**به نام خدا**

**موضوع: مستند مربوط به پروژه Front-End برای Gita expert**

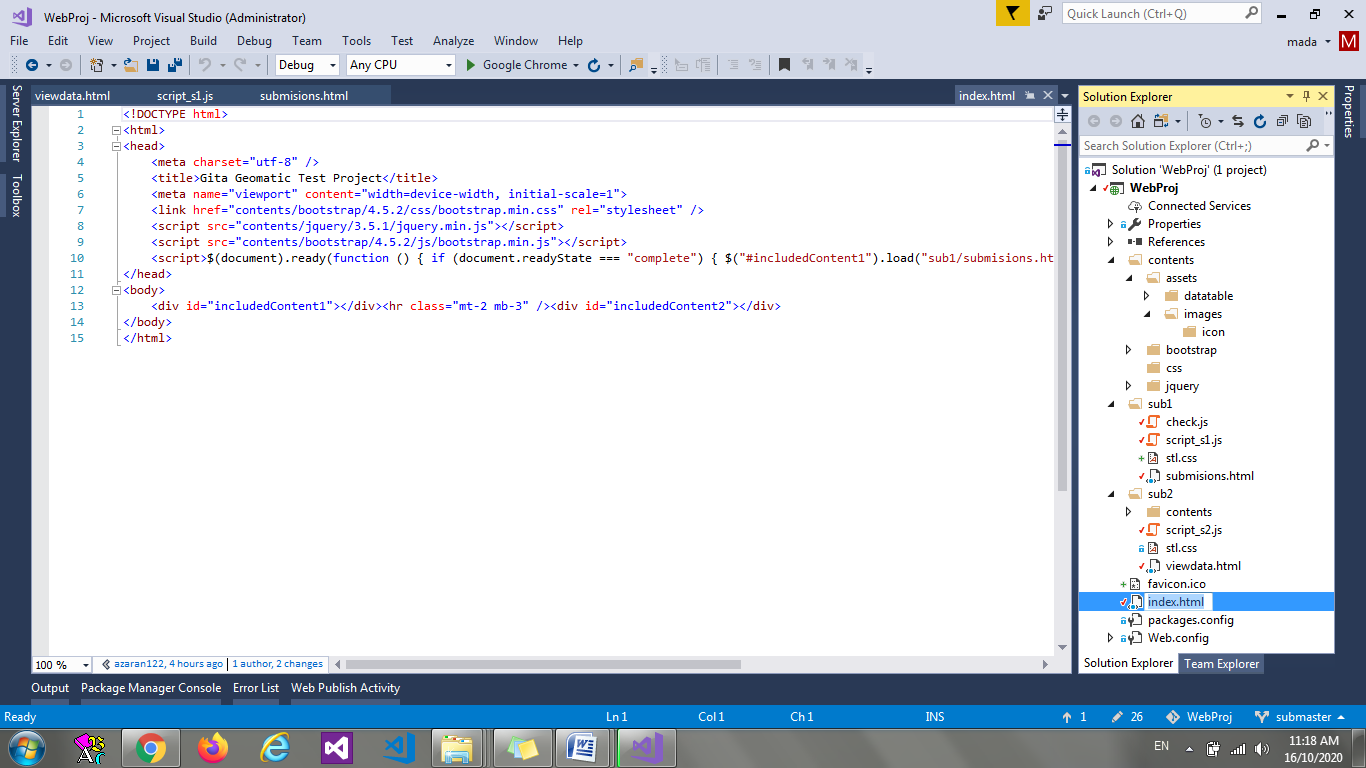
مستند حاضر به توضیح در مورد کد به کار رفته در این پروژه می­پردازد:

1. **صفحه index**
2. **پلاگین ورود اطلاعات**
3. **پلاگین نمایش اطلاعات**

\*\*\*

1. **صفحه index :**

در کد مربوط به index.html ، فایلهای css , js مربوط به bootstrap و jquery را import کرده­ایم. از آنجا که از بوت­استرپ و jquery در تمام پروژه استفاده می­شود و در صفحه index نیز تمام پلاگینها باید در همان صفحه به­کار روند، بهتر است در index.html عمل import انجام شود.

