Modul 7

Pemrograman Mobile - Data Access with Ajax

Tujuan Praktikum

- Mahasiswa mampu membuat aplikasi client berbasis web yang memanfaatkan jQuery untuk berkomunikasi dan mengambil konten dari web service dalam format JSON secara *asynchronous* dengan AJAX.
- Mahasiswa mampu menerapkan komunikasi client-server secara asynchronous untuk mengonsumsi hasil olahan data dari aplikasi berbasis web secara asynchronous dengan AJAX.
- Mahasiswa mampu melakukan proses build aplikasi web menjadi aplikasi native

---

7.1 Pendahuluan: Pengenalan AJAX

AJAX adalah singkatan dari Asynchronous JavaScript and XML. AJAX adalah sebuah teknik untuk membuat sebuah aplikasi berbasis web yang lebih interaktif, lebih cepat, dan lebih baik dengan bantuan teknologi XML, HTML, CSS, dan Javascript. Pada awalnya AJAX menggunakan konten dalam format XHTML yang menggabungkan teknologi HTML dengan XML untuk meningkatkan fleksibilitas dan portabilitas dokumen HTML sehingga tidak hanya dapat dikonsumsi oleh web browser, melainkan program aplikasi lain yang dapat berkomunikasi dengan web server melalui internet. Dalam komunikasi AJAX, XHTML umumnya digunakan untuk merepresentasikan konten, CSS untuk mengubah representasi konten secara visual, serta Document Object Model (DOM) dan Javascript untuk mengubah konten yang ditampilkan secara dinamis.

Pada aplikasi berbasis web konvensional, informasi akan dikirimkan pada web server dan diperoleh dari web server melalui permintaan HTTP (HTTP request) yang dilakukan secara *synchronous*. Hal tersebut berarti bahwa ketika pengguna berinteraksi dengan halaman web dan web server untuk mendapatkan konten baru, maka pengguna akan dialihkan ke suatu halaman yang baru dengan informasi yang baru. Dengan menggunakan AJAX, Javascript akan melakukan HTTP *request* ke web server secara background, menginterpretasikan hasil yang diperoleh dari request tersebut, dan memperbarui konten yang ditampilkan pada halaman web dengan konten hasil interpretasi yang diperoleh dari request yang dilakukan secara background tersebut.

Perbedaan yang jelas antara komunikasi client-server konvensional dengan komunikasi secara AJAX terletak pada interaksi dengan pengguna. Pada komunikasi dengan menggunakan AJAX, selama request dilakukan di-*background* pengguna tetap dapat berinteraksi dengan halaman

aplikasi web yang sedang digunakannya tanpa harus menunggu request yang sedang dilakukan di-*background* tersebut selesai. Pengguna tidak akan secara sadar mengetahui bahwa request HTTP sedang dilakukan di background untuk memperbarui konten dan informasi yang sedang ditampilkan.

Dengan demikian interaksi antara aplikasi dengan pengguna dapat dilakukan dengan lebih alami dan intuitif, karena pengguna tidak perlu menekan suatu tombol atau mengetikkan alamat URL untuk memperbarui konten yang ditampilkan. Request HTTP yang dilakukan secara AJAX di *background* dapat dipicu melalui Event yang berasal dari pergerakan mouse, atau dari pergeseran halaman web yang ditampilkan pada layar oleh pengguna.

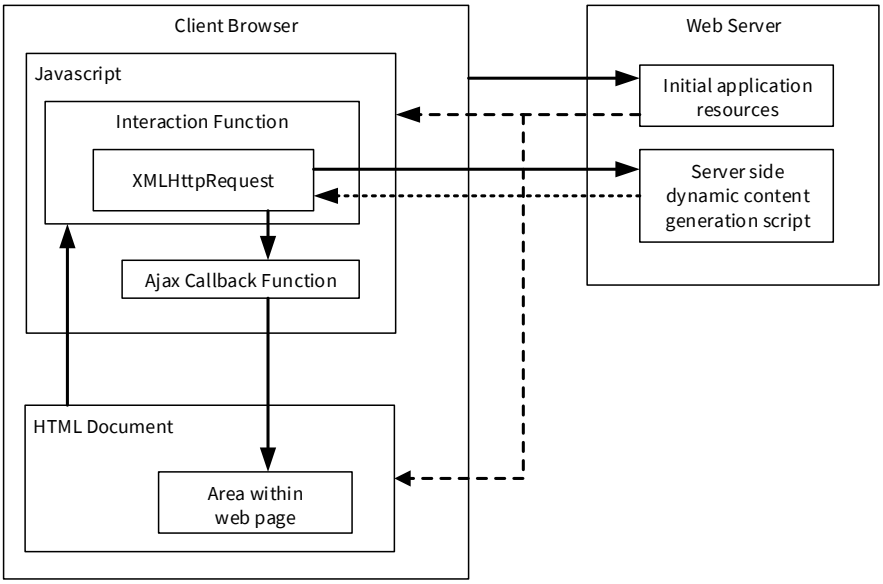
Walaupun pada dasarnya teknologi AJAX menggunakan dokumen XML atau XHTML sebagai standar format dokumennya, saat ini kebanyakan pengembang menggunakan format JSON yang diklaim lebih cepat, lebih ringkas, dan lebih sederhana sebagai format dokumen yang digunakan sebagai format data yang digunakan dalam berkomunikasi antara client dengan server.

