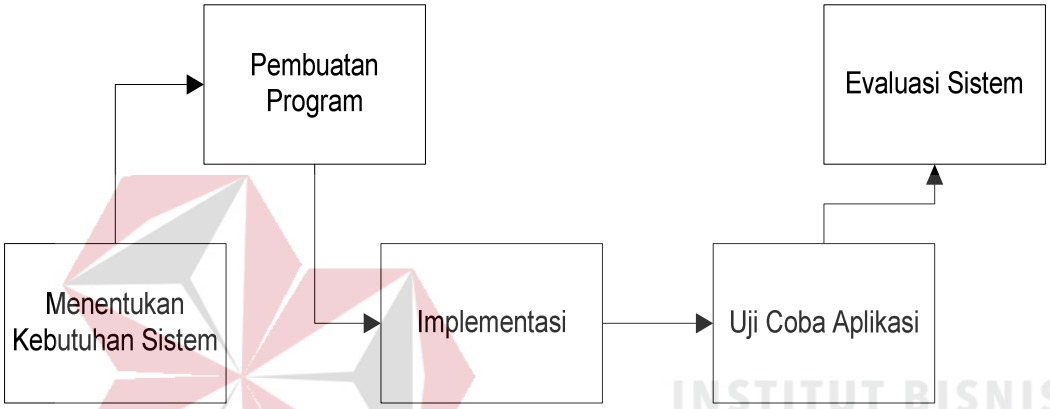
BAB IV

IMPLEMENTASI DAN EVALUASI

Pada bab ini akan menjelaskan tentang implementasi dan evaluasi dalam pengembangan aplikasi yang dibangun berbasis *mobile*. Adapun langkah-langkah pengerjaan yang dilakukan pada bab ini seperti pada Gambar 4.1.



Gambar 4.1. Blok Diagram Implementasi dan Evaluasi

4.1 Kebutuhan sistem

Sistem ini membutuhkan media yaitu sebuah *smartphone*. Media *smartphone* nantinya untuk latihan *user* dalam menulis huruf hijaiyah. Sistem nantinya akan mendeteksi tulisan *user* yang telah dituliskan pada layar *smartphone*. Hasil titik *point* tulisan *user* dikirim dan dicocokkan oleh sistem sehingga mendapatkan hasil salah dan benar.

Untuk dapat menjalankan sistem ini, maka diperlukan perangkat keras untuk *user* dan perangkat lunak untuk pengembangan dengan kondisi dan spesifikasi tertentu agar Aplikasi Menulis Huruf Arab dapat berjalan dengan baik.

4.1.1 Kebutuhan Perangkat Keras

Aplikasi menulis huruf arab berbasis Android dijalankan pada perangkat *mobile* Android. Spesifikasi *handphone* yang dibutuhkan untuk menjalankan aplikasi ini adalah:

1. *Best view* layar 4 inc .
2. Minimum CPU *Single-core* 1 GHz *processor*.
3. Minimum *Internal memory* 2 GB, tersedia 8 MB untuk aplikasi ini.
4. Minimum RAM 512

4.1.2 Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak yang digunakan untuk mengembangkan Aplikasi *Mobile* menulis huruf arab adalah:

A. Kebutuhan Perangkat Lunak *User*

Perangkat lunak yang dibutuhkan untuk menjalankan aplikasi huruf arab pada *smartphone* sebenarnya tidak ada, karena sistem hanya membutuhkan *operating system* Android minimal Android 2.3. Semua yang dibutuhkan agar sistem ini berjalan sudah menjadi satu dalam *operating system* Android tersebut.

B. Kebutuhan Perangkat Lunak Pengembangan

Perangkat lunak untuk pengembangan dibutuhkan dalam *develop* dan *testing* aplikasi huruf arab. berikut spesifikasi perangkat lunak:

1. Android SDK r10
2. ADT Plugin *for* Eclipse
3. Eclipse Indigo
4. Dreamweaver

5. Mozila Firefox
6. Phonegapp (*Native Android*)
7. *Bluestack* (Simulator Android) atau *handphone*.
8. Adobe Photoshop CS 6

