به نام خدا

تحلیل سمت کلاینت نرم افزار چت با قابلیت استفاده از

GET, PUT, PATCH, DELETE

MQTT،XMPP و WebSocket

و تبادل اطلاعات در سه فرمت

Json, XML وByte

مقدمه

برنامه های چت گروهی از دسته برنامه ها هستند، که این روزها کاربران بیش از هر چیز به سمت آن ها کشیده شده اند، و به دلیل هزینه های کم و تقریبا نا چیز پیام رسان ها شرکت ها، شروع به خریداری این نرم افزار ها کرده اند. در حال حاضر نرم افزار های چت به نرم افزار های جامعی تبدیل شده اند که می توانند مجموع نیاز های انسان ها را پوشش دهند و به همین خاطر گستردگی زیادی که در بین افراد جامعه پیدا کرده، هر دسته از فعالیت ها نیازمند یک سبک از تبادل اطلاعات است، به عنوان مثال چت در حالتی که دو طرف آنلاین باشند می تواند با پروتکل هایی مانند MQTT،XMPP ویا WebSocket ارتباط صورت گیرد ولی در حالت هایی مانند تغییر نام پروفایل کاربر نیاز به تغییرات لحظه ای نیست و می توان سرور را از طریق متدPUT از ویرایش اطلاعات با خبر کرد. در متن زیر تلاش شده علاوه بر هدف گزاری برای طراحی یک سامانه چت تحت وب، تفکیک پروتکل ارتباطی مورد نیاز نیز صورت گیرد.

توابع پایه

ابتدا نیاز به یک کلاس پایه داریم که در آن کلیه توابع مورد نیاز را بگذاریم، توابعی که ارتباط با مرورگر و همچنین ارتباط با سرور را تسهیل کند. همچنین بتواند تبدیل اطلاعات را به خوبی مدیریت کند.
این تابع به صورت یک Object کار میکند و هر کجایی از پروژه می تواند فرآخوانی شود.

نحوه نگارش این Object به این صورت است.