Adapun beberapa hal yang perlu diperhatikan adalah tidak semua browser mendukung AJAX. Beberapa browser yang telah mendukung AJAX diantaranya:

- Mozilla Firefox 1.0+,
- Netscape 7.1+,
- Apple Safari 1.2+,
- Google Chrome,
- Microsoft Internet Explorer 5+,
- Konqueror, dan
- Opera 7.6+.

Beberapa browser yang telah disebutkan mendukung komunikasi dengan AJAX, namun tidak semua mendukung dengan metode AJAX yang sama, utamanya pada browser Internet Explorer ketika melakukan instansiasi objek XMLHttpRequest atau biasa disingkat dengan XHR.

Untuk dapat berkomunikasi dengan menggunakan teknik AJAX, diperlukan sebuah web server yang digunakan untuk menyediakan resource dokumen HTML5, CSS, dan Javascript. AJAX hanya dapat bekerja dengan menggunakan protokol web (HTTP/HTTPS). AJAX tidak dapat bekerja dengan menggunakan protocol file:// atau protokol web dimana domain yang menyediakan resource dokumen HTML, Javascript berbeda dengan domain yang menyediakan resource yang diminta dengan menggunakan AJAX. Kondisi komunikasi client-server dimana nama domain resource request AJAX dan dokumen resource request dengan XHR berbeda tersebut dinamakan Cross-Side Scripting (XSS).



Gambar 7.1 Komunikasi *client-server* dengan AJAX

Diagram blok komunikasi client-server dengan AJAX diperlihatkan dalam Gambar 9.1. Komunikasi secara AJAX dimulai dengan request HTTP standar melalui URL pada umumnya untuk mendapatkan resource dokumen HTML dan Javascript. Komunikasi client-server selanjutnya akan memanfaatkan objek XMLHttpRequest (XHR) yang dibangkitkan oleh fungsi interaksi dokumen Javascript yang diperoleh dari proses request HTTP awal. Ketika request HTTP secara AJAX dilakukan oleh objek XHR, maka web server akan memberikan resource yang diminta seperti layaknya request HTTP biasa. Resource yang diberikan oleh web server akan dikembalikan pada objek XHR yang melakukan request AJAX tersebut. Ketika dokumen objek XHR menerima respon dari web server, maka objek XHR akan memanggil fungsi callback dengan memberikan parameter pada fungsi callback tersebut nilai-nilai respon dari eksekusi komunikasi client-server secara AJAX tersebut. Selanjutnya, fungsi callback yang dipanggil bertanggungjawab untuk memperbarui dokumen HTML sesuai dengan respon dan resource yang diberikan oleh objek XHR.

7.2 Mengakses data dari web service dengan protokol REST menggunakan AJAX dan jQuery

jQuery menyediakan beberapa fungsi yang dapat digunakan dalam berkomunikasi secara client-server secara asynchronous dengan menggunakan AJAX. Fungsi utama yang disediakan oleh jQuery untuk berkomunikasi secara AJAX adalah jQuery.ajax(). Method tersebut memiliki bentuk prototype sebagai berikut:

```
jQuery.ajax([settings]);
```

Parameter fungsi tersebut bersifat opsional. Perilaku dari fungsi ajax() akan sesuai dengan jenis tipe data yang diberikan pada parameter settings. Parameter yang dapat diberikan pada fungsi

tersebut dapat dilihat pada halaman dokumentasi fungsi ajax() jQuery di: https://api.jquery.com/jQuery.ajax/

Catatan:

Fungsi ajax() pada jQuery hanya akan berjalan dengan baik jika dokumen HTML yang akan berkomunikasi secara asynchronous dan alamat URL yang diberikan pada parameternya berasal dari sebuah web server dengan menggunakan protokol http:// atau https://. Protokol file:// (seperti yang didapat ketika membuka file html secara langsung tanpa disediakan melalui komunikasi dengan web server) tidak dapat digunakan untuk berkomunikasi secara asynchronous.