**در قسمت body:**

در دو تگ div پلاگین مربوط به بخش ورود اطلاعات با **id=includeConten1** و پلاگین بخش نمایش اطلاعات با **id=includeContetn2**  اضافه شده­اند.

جهت ایجاد یک جدا کننده در صفحه از نظر نمایش بهتر، یک خط افقی در میان این دو بخش ایجاد میکنیم: تگ hr.

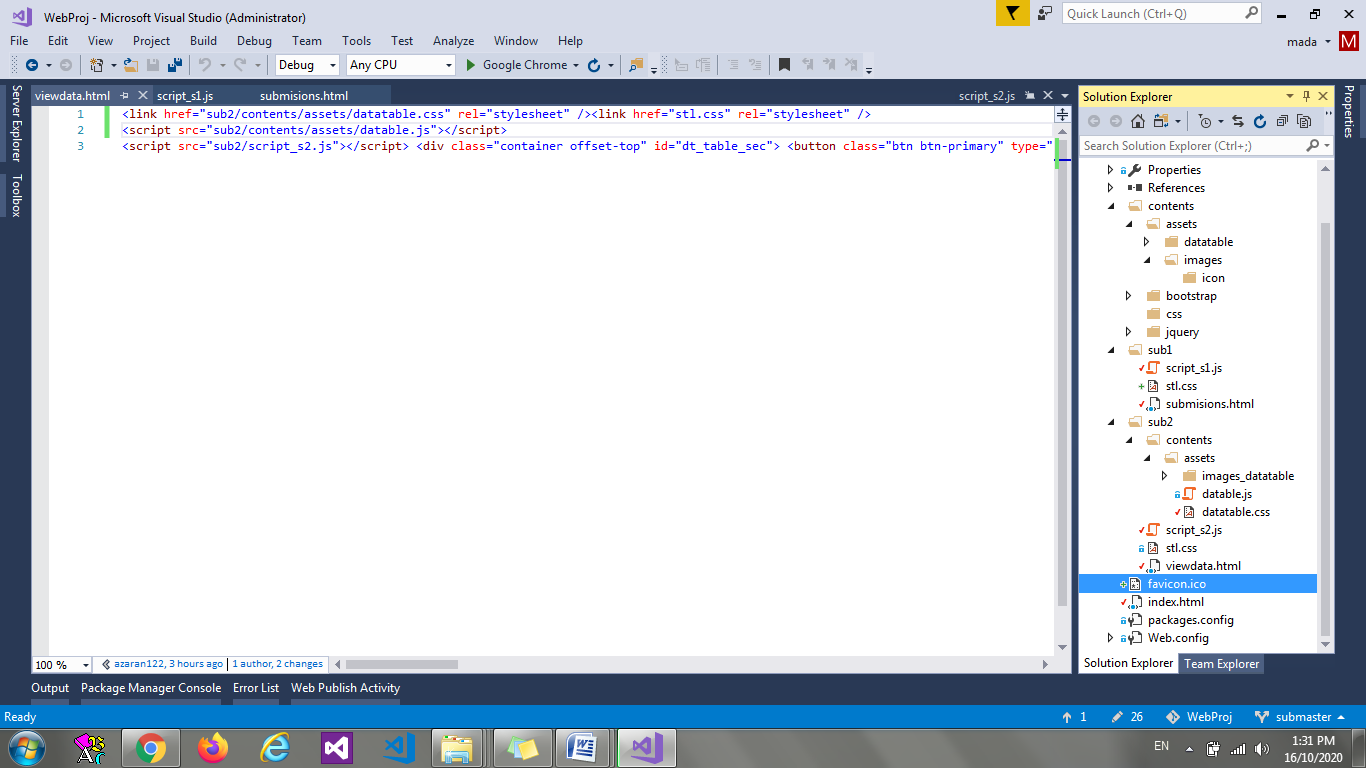
پلاگینها را در فولدر sub1 و sub2 به همراه فایلهای html/css/js مربوط به خود درنظر گرفتیم.