4.2 Pembuatan Program

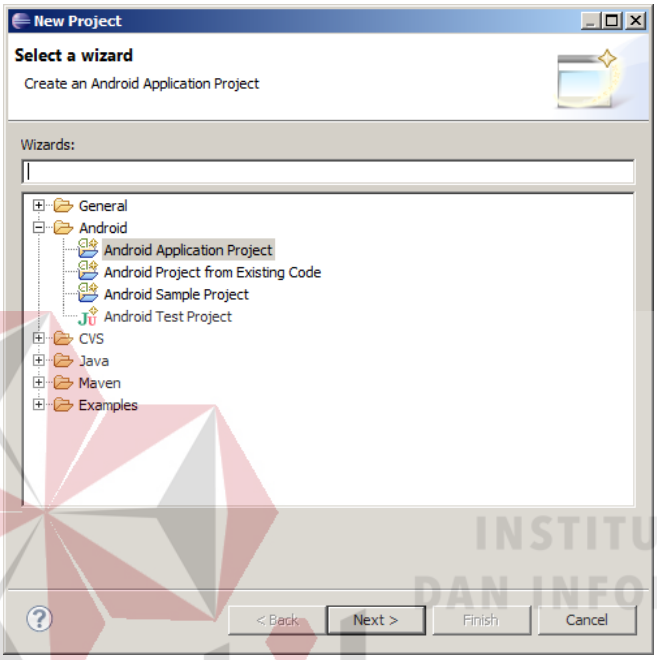
Aplikasi menulis huruf arab dibangun dengan menggunakan Android SDK r-10 dan *text editor* Eclipse Indigo yang didukung dengan ADT plugin for Eclipse dan juga Dreamweaver. Penulisan kode program pada *text editor* disimpan dalam file dengan extensi *.xml, *.js dan *.html. Android SDK akan meng-*compile file* berekstensi *.js dan *.html, kemudian *phonegapp*/ *cordova* meng-*compile* file berekstensi *.js dan *.html tersebut, akan membuat *package* untuk aplikasi *mobile* menulis huruf arab berekstensi *.apk. Nantinya *file* *.apk tersebut yang akan digunakan untuk menjalankan aplikasi melalui *handphone* ber-*operating system* Android.

4.3 Implementasi Sistem

Hal yang dilakukan setelah kebutuhan sistem terpenuhi adalah mengimplementasikan rancangan sistem ke dalam sebuah aplikasi menulis huruf arab. Aplikasi ini terdapat beberapa modul, yaitu modul untuk database, modul gestures, modul metode dalam pecocokan yaitu ndollar, modul nilai berdasarkan *input gestures* dengan *gestures template* dan modul titik koordinat huruf arab yang sudah ditentukan yang digunakan dalam pencocokan *input gestures* oleh *user*.

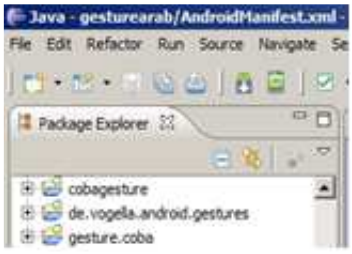
4.3.1 Membuat Project Android

Pembuatan aplikasi menulis huruf arab berbasis Android dibuat pada *software* eclipse. Cara membuat project pada eclipse adalah pilih File – New – Project, kemudian pilih "Android Application Project" seperti pada Gambar.



Gambar 4.2 Pembuatan Project Android

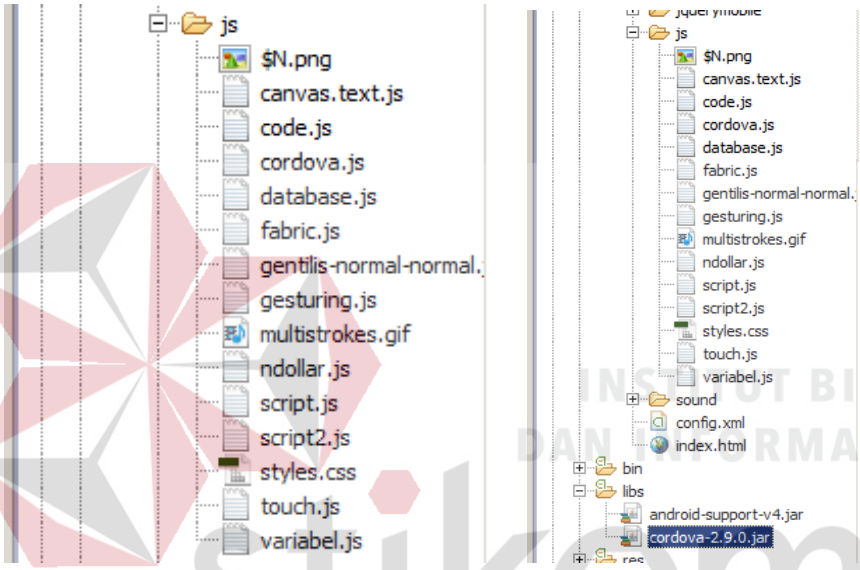
Setelah itu isi nama project beserta minimal requirement .sdk hal ini untuk menentukan minimal aplikasi dijalankan pada *operating system* Android versi berapa. Kemudian menentukan bentuk *icon*, nama *activity* dan *activity main*. Disini saya membuat *project* dengan nama *gesturesarab*. Berikut hasil pembuatan *project* Android.