Berikut ini adalah contoh penggunaan method ajax() untuk berkomunikasi secara asynchronous dengan menggunakan method POST:

```
$.ajax({
  method: "POST",
  url: "some.php",
  data: { name: "John", location: "Boston" }
}).done(function( msg ) {
    alert( "Data Saved: " + msg );
  });
```

Pada contoh tersebut, request AJAX akan dilakukan dengan menggunakan method HTTP POST pada alamat URL "some.php" yang terletak di dalam direktori yang sama dengan dokumen HTML yang menginisiasi komunikasi secara asynchronous tersebut. Ketika request AJAX dilakukan, sebuah objek yang didefinisikan dengan JSON pada parameter "data" akan dikirimkan ke file "some.php". Setelah request HTTP selesai dilakukan, informasi/output yang dihasilkan oleh file "some.php" akan disimpan dalam variable "msg" untuk diproses lebih lanjut di sisi client.

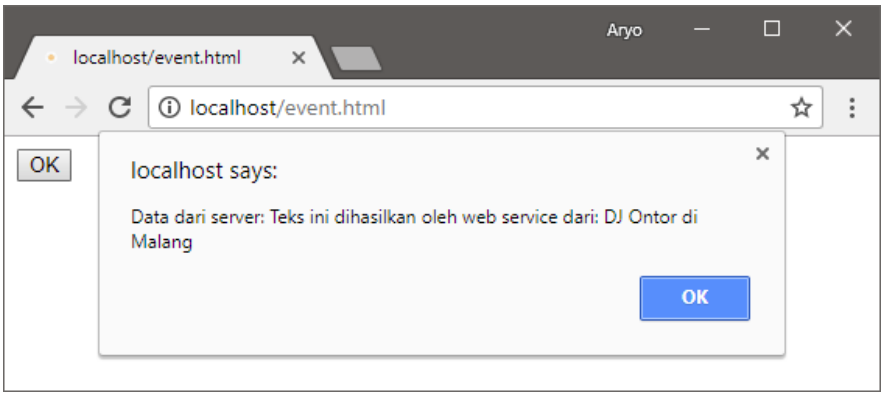
Sebagai contoh, file "some.php" berisi kode program berikut:

```
<?php
    echo "Teks ini dihasilkan oleh web service dari: ";
    echo $_POST['name'] . " di " . $_POST['location'];
?>
```

File "some.php" diakses secara asynchronous dengan menggunakan kode javascript berikut:

```
$.ajax({
  method: "POST",
  url: "some.php",
  data: { name: "DJ Ontor", location: "Malang" }
}).done(function( msg ) {
    alert( "Data dari server: " + msg );
  });
```

Akan menghasilkan tampilan seperti pada gambar berikut ini:

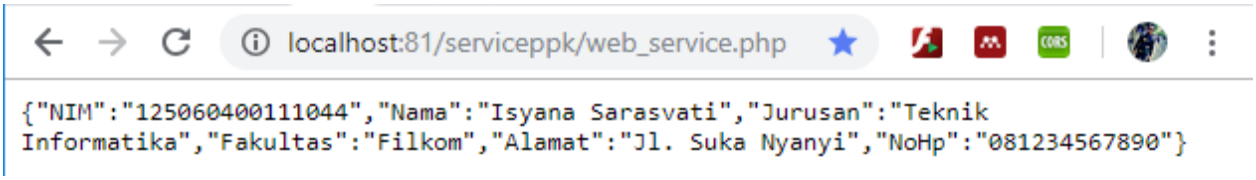


Jika method HTTP yang digunakan untuk berkomunikasi secara AJAX adalah POST, maka di sisi server (PHP) nilai variabelnya dapat diambil dengan menggunakan variabel superglobal $_POST[]. Jika method HTTP yang digunakan adalah GET, maka PHP dapat mengambil variabel tersebut dengan variabel superglobal $_GET. Jika method HTTP yang digunakan tidak dapat dipastikan apakah GET/POST, maka PHP dapat mengambil variabel dengan menggunakan variabel superglobal $_REQUEST[].