از آنجا که باید پلاگینها در index.html لود شوند و نه مستقل؛ بنابراین مسیردهی به فایلهای js و css و دیگرفایلهای احتمالی در این پلاگینها نسبت به مسیری است که در فایل index.html بدهیم نه نسبت به مسیری که در خود فایلهای پلاگین بدهیم.

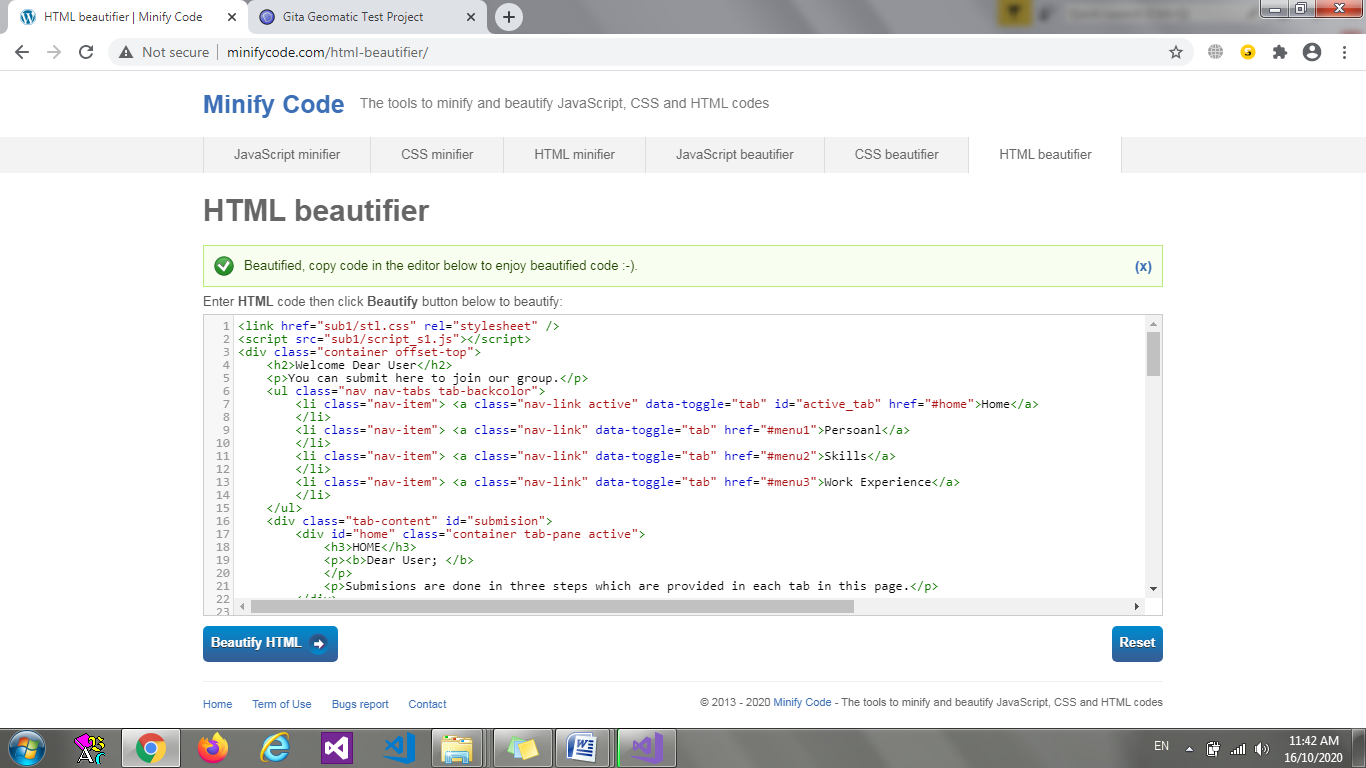
\*\*\*

**2-پلاگین ورود اطلاعات**

این پلاگین توسط submission.html پیاده سازی شده و به همراه script\_s1.js , stl.css می باشد.



**پیاده سازی بخش tab:**



به این ترتیب 4 div با id های Home, menu1,menu2,menu3 در ادامه کد داریم.