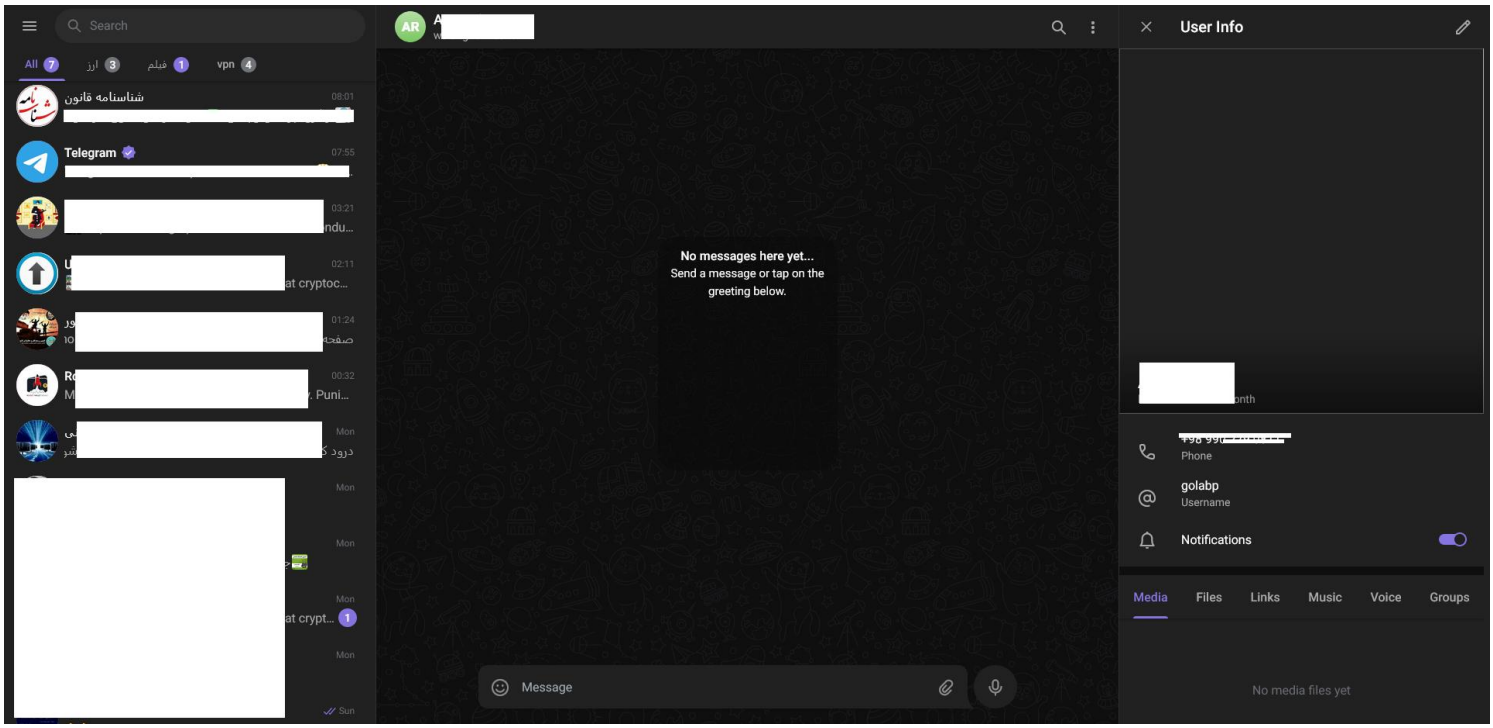
```
Var PublicFunction={
  Notification:{
    Init:function(){
      //initialize
    },
    Show:function(Title,body,onclick){
      //نمایش یک نوتیفیکیشن
    }
  },
  Server:{
    Init:function(){
      //initialize
    },
    WebSocket:{
      Send:function(){},
      Receive:function(){}
    },
    XMPP:{
      Send:function(){},
      Receive:function(){}
    },
    MQTT:{
      Send:function(){},
      Receive:function(){}
    },
    GET:function(URL,Data,Callback){
      // Callback like this
      // function(data,status){
      //   alert("Data: " + data + "\nStatus: " + status);
      // }
      $.get(URL, Callback);
    },
    POST:function(URL,Data,OnSuccess,OnError){
      // var OnSuccess=function (data, status, xhr) {
      //     $('p').append('status: ' + status + ', data: ' + data);
      // },
      // var OnError= function (jqXhr, textStatus, errorMessage) {
      //     $('p').append('Error' + errorMessage);
      // }

      $.ajax(URL, {
        type: 'POST',  // http method
        data: Data,  // data to submit
        success: OnSuccess ,
        error: OnError
      });
    },
    PUT:function(URL,Data,OnSuccess,OnError){
      // var OnSuccess=function (data, status, xhr) {
      //     $('p').append('status: ' + status + ', data: ' + data);
      // },
      // var OnError= function (jqXhr, textStatus, errorMessage) {
      //     $('p').append('Error' + errorMessage);
      // }

      $.ajax(URL, {
        type: 'PUT',  // http method
        data: Data,  // data to submit
        success: OnSuccess ,
        error: OnError
      });
```

```
    },
    PATCH:function(URL,Data,OnSuccess,OnError){
      // var OnSuccess=function (data, status, xhr) {
      //     $('p').append('status: ' + status + ', data: ' + data);
      // },
      // var OnError= function (jqXhr, textStatus, errorMessage) {
      //     $('p').append('Error' + errorMessage);
      // }

      $.ajax(URL, {
        type: 'PATCH',  // http method
        data: Data,  // data to submit
        success: OnSuccess ,
        error: OnError
      });
    },
    DELETE:function(URL,Data,OnSuccess,OnError){
      // var OnSuccess=function (data, status, xhr) {
      //     $('p').append('status: ' + status + ', data: ' + data);
      // },
      // var OnError= function (jqXhr, textStatus, errorMessage) {
      //     $('p').append('Error' + errorMessage);
      // }

      $.ajax(URL, {
        type: 'DELETE',  // http method
        data: Data,  // data to submit
        success: OnSuccess ,
        error: OnError
      });
    },
  },
  Convert:{
    //کلیه توابع تبدیل مانند تبدیل ساعت و.. که در طول پروژه مخص می شود
  },
  Browser:{
    //کلیه توابع مربوط به ارتباط با مرورگر و استفاده از حافظه و کویری استرینگ و..
  }
}
```

درواقع هسته اصلی ارتباطی این سامانه در این قسمت نوشته می شود. ما در این هسته تلاش می کنیم ابتدا تمام توابع ارتباطی خود و تبدیل دیتا ها را یک بار برای همیشه بنویسیم تا در مسیر پروژه برنامه نویسان نیازمند هر نوع ارتباطی باشند با سرور یا با مرورگر زیر ساخت آن فرآهم شده باشد و بدون مشکل این توابع کار دهد.

ساختار برنامه نویسی پروژه چت

پروژه چت از بخش های تشکیل می شود که این بخش ها به شرح زیر است.



1- مرکز چت (مانند تلگرام)

2- منو سمت چپ.