7.3 Contoh Kasus Mengakses data dalam format JSON dari web service dengan protokol REST menggunakan AJAX dan jQuery oleh aplikasi Front-end

Berikut ini adalah contoh penggunaan method ajax() untuk berkomunikasi secara asynchronous dengan menggunakan method GET:

Sebuah web service memiliki hasil tampilan berikut (hasil dari modul sebelumnya):



Untuk mendapatkan objek data mahasiswa yang diberikan oleh web service menggunakan javascript secara AJAX, maka lakukan perubahan kode di aplikasi front-end (hasil dari modul sebelumnya) pada file index.html menjadi:

```
<body>
    <div data-role="page" id="page-one">
        <div data-role="header">
            <h1>Daftar Mahasiswa</h1>
        </div>
        <div role="main" class="ui-content">
            <ul id="list-mhs" data-role="listview" data-inset="true" data-filter="true">
            </ul>
        </div>
    </div>

    <div data-role="page" id="page-two">
        <div data-role="header">
            <h1>Detail Mahasiswa</h1>
        </div>
        <div role="main" class="ui-content" id="main2">
            <p id="p-nim"></p><br>
            <p id="p-nama"></p><br>
            <p id="p-jurusan"></p><br>
            <p id="p-fakultas"></p><br>
            <p id="p-alamat"></p><br>
            <p id="p-nohp"></p><br>
        </div>
    </div>
    <script src="js/app.js" type="text/javascript"></script>
    <script>
        Application.initApplication();
    </script>
</body>
```

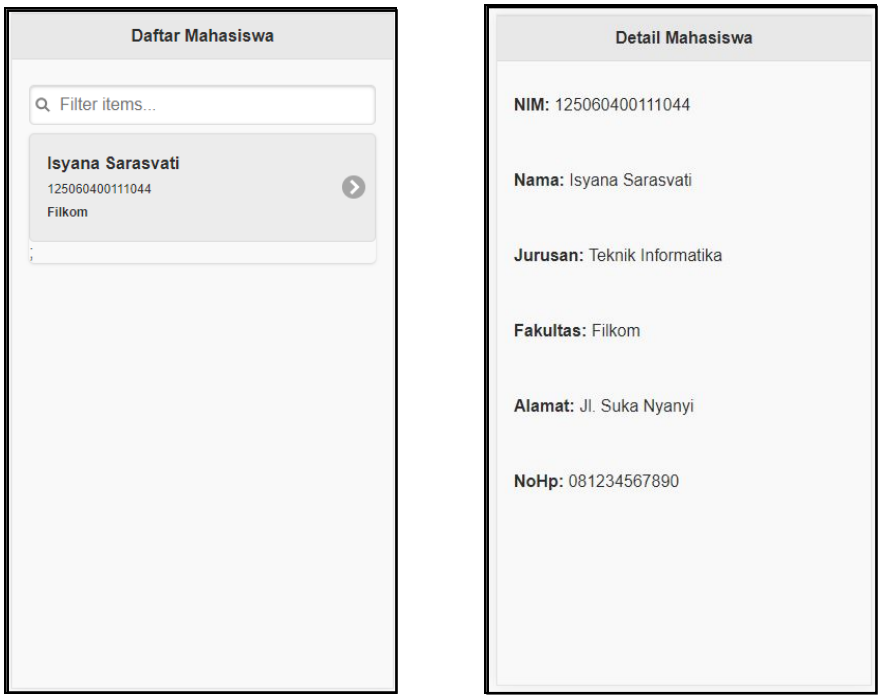
Kemudian lakukan perubahan pada file app.js menjadi:

```
    initShowMhs : function() {
        $.ajax({
            url : 'http://localhost:81/serviceppk/web_service.php',
            type : 'get',
            beforeSend : function() {
                $.mobile.loading('show', {
                    text : 'Please wait while retrieving data...',
                    textVisible : true
                });
            },
            success : function(dataObject) {
                var appendList = '<li><a href="#page-two?id='+
                dataObject.NIM+'" target="_self" id="detail-mhs" data-nimmhs="'+
                dataObject.NIM+'"><h2>'+dataObject.Nama+'</h2><p>'+dataObject.NIM+
                '</p><p><b>'+dataObject.Fakultas+'</b></p></a></li>';
                $('#list-mhs').append(appendList);
                $('#list-mhs').listview('refresh');
            },
            complete : function() {
                $.mobile.loading('hide');
            }
        });
    },
```

```
36      initShowDetailMhs : function(nim) {
37          $.ajax({
38              url : 'http://localhost:81/serviceppk/web_service.php',
39              type : 'get',
40              beforeSend : function() {
41                  $.mobile.loading('show', {
42                      text : 'Please wait while retrieving data...',
43                      textVisible : true
44                  });
45              },
46              success : function(dataObject) {
47                  $('#p-nim,#p-nama,#p-jurusan,#p-fakultas,#p-alamat,#p-nohp').empty();
48                  $('#p-nim').append('<b>NIM: </b>'+dataObject.NIM);
49                  $('#p-nama').append('<b>Nama: </b>'+dataObject.Nama);
50                  $('#p-jurusan').append('<b>Jurusan: </b>'+dataObject.Jurusan);
51                  $('#p-fakultas').append('<b>Fakultas: </b>'+dataObject.Fakultas);
52                  $('#p-alamat').append('<b>Alamat: </b>'+dataObject.Alamat);
53                  $('#p-nohp').append('<b>NoHp: </b>'+dataObject.NoHp);
54              },
55              complete : function() {
56                  $.mobile.loading('hide');
57              }
58          });
59      }
```