آیتم هایی که قرار است در این فرم تکمیل شود هریک با id مشخص شده­اند. مانند ایمیل id=” email”

از کنترلر بوت استرپ جهت کنترل خالی نبودن فیلدها استفاده نموده ایم که باrequired در تگها مشخص شده­اند.

جهت کنترل یکسان بودن پسورد، مقایسه میکنیم که همانند باشند، این کنترل با استفاده از رخدادoninput در تگ form اول انجام می­شود. اگر همانند باشند این دو فیلد مربوط به پسورد سبزرنگ نمایش داده می­شوند.

ثبت فرم: توسط رویداد onclick بر روی دکمه submit انجام می­شود. از آنجا که تابع مربوطه به شکل ماژول نوشته شده است فراخوانی آن به شکل: $(this).sendform() میباشد.

**ارسال اطلاعات توسط ajax در script\_s1.js**

$(this).sendform()

متد ارسال: Post ؛

ارسال داده در فیلد data به شکل json ؛

و اعمال Json.stringify بر روی داده ارسالی به منظور ارسال object در قالب یک رشته.

url فرضی جهت فراخوانی api : <http://localhost:8080/action/method>

در صورت موفقیت آمیز بودن دریافت پاسخ یعنی کد200 : success اجرا می شود.

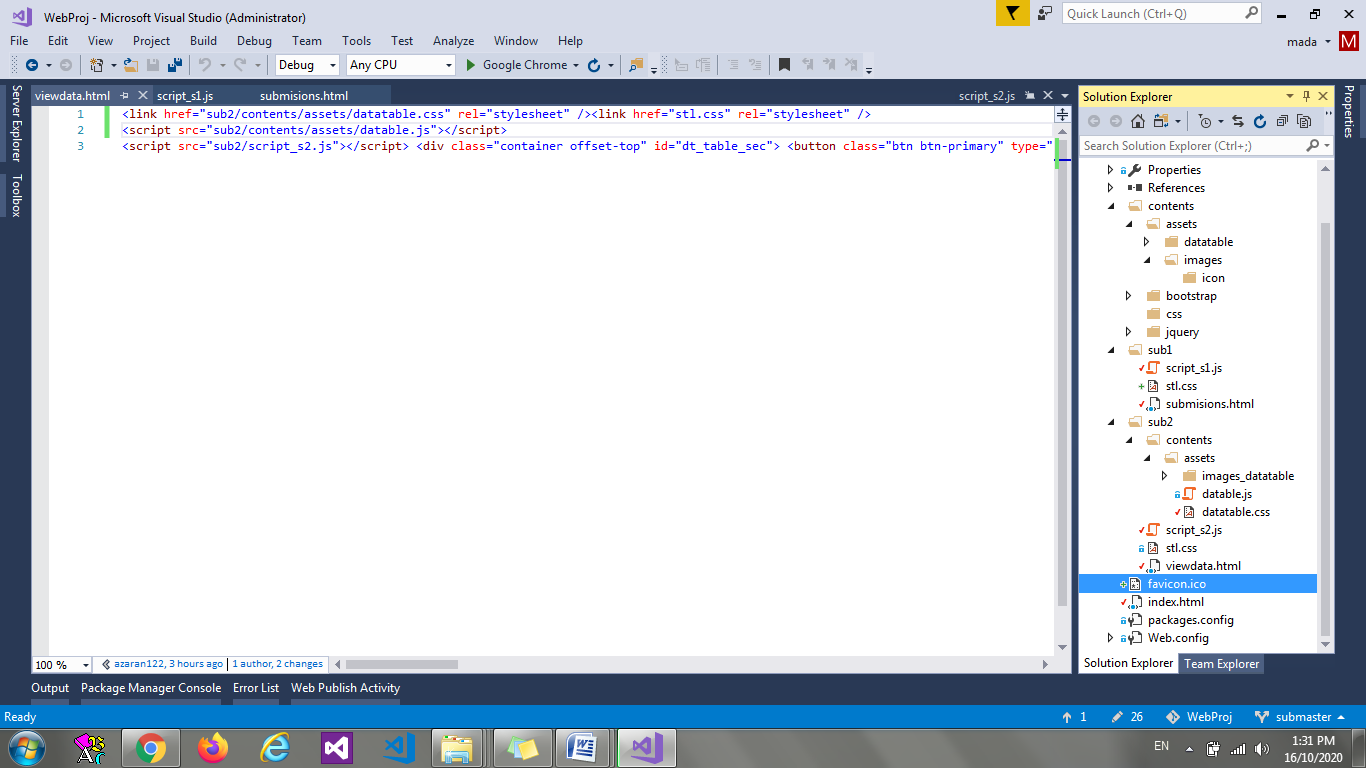
در صورت بروز خطا در دریافت پاسخ: error اجرا می­شود.