3- منو سمت راست.

هر کدام از این قسمت های زیر سیستم هایی دارد که میتوان در ازاء هرکدارم یک زیر Object ایجاد نمود مثلا اگر ساختار پروژه به صورت زیر باشد.

```
var Chat={
   left:{},
   Center:{},
   Right:{}
}
```

کلیه موارد مربوط به منو همبرگری از منو سمت چپ در این قسمت قرار میگیرد

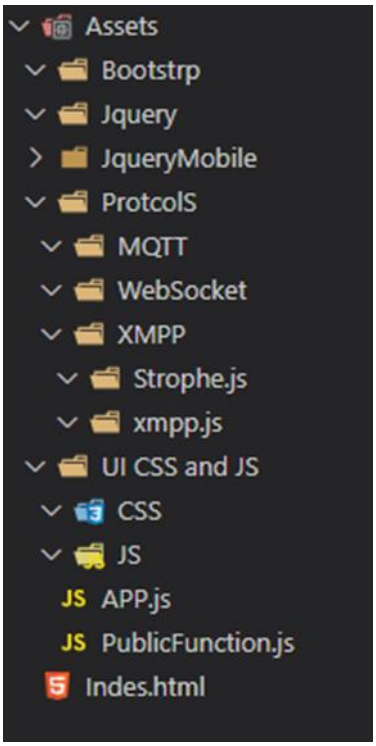
```
var Chat={
   left:{
      search:{
         control:{Input,search}
      },
      HambergerMenu:{
         control:{}
      }
   },
   Center:{},
   Right:{}
}
```

فریم ورک های مورد استفاده

با توجه به حجم پروژه پیشنهاد می شود از فریم ورک هایی زیر برای سرعت در توسعه سمت کلاینت استفاده شود.

1. Jquery
2. در صورت نیاز به ایجاد نسخه موبایل میتواند از jquerymobile استفاده کرد.
3. xmpp.js
   https://github.com/xmppjs/xmpp.js.git
4. bootstrap
5. Strophe.js
   https://strophe.im/strophejs

ساختار پوشه بندی پروژه



کلیه منابع نرم افزاری برای سمت کلاینت در پوشه Assets قرار میگیرد، در صورت نیاز به Bootstrap کلیه فایل های آن در این پوشه ذخیره می شود. از Jquery و JqueryMobile به عنوان فریم ورک برای سرعت بخشیدن به پروژه در کد نویسی استفاده می شود هم چنین کلیه توابع مورد نیاز به Tuch در این فریم ورک ها وجود دارد. در پوشه Protcols فایل های javascript مربوط به پروتکل های ارتباطی گذاشته می شود. همچنین در فایل UI CSS and JS کلیه UI مربوط به شاکله نرم افزار قرار میگیرد. به طور کلی نام پوشه ها گویای محتویات داخل آن است. لازم به توضیح است کلیه فرآیند ها از Index.Html و App.js آغاز می شود. و پیشنهاد می شود برای سرعت عمل بیشتر هم سمت سرور و هم سمت کلایت پیشنهاد می شود از تکنولوزی های websoket و Json استفاده شود. تا علاوه بر سرعت کیفیت و حجم دیتا در حالت Online و Poin To Point بهینه باشد.

رویداد ها و نحوه ارتباط Client با Server

به طور کلی رویداد های در سامانه **Client** چت به دسته های زیر تقسیم می شود.

1- رویداد On Load

در این رویداد ضمن فرآخوانی کلیه فایل های JavaScript باید سامانه به گونه ای باشد که دیتا های راه اندازی را نیز از طریق webservice به صورت Json دریافت نماید. مثلا لیست چت های قبلی.

2- رویداد On Get Message

به طور کلی سامانه های چت یا پیام دریافت می کنند یا پیام ارسال میکنند که On Get Message باید برای دریافت کلیه پیام ها استفاده شود، در این حالت یک پیام می تواند یک Notification باشد یا یک Message از دوستی

3- رویداد On Send Message

به طور کلی سامانه های چت یا پیام دریافت می کنند یا پیام ارسال میکنند که On Send Message باید برای دریافت کلیه پیام ها استفاده شود، در این حالت یک پیام می تواند یک Notification باشد یا یک Message از دوستی

ساختار Social network

ممکن است بخواهیم برای سامانه چت زیر سیستم هایی داشته باشیم مانند بله. در این حالت باید ساختار سامانه از سبک تلگرام تغییر کند و UI بهتری نیازمند است.

پیشنهادات:

می توان موارد زیر را به سامانه اضافه کرد

Social network
Shop