Gambar 4.3 Hasil Pembuatan Project Android baru

4.3.2 Pembuatan Aplikasi Menggunakan Phonegap

Aplikasi menulis huruf arab menggunakan bahasa pemograman .html. Agar aplikasi ini bisa berjalan pada *mobile Android* maka dibutuhkan *phonegap* dengan cara menyertakan sebuah *library* JavaScript dari Phonegap. *Library* ini diberi nama *cordova.js*. Berikut pada Gambar 4.4 dari *library* Phonegap yang terdapat pada aplikasi.

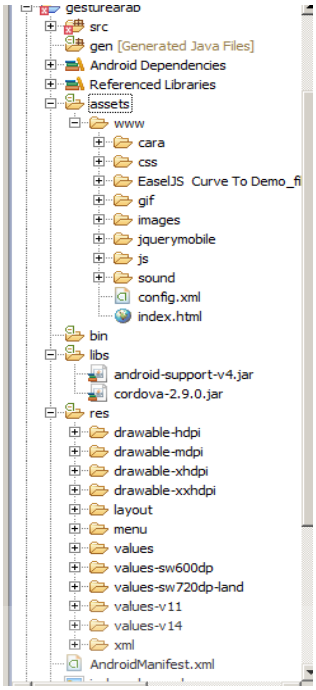


Gambar 4.4 *Library* Phonegap Pada Aplikasi Menulis Huruf Arab

Cordova.js dan *cordova.js* terdapat sebuah API tertentu yang dapat digunakan dalam pembuatan aplikasi Android. *Library cordova.js* inilah yang menjembatani antara bahasa pemograman Phonegap dan fitur asli dalam aplikasi *mobile* seperti *gestures*, gambar, suara, dan *pop-up*.

4.3.3 Struktur Bagian Android

Pada *project* Android yang dibuat terdapat beberapa strukur atau bagian yang masing-masing mempunyai fungsi tersendiri. Berikut pada Gambar 4.5 adalah gambarannya.

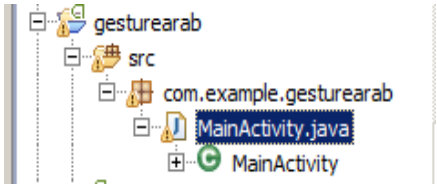


Gambar 4.5 Struktur Android pada Aplikasi Menulis Huruf Arab

Pada *project* aplikasi menulis huruf arab terdapat beberapa bagian struktur, yaitu :

a. *Src*

Pada *project* Android terdapat src, pada bagian ini berfungsi sebagai *activity* awal dari sebuah aplikasi Android atau disebut juga *mainactivity*. Seperti terlihat pada Gambar 4.6.

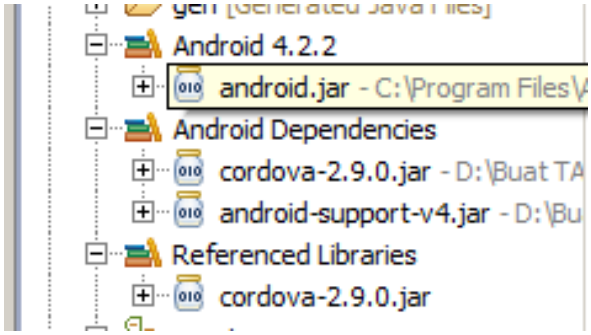


Gambar 4.6 MainActivity Android

b. *Android 4.2.2, Android Dependences* dan *Referenced Libraries*

Pada *project* Android terdapat *Android 4.2.2, Android Dependences* dan *Referenced Libraries*, pada bagian ini berfungsi sebagai *library* Android. Jika