**تابع reset:**

جهت خالی نمودن فیلدها پس از ارسال فرم نوشته شده است.

3**-پلاگین نمایش اطلاعات**

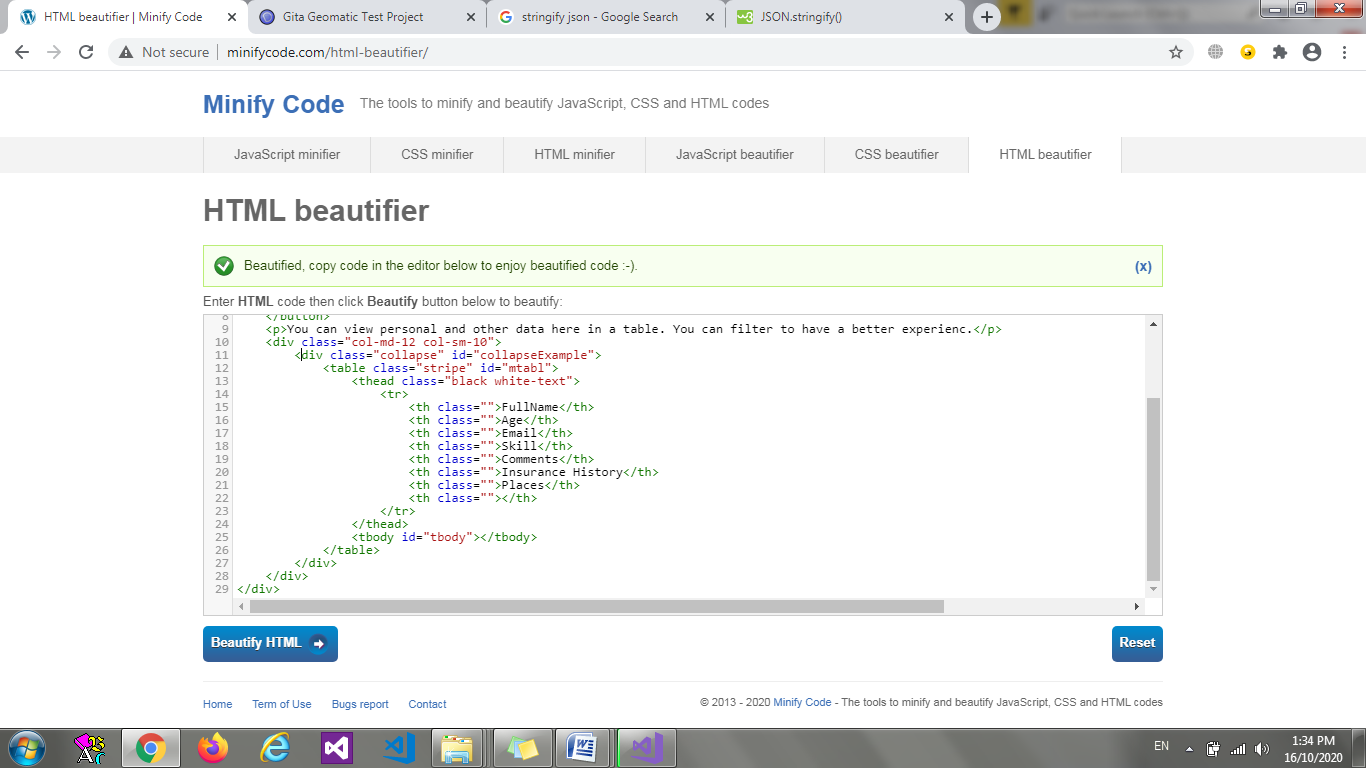
این پلاگین توسط viewdata.hml پیاده سازی شده؛ همچنین فایلهای دیگر: stl.css, script\_s2.js, و در فولدر asset فایلهای مربوط به پلاگین مورد استفاده برای نمایش جدول با امکانات جستجو، pagination، مرتب سازی،نمایش تعداد می­باشد.



در فایل viewdata.html برای نمایش در جدول، تگ table با id=”mtabl” درج شده است.

نام ستونهای جدول در تگ thead قرارگرفته است.

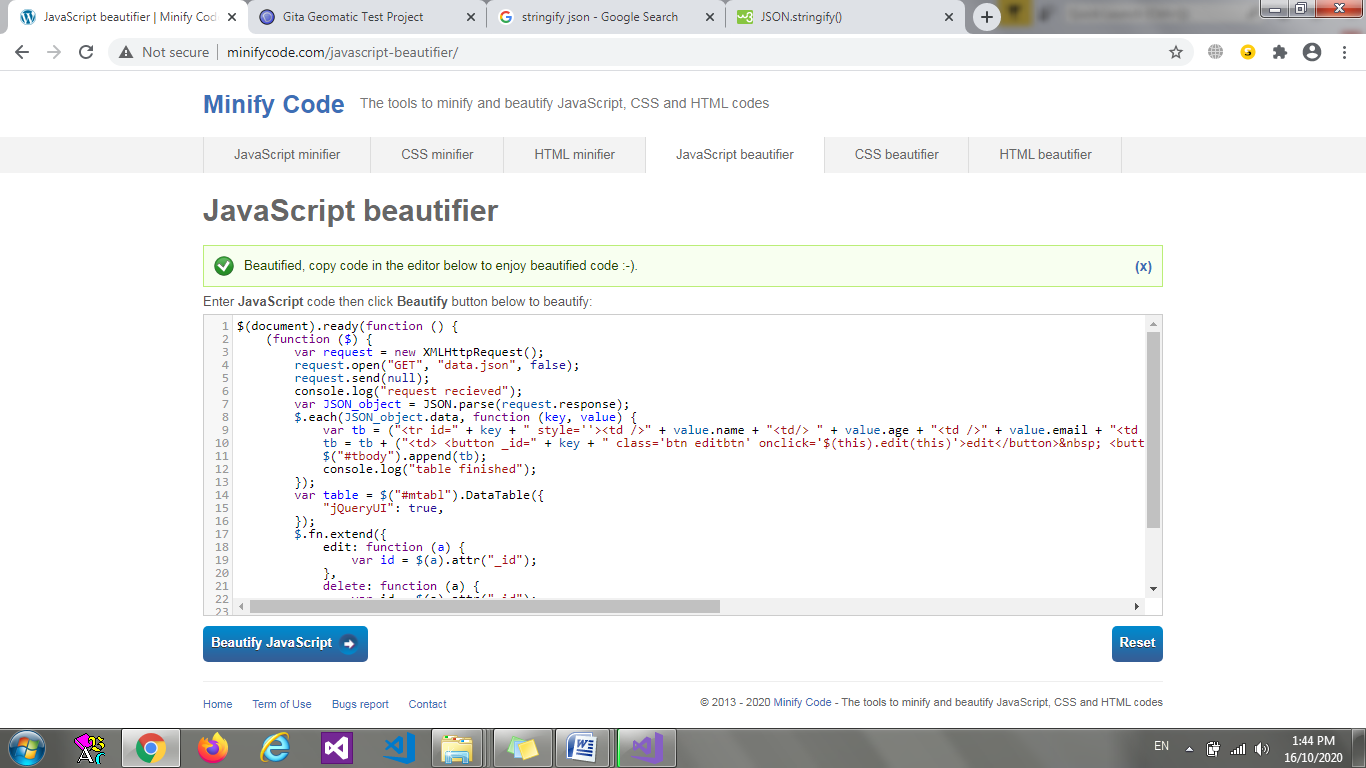
بدنه جدول با id=”"tbody” مشخص شده است.