No Baris Kode	Penjelasan
12-34	Fungsi yang dipanggil untuk menambah dan menampilkan kode HTML berupa list item beserta isi data yang terdapat pada data JSON kepada list view daftar mahasiswa (halaman pertama aplikasi) di file index.html
13-33	Kode untuk request AJAX.
14	Alamat URL request AJAX atau alamat URL backend API.
15	Method HTTP pada AJAX yang digunakan untuk mendapatkan data dari server yaitu method GET
16-21	Fungsi yang dipanggil untuk melakukan perubahan pada sebuah objek, dalam hal ini adalah menampilkan pesan loading, ketika sebelum melakukan request AJAX.
22-29	Fungsi yang dipanggil ketika request AJAX telah sukses. Pada fungsi ini, data JSON telah didapatkan dan kemudian dilakukan konsumsi JSON (dalam hal ini nilai dari objek JSON dimasukkan pada sebuah variable) yang selanjutnya dilakukan append ke elemen HTML (dalam hal ini ID “list-mhs”).
30-32	Fungsi yang dipanggil ketika request telah selesai (setelah melakukan fungsi success atau error). Pada kode ini, jika request telah selesai, pesan loading yang ditampilkan oleh fungsi beforesend akan disembunyikan.

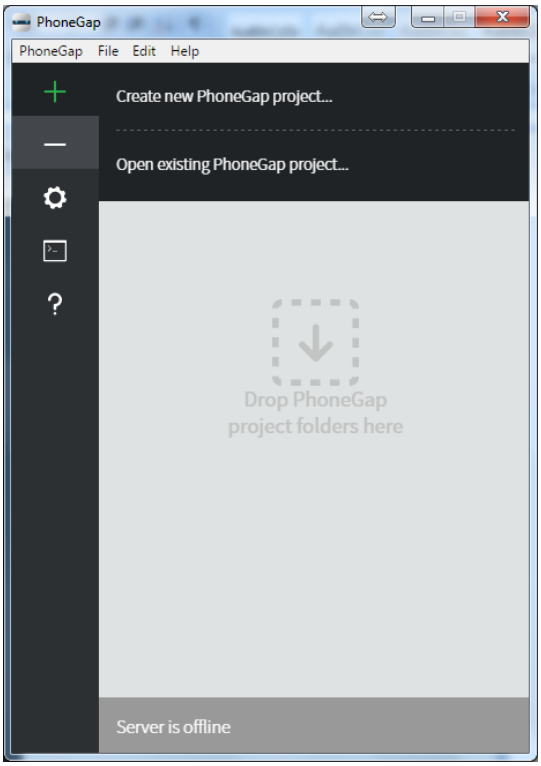
Jika kode program javascript tersebut dijalankan akan menghasilkan tampilan seperti berikut ini:



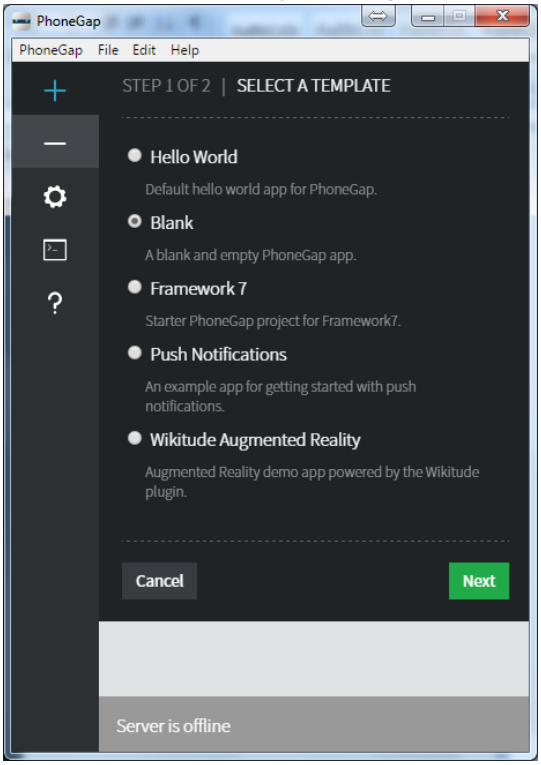
7.4 Build Aplikasi menggunakan Phonegap

PhoneGap adalah sebuah kerangka kerja/framework open source yang dipakai untuk membuat aplikasi cross-platform mobile dengan HTML, CSS, dan JavaScript. Artinya, hanya dengan 1 code, pengembang dapat membuat aplikasi untuk smartphone iPhone, Android, Blackberry, Symbian dan Palm. Untuk build aplikasi Mobile Hybrid dibutuhkan Phonegap desktop dan online. Pengisntall-an Phonegap desktop dapat di di baca pada link http://docs.phonegap.com/getting-started/1-install-phonegap/desktop. Setelah instal Phonegap desktop, selanjutnya adalah membuat aplikasi Mobile Hybrid dengan langkah-langkah sebagai berikut:

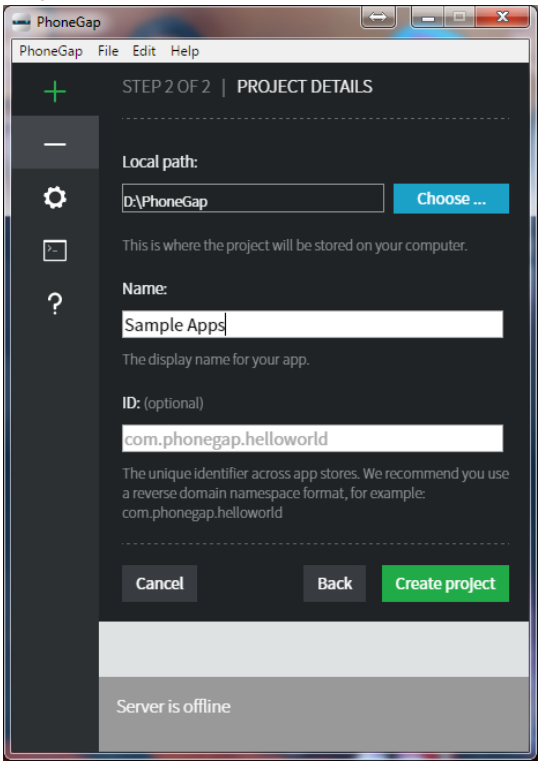
1. Klik tanda plus (+) dan pilih **Create new PhoneGap project**



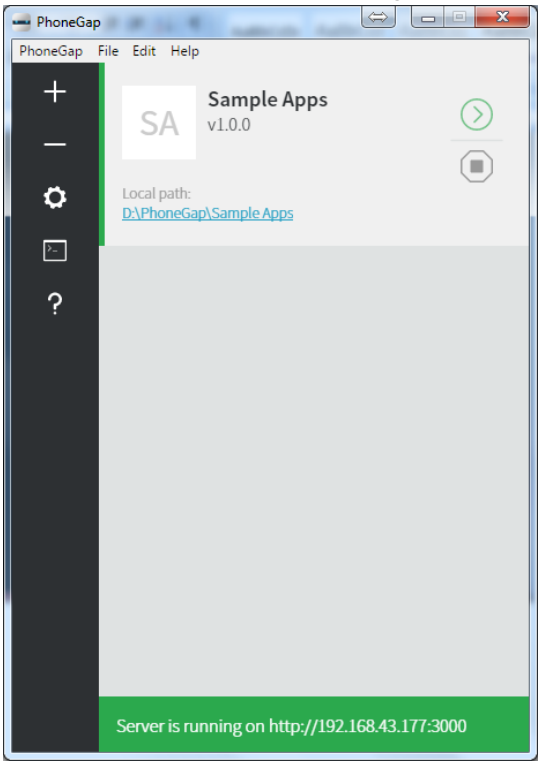
2. Pilih **Blank Template** untuk membuat project dengan template kosong



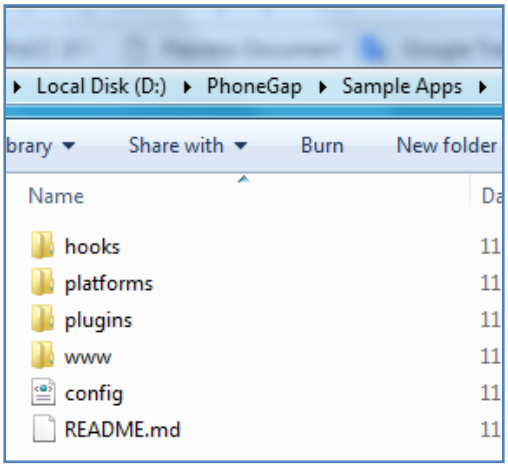
3. Isi **Detail Project** dimana terdapat isian **Local Path** yaitu dimana nantinya project akan di tempatkan dalam drive. Setelah diisi **Detail Project**, kemudian klik tombol **Create Project**



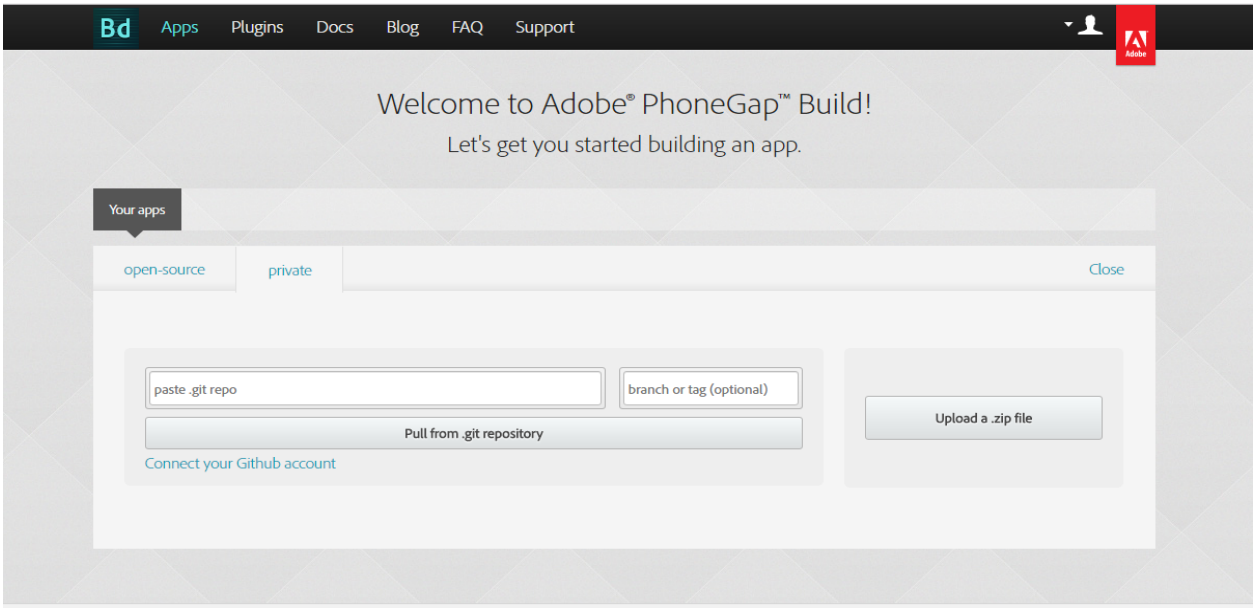
4. Setelah Berhasil, maka akan tampil tampilan sebagai berikut



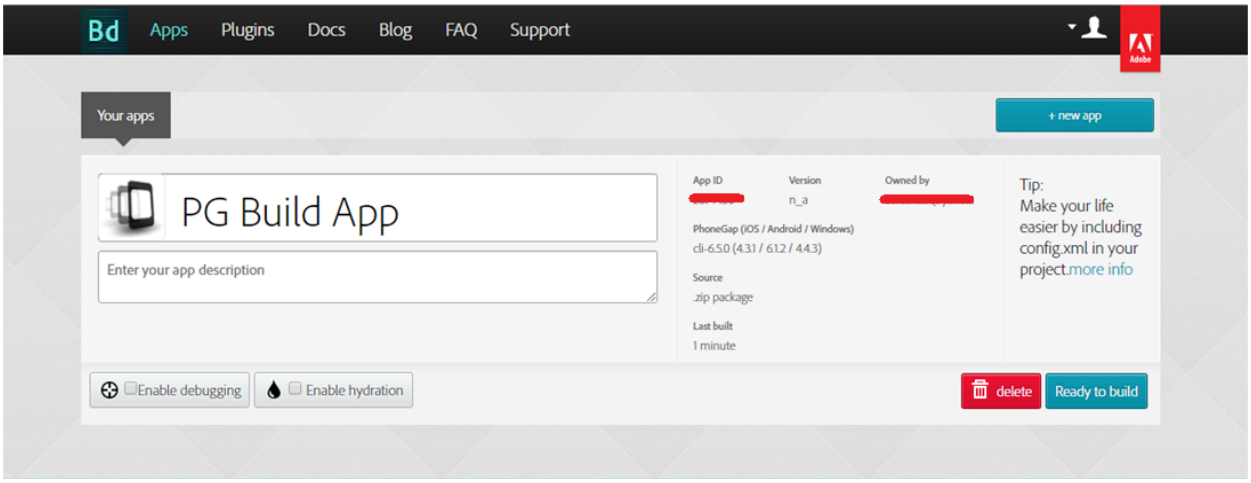
5. Buka path yang telah dimasukkan pada langkah ke 3. Folder www merupakan folder yang berisi HTML, CSS dan JS code aplikasi yang telah/akan dibuat.



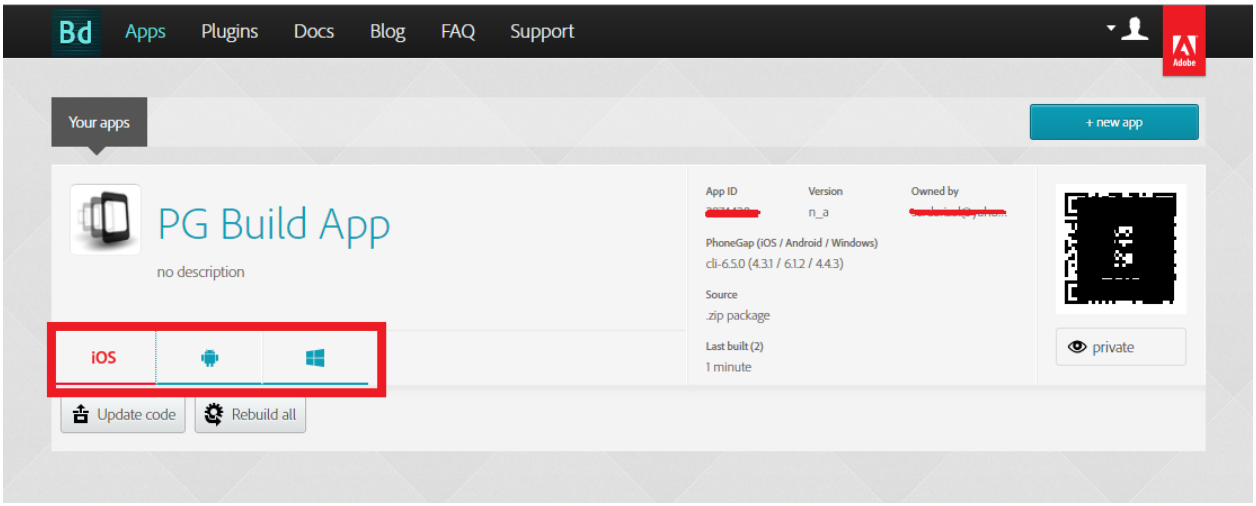
6. Setelah code dirasa code telah benar dan siap, Folder Project yang telah dibuat kemudian di extract dengan ekstensi .zip dan siap untuk di Build secara online di http://build.phonegap.com dengan login/signin terlebih dahulu



7. Unggah File .zip project anda dan beri deskripsi tentang aplikasi jika diperlukan dan kemudian klik tombol **Ready to Build**



8. Download aplikasi sesuai dengan platform yang dibutuhkan



Instruksi Tugas 9

1. Ubah kode pada aplikasi frontend (hasil dari tugas Bab 7) sehingga dapat mengkonsumsi data JSON hasil dari tugas Bab 6!
2. Build kode tersebut menggunakan phonegap menjadi aplikasi android (.apk)!
3. Install pada smartphone anda!
4. Screenshot hasil dari program anda!
5. Jelaskan kode yang telah anda ubah!