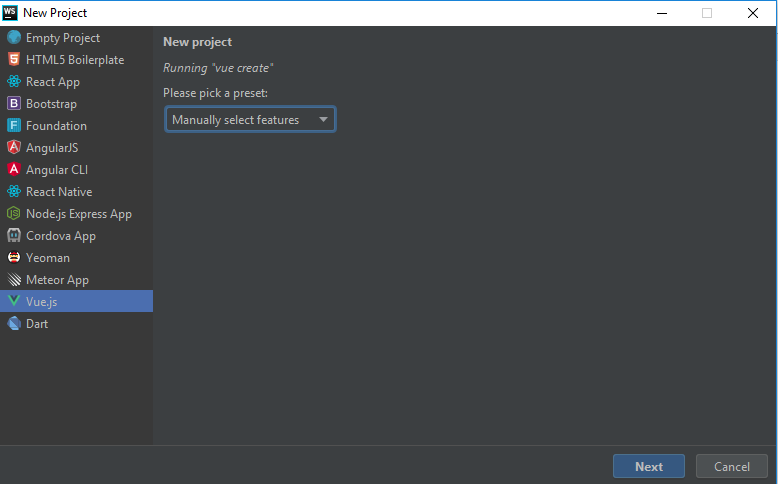
npm i -g @vue/cli

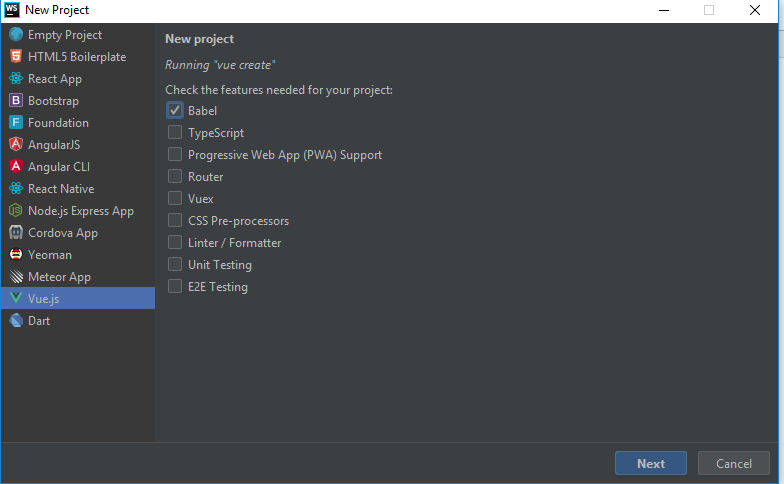
### ایجاد پروژه

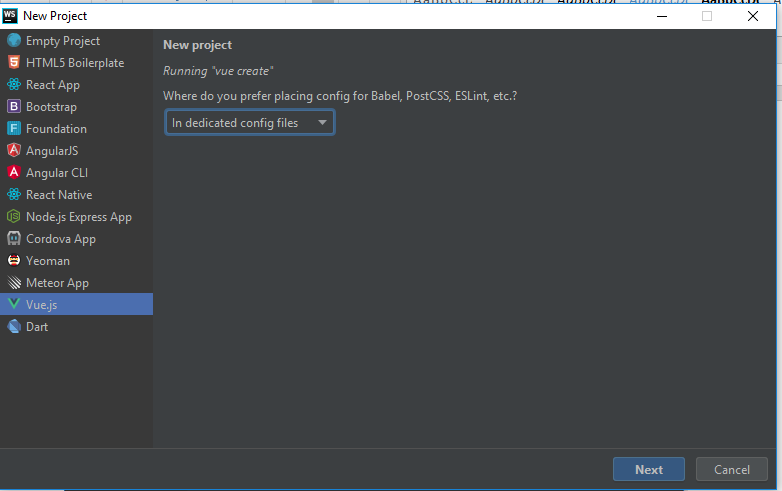
مطابق شکل زیر پروژه جدیدی در Webstorm از نوع Vue ایجاد میکنیم.

ورژن ما از نود 10.13.0 و برای Vue 3.1.3 میباشد.



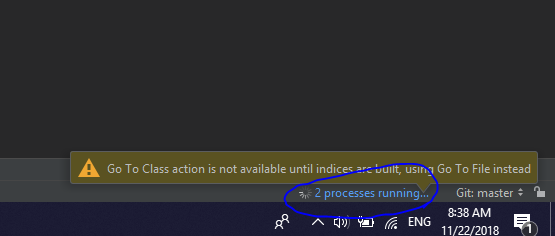




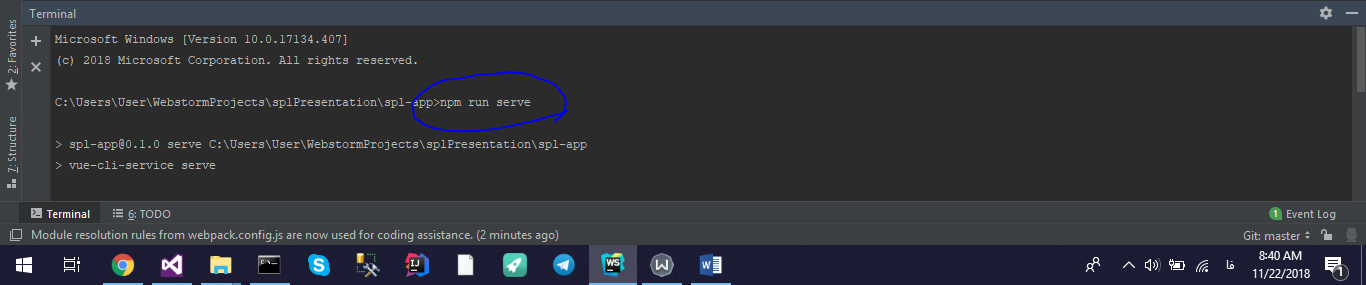


بقیه مراح را Next میزنیم تا پروژه ساخته شود.

سپس صبر مکنیم تا 2 پروسس زیر تکمیل شوند



اکنون پروژه VueJs ما آماده است و با دستور زیر اجرا میشود و با آدرس <http://localhost:8080> قابل دسترس است: (این دستور عملا در فایل package.json اسکریپت با نام serve را برای ما در کنسول اجرا می کند)

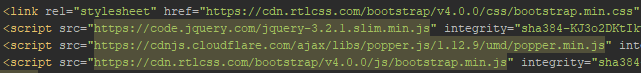


npm run serve

جهت آموزش و یادگیری Vue سایت رسمی <https://cli.vuejs.org/guide/creating-a-project.html> پیشنهاد میشود.

### استفاده از Bootstrap در پروژه

فایل index.html در پوشه public را باز کرده و پیشنیاز های bootstrap را به آن اضافه کنید.



اکنون bootstrap به پروژه اضافه شده و میتوانید از کلاس های آن استفاده کنید.

### مدل های استفاده شده

مدل های استفاده شده شامل یکسری کلاس (member+methods) هستند که عمده آنها در زمان تولید فرم و واکشی اطلاعات از سمت سرور مقدار دهی میشوند. این مدل ها باید در پوشه src\models به صورت زیر ایجاد شوند:

#### کلاس QuestionModel

شامل کلیه اطلاعات مربوط به تعریف سوال (خصوصیت) است. در زمان ساخت فرم و سوال به صورت html از اطلاعات این کلاس استفاده می شود.

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| title | string | عنوان سوال |
| fieldInfo | FieldInfoModel | نحوه نمایش (select,date,text) |
| validation | DelegateModel | اسکریپتی که خروجی ValidationResult بر میگرداند |
| exited | DelegateModel | اسکریپتی که خروجی CommandResult[] بر میگرداند |
| exiting | DelegateModel | اسکریپتی که خروجی Boolean برمیگرداند |
| entered | DelegateModel | اسکریپتی که خروجی CommandResult[] بر میگرداند |
| entering | DelegateModel | اسکریپتی که خروجی Boolean برمیگرداند |
| changed | DelegateModel | اسکریپتی که خروجی CommandResult[] بر میگرداند |
| changing | DelegateModel | اسکریپتی که خروجی Boolean برمیگرداند |

**import** {FieldInfoModel} **from** "@/models/FieldInfo/FieldInfoModel";  
**import** {DelegateModel} **from** "@/models/DelegateModel";  
**import** {TextFieldInfoModel} **from** "@/models/FieldInfo/TextFieldInfoModel";  
**import** {SelectFieldInfoModel} **from** "@/models/FieldInfo/SelectFieldInfoModel";  
**import** {DateFieldInfoModel} **from** "@/models/FieldInfo/DateFieldInfoModel";  
  
**export class** QuestionModel {  
 constructor() {  
 **this**.title = '';  
 **this**.fieldInfo = **null**;//FieldInfoModel  
 **this**.validation = **null**;//DelegateModel  
 **this**.exitted = **null**;//DelegateModel  
 **this**.exitting = **null**;//DelegateModel  
 **this**.entered = **null**;//DelegateModel  
 **this**.entering = **null**;//DelegateModel  
 **this**.changed = **null**;//DelegateModel  
 **this**.changing = **null**;//DelegateModel  
 }  
 deserialize(input){  
 **this**.title = input.title;  
 **if**(input.fieldInfo.name === 'text')  
 **this**.fieldInfo = **new** TextFieldInfoModel().deserialize(input.fieldInfo);  
 **else if**(input.fieldInfo.name === 'date')  
 **this**.fieldInfo = **new** DateFieldInfoModel().deserialize(input.fieldInfo);  
 **else if**(input.fieldInfo.name === 'select')  
 **this**.fieldInfo = **new** SelectFieldInfoModel().deserialize(input.fieldInfo);  
 **this**.validation = **new** DelegateModel().deserialize(input.validation);  
 **this**.exitted = **new** DelegateModel().deserialize(input.exitted);  
 **this**.exitting = **new** DelegateModel().deserialize(input.exitting);  
 **this**.entered = **new** DelegateModel().deserialize(input.entered);  
 **this**.entering = **new** DelegateModel().deserialize(input.entering);  
 **this**.changed = **new** DelegateModel().deserialize(input.changed);  
 **this**.changing = **new** DelegateModel().deserialize(input.changing);  
 **return this**;  
 }  
}

#### کلاس پایه FieldInfoModel

اطلاعات نحوه نمایش یک سوال در این کلاس گنجانده شده است. برای مثال نوع کنترل نمایشی (تاریخ، انتخابی، متنی و ...) و خصوصیات مرتبط با هرکدام از آنها.

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| name | string | نوع داده کنترلی سوال – date,text,select,… |
| style | JSON | تعریف ظاهر سوال - مثال  {  "background":"gray",  "font-family":"B Nazanin",  "font-weight":"bold" } |
| deserialize | method | این تابع داده خوانده شده از دیتا بیس را به این کلاس تبدیل میکند – به عبارتی JSON ورودی را تبدیل به FieldInfoModel میکند |

**export class** FieldInfoModel {  
 constructor() {  
 **this**.name = '';  
 **this**.style = {};  
 }  
 deserialize(input){  
 **if**(input == **null**)  
 **return null**;  
 **this**.name = input.name;  
 **this**.style = input.style;  
 **return this**;  
 }  
}

نکته: این کلاس یک کلاس پایه است و نباید از آن شیء ایجاد شود، و تنها باید از مشتقات آن که در ادامه توضیح داده شده است استفاده شود.

#### کلاس TextFieldInfoModel (ارث میبرد از FieldInfoModel)

مربوط به سوال های از نوع متنی است، هر کنترل متنی ممکن است متغیر های مخصوص خود داشته باشد، مانند mask که در این کلاس تعریف میشوند.

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| mask | string | فرمت نمایش متن |
| deserialize | method | این تابع داده خوانده شده از دیتا بیس را به این کلاس تبدیل میکند – به عبارتی JSON ورودی را تبدیل به TextFieldInfoModel میکند |

**import** {FieldInfoModel} **from** "@/models/FieldInfo/FieldInfoModel";  
  
**export class** TextFieldInfoModel **extends** FieldInfoModel {  
 constructor() {  
 **super**();  
 **this**.mask = '';  
 }  
 deserialize(input){  
 **if**(input == **null**)  
 **return null**;  
 **super**.deserialize(input);  
 **this**.mask = input.mask;  
 **return this**;  
 }  
}

#### کلاس DateFieldInfoModel (ارث میبرد از FieldInfoModel)

مربوط به سوال های از نوع تاریخ است، هر کنترل تاریخی ممکن است متغیر های مخصوص خود داشته باشد، مانند Format تاریخ و یا نوع تقویم (میلادی، شمسی و ...) که در این کلاس تعریف میشوند.

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| format | string | فرمت نمایش تاریخ – YYYY/MM/DD |
| deserialize | method | این تابع داده خوانده شده از دیتا بیس را به این کلاس تبدیل میکند – به عبارتی JSON ورودی را تبدیل به DateFieldInfoModel میکند |

**import** {FieldInfoModel} **from** "@/models/FieldInfo/FieldInfoModel";  
  
**export class** DateFieldInfoModel **extends** FieldInfoModel {  
 constructor() {  
 **super**();  
 **this**.format = 'yyy-mm-dd';  
 }  
  
 deserialize(input){  
 **if**(input == **null**)  
 **return null**;  
 **super**.deserialize(input);  
 **this**.format = input.format;  
 **return this**;  
 }  
}

#### کلاس SelectFieldInfoModel (ارث میبرد از FieldInfoModel)

مربوط به سوال های از نوع تاریخ است، هر کنترل تاریخی ممکن است متغیر های مخصوص خود داشته باشد، مانند Format تاریخ و یا نوع تقویم (میلادی، شمسی و ...) که در این کلاس تعریف میشوند.

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| source | SourceModel | نوع *منبع* داده را تعریف می کند. |
| isMultiSelect | Boolean | آیا فیلد ما چند انتخابی است؟ |
| isFree | Boolean | آیا میتوان عناصر جدید و دلخواه به منبع اضافه و انتخاب کرد |
| deserialize | method | این تابع داده خوانده شده از دیتا بیس را به این کلاس تبدیل میکند – به عبارتی JSON ورودی را تبدیل به SelectFieldInfoModel میکند |

**import** {FieldInfoModel} **from** "@/models/FieldInfo/FieldInfoModel";  
**import** {SourceModel} **from** "@/models/FieldInfo/SourceModel";  
  
**export class** SelectFieldInfoModel **extends** FieldInfoModel {  
 constructor() {  
 **super**();  
 **this**.source = **null**;  
 **this**.isMultiSelect = **false**;  
 **this**.isFree = **false**;  
 }  
 deserialize(input){  
 **if**(input == **null**)  
 **return null**;  
 **super**.deserialize(input);  
 **this**.isFree = input.isFree;  
 **this**.isMultiSelect = input.isMultiSelect;  
 **this**.source = **new** SourceModel().deserialize(input.source);  
 **return this**;  
 }  
}

#### کلاس SourceModel

نوع *منبع* داده را تعریف می کند.

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| url | string | ممکن است منبع داده یک سرور یا Web APi باشد |
| script | string | ممکن است منبع داده ثابت و سمت کلاینت باشد، مانند جنسیت  که به صورت زیر تعریف میشود:  [  {id:1,name:'مدکر'},  {id:1,name:'مونث'},  {id:1,name:'ناشناس'}, ] |
| refreshTime | int | مدت زمان بروزرسانی سورس به میلی ثانیه. مقدار -1 یعنی عدم بروزرسانی دوره ای و عدد 0 یعنی بروز رسانی در زمان ورود به فیلد. |
| deserialize | method | این تابع داده خوانده شده از دیتا بیس را به این کلاس تبدیل میکند – به عبارتی JSON ورودی را تبدیل به SourceModel میکند |

**export class** SourceModel {  
 constructor() {  
 **this**.url = '';  
 **this**.script = '';  
 **this**.refreshTime = -1;  
 }  
 deserialize(input){  
 **if**(input == **null**)  
 **return null**;  
 **this**.refreshTime = input.refreshTime;  
 **this**.script = input.script;  
 **this**.url = input.url;  
 **return this**;  
 }  
}

#### کلاس DelegateModel

این کلاس مسئول اجرای اسکریپت خوانده شده و بر گرداندن مقدار نهایی آن را بر عهده دارد. کلاس های دیگری که از DelegateModel استفاده می کنند مطمئننا خروجی های متفاوت و مد نظر خود را انتظار دارند که باید رعایت شود – مثلا Validation انتظار خروجی ValidationResult و exited انتظار خروجی CommandResult[] را دارد.

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| script | string | اسکریپت اجرا شده سمت کلاینت میباشد و مثلا برای validation به صورت زیر است  (**function** () {  **if** (**this**.value == **null** || **this**.value === '')  **return** {result:**false**,message:'این فیلد اجباری است'}; })() |
| exec | method | مقدار script را اجرا کرده و نتیجه نهایی را بر می گرداند |
| deserialize | method | این تابع داده خوانده شده از دیتا بیس را به این کلاس تبدیل میکند – به عبارتی JSON ورودی را تبدیل به SourceModel میکند |

**export class** DelegateModel{  
 constructor(){  
 **this**.script='';  
 }  
 exec(){  
 **return** eval(**this**.script);  
 }  
 deserialize(input){  
 **if**(input == **null**)  
 **return null**;  
 **this**.script = input.script;  
 **return this**;  
 }  
}

#### کلاس GroupModel

شامل کلیه اطلاعات مربوط به تعریف گروه سوال است. در زمان ساخت فرم چیدمان آن به صورت html از اطلاعات این کلاس استفاده می شود.

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| title | string | عنوان گروه |
| fieldInfo | FieldInfoModel | نحوه نمایش (select,date,text) |
| questions | QuestionModel[] | سوالات این گروه |
| groupValues | GroupValue[] | مقادیر این گروه – در صورتی که گروه جدولی باشد این آرایه 0 یا n عنصر دارد و در صورتی که گروه فرمی باشد این آرایه همواره یک عنصر دارد. |
| init | method | مقدار دهی اولیه یک گروه |
| deserialize | method | این تابع داده خوانده شده از دیتا بیس را به این کلاس تبدیل میکند و تک تک سوالات را نیز deserialize می کند – به عبارتی JSON ورودی را تبدیل به GroupModel میکند |

**import** {QuestionModel} **from** "@/models/QuestionModel";  
**import** {GroupValue} **from** "@/models/GroupValue";  
  
**export class** GroupModel {  
 constructor() {  
 **this**.title = '';  
 **this**.questions = [];//Question[]  
 **this**.groupValues = [];//GroupValue[]  
 }  
  
 init() {  
 **let** groupValue = **new** GroupValue(**this**);  
 **this**.groupValues.push(groupValue)  
 }  
  
  
 deserialize(input) {  
 **if** (input == **null**)  
 **return null**;  
 **this**.title = input.title;  
 **for** (**let** q **of** input.questions)  
 **this**.questions.push(**new** QuestionModel().deserialize(q));  
 **return this**;  
 }  
}

#### کلاس MenuModel

شامل کلیه اطلاعات مربوط به تعریف یک منو است. در زمان ساخت پنجره و چیدمان گروه ها به صورت html از اطلاعات این کلاس استفاده می شود.

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| title | string | عنوان منو |
| groups | GroupModel[] | گروه های این منو |
| questions | QuestionModel[] | سوالات این گروه |
| init | method | مقدار دهی اولیه یک منو |
| deserialize | method | این تابع داده خوانده شده از دیتا بیس را به این کلاس تبدیل میکند و تک تک گروه ها را نیز deserialize می کند – به عبارتی JSON ورودی را تبدیل به MenuModel میکند |

**import** {GroupModel} **from** "@/models/GroupModel";  
  
**export class** MenuModel {  
 constructor() {  
 **this**.title = "";  
 **this**.groups = []  
 }  
 init(){  
 **for**(**let** g **of this**.groups)  
 g.init();  
 }  
 deserialize(input){  
 **if**(input == **null**)  
 **return null**;  
 **this**.title = input.title;  
 **for**(**let** g **of** input.groups)  
 **this**.groups.push(**new** GroupModel().deserialize(g));  
 **return this**;  
 }  
}

#### کلاس spl\api

مسئول ارتباط با سرور WebApi را بر عهده دارد و کلیه درخواست ها به سرور از این کلاس استفاده می کند.

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| menu | MenuModel | خواندن اطلاعات مربوط به منوی خاص از WebApi |

**import** {GroupModel} **from** "@/models/GroupModel";  
**import** {QuestionModel} **from** "@/models/QuestionModel";  
**import** {MenuModel} **from** "@/models/MenuModel";  
  
**export class** spl {  
 **static** *menu*(menuId) {  
 **var** m =  
 {  
 title:'منوی افراد',  
 groups:[  
 {  
 title:'اطلاعات اولیه',  
 questions:[  
 {  
 title: 'نام',  
 fieldInfo:{  
 name:'text',  
 style:**null**,  
 mask:''  
 },  
 validation:{  
 script:'abc'  
 }  
 },{  
 title: 'فامیل',  
 fieldInfo:{  
 name:'text',  
 style:**null**,  
 mask:''  
 },  
 validation:{  
 script:'abc'  
 }  
 },{  
 title: 'تولد',  
 fieldInfo:{  
 name:'date',  
 style:**null**,  
 mask:''  
 },  
 validation:{  
 script:'abc'  
 }  
 },{  
 title: 'محل تولد',  
 fieldInfo:{  
 name:'select',  
 style:**null**,  
 mask:''  
 },  
 validation:{  
 script:'abc'  
 }  
 }  
 ]  
 },  
 {  
 title:'اطلاعات پرسنلی',  
 questions:[  
 {  
 title: 'شماره پرسنلی',  
 fieldInfo:{  
 name:'text',  
 style:**null**,  
 mask:''  
 },  
 validation:{  
 script:'abc'  
 }  
 },{  
 title: 'سمت',  
 fieldInfo:{  
 name:'text',  
 style:**null**,  
 mask:''  
 },  
 validation:{  
 script:'abc'  
 }  
 }  
 ]  
 }  
 ]  
 };  
 **let** menu = **new** MenuModel().deserialize(m);  
 menu.init();  
 **return** menu;  
 }  
  
}

#### کلاس GroupValue

شامل کلیه اطلاعات مربوط به مقادیر یک ردیف از گروه است. گروه هایی که به صورت جدولی هستند دارای 0 یا n شیء از این کلاس هستند. گروه هایی که به صورت فرمی هستند دارای یک شیء از این کلاس هستند.

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| questionValues | questionValue[] | آرایه ای از مقدار سوالات این گروه |

**import** {QuestionValue} **from** "@/models/QuestionValue";  
  
**export class** GroupValue{  
 constructor(groupModel){  
 **this**.questionValues = []; //QuestionValue[]  
 **for**(**let** q **of** groupModel.questions){  
 **let** questionValue = **new** QuestionValue(q);  
 **this**.questionValues.push(questionValue);  
 }  
 }  
}

#### کلاس QuestionValue

شامل داده مربوط به یک سوال است.

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| value | string | مقدار یک سوال |
| oldValue | string | مقدار قبلی سوال |
| questionModel | QuestionModel | تعریف سوال مربوطه – که در زمان بارگذاری فرم از سرور خوانده میشود. |

**export class** QuestionValue{  
 constructor(questionModel){  
 **this**.value = '';  
 **this**.oldValue = '';  
 **this**.questionModel = questionModel;//QuestionModel  
 }  
}

### کامپوننت ها

کامپوننت در Vue عبارت است از یک UserControl که شامل موارد زیر است:

template: نحوه نمایش کامپوننت به صورت HTML+JavaScript

data: داده های اطلاعاتی استفاده شده در کامپوننت

props: متغیر هایی که کامپوننت پدر فراهم کرده است

methods: توابع مورد نیاز، مثلا onClick چه عملیاتی صدا زده شود

computed: فیلد های محاسباتی – مثلا FullName = FirstName+’ ‘+LastName

#### کامپوننت MenuComponent

مسئول چیدمان گروه های منو روی صفحه است.

**data**:

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| menu | MenuModel | منوی در حال نمایش |
| selectedGroup | GroupModel | گروه انتخاب شده |

**let** context = {  
 menu: **new** spl.*menu*(1),  
 selectedGroup: **null**};

data() {  
 **return** context;  
}

**methods:**

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| setSelectedGroup |  | انتخاب گروه در حال نمایش (در زمان کلیک روی گروه صدا زده می شود) |

methods: {  
 setSelectedGroup(g) {  
 context.selectedGroup = g;  
 }  
}

#### کامپوننت GroupComponent

مسئول چیدمان سوالات یک گروه فرمی در صفحه است.

**data**:

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |

**props:**

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| groupModel | GroupModel | گروهی که باید نمایش داده شود |

**let** context = {  
 menu: **new** spl.*menu*(1),  
 selectedGroup: **null**};

data() {  
 **return** context;  
}

**methods:**

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| save |  | ذخیره گروه |

methods: {  
 setSelectedGroup(g) {  
 context.selectedGroup = g;  
 }  
}

#### کامپوننت TextFieldComponent

**props:**

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| questionValue | QuestionValueModel | مقدار سوالی که باید نمایش داده شود |

**Template:**

<template>  
 <div>  
 <label class="control-label">{{questionValue.questionModel.title}}</label>  
 <input class="form-control" v-model="questionValue.value"/>  
 </div>  
</template>

#### کامپوننت SelectFieldComponent

**props:**

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| questionValue | QuestionValueModel | مقدار سوالی که باید نمایش داده شود |

**Template:**

<template>  
 <div>  
 <label class="control-label">{{questionValue.questionModel.title}}</label>  
 <select class="form-control" v-model="questionValue.value">  
 <option value="1">مقدار 1</option>  
 <option value="2">مقدار 2</option>  
 </select>  
 </div>  
</template>

#### کامپوننت DateFieldComponent

**props:**

|  |  |  |
| --- | --- | --- |
| عضو | نوع داده | توضیحات |
| questionValue | QuestionValueModel | مقدار سوالی که باید نمایش داده شود |

**Template:**

<template>  
 <div>  
 <label class="control-label">{{questionValue.questionModel.title}}</label>  
 <input class="form-control" type="date" v-model="questionValue.value"/>  
 </div>  
</template>

### رابطه ها و نمودار ها

#### رابطه میان QuestionModel, QuestionValue,GroupModel, GroubValue

نکته: با وجود اینکه ممکن است از کلاس QuestionValue چندین شیئ در حافظه بوجود آید (به اندازه ورودی های کاربر)، همواره تنها یک شئ برای QuestionModel در حافظه وجود خواهد داشت.