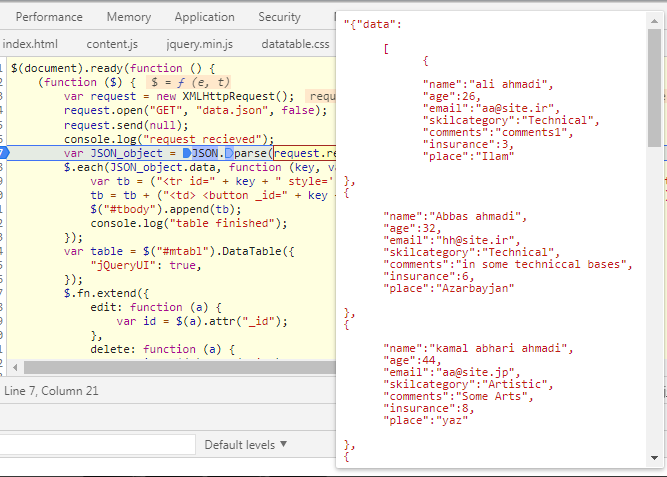
جهت خواندن داده برای درج در جدول یک فایل لوکال به نام data.json درنظر گرفته­ایم. که محتویات آن json می باشد.

**درج داده در جدول در فایل script\_s2.js**

ازآنجا که به محض لود شدن صفحه می خواهیم داده در جدول bind شود، در $(document).ready توسط xmlhttprequest درخواست آبجکت data.json را می دهیم؛ و چون میخواهیم فایل را از سرور بخوانیم، متد ارسالی GET می باشد و null ارسال می شود. پاسخ ارسالی از سرور را جهت تبدیل رشته به object ، JSON.parse را به کار میبریم.



در صورت دریافت آبجکت به درستی، در متغیر JSON\_object پاسخ دریافتی را در نظر میگیریم. در هنگام debug در شکل زیر آبجکت دریافتی، همان data است که request.response آن را در بردارد و به صورت یک آرایه با آن کار می­کنیم :



حال با دستور $.each در این آرایه loop میزنیم. و مقادیر سلول های جدول را در این مرحله پر می کنیم که در متغیر tb قرار می­گیرد. در هر بار پس از تکمیل هر ردیف جدول با دستور $(“#tbody”).append(tb) یک ردیف یا سطر جدول تکمیل می شود. و به جهت آنکه به مقدار قبلی اضافه شود، append استفاده می شود.

جهت افزودن دکمه های ویراش و حذف کد زیر را دقت نمایید :

<button \_id=" + key + " class='btn editbtn' onclick='$(this).edit(this)'>edit</button>&nbsp; <button \_id=" + key + " class='btn delbtn' onclick='$(this).delete(this)'>Delete </button>

مشخصه \_id را به این دکمه ها اضافه کرده ایم، که مقدارش با id هر ردیف از جدول یکسان است.

جهت حذف ردیف از جدول که توضیح آنرا خواهید دید از این id استفاده می شود.

* به منظور افزودن امکانات بیشتر به این جدول از پلاگین datatable.js استفاده نمودیم.

فرمت کلی استفاده از آن:

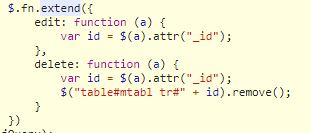
$(tableid).Datatabl() می باشد.

مستند کامل این افزونه را می توانید از آدرس <https://datatables.net/> دریافت نمایید.

**حذف و ویرایش**

جهت فراخوانی این عملیات:

از دستور $(this).edit(this) و $(this).delete(this) استفاده کردیم.



به جهت اینکه id دریافتی در این توابع، مربوط به id دکمه مربوطه می باشد و نیز به جهت آنکه این id برابر با id همان ردیف از جدول می باشد، می توان به راحتی با دستور

$("table#mtabl tr#" + id).remove();

آن ردیف را حذف نمود. mtabl ، id جدول است.

البته توجه شود که فقط در view حذف می شود نه از داده اصلی.

در عمل باید هنگام حذف و یا ویرایش می بایست api مربوطه نیز فراخونی شود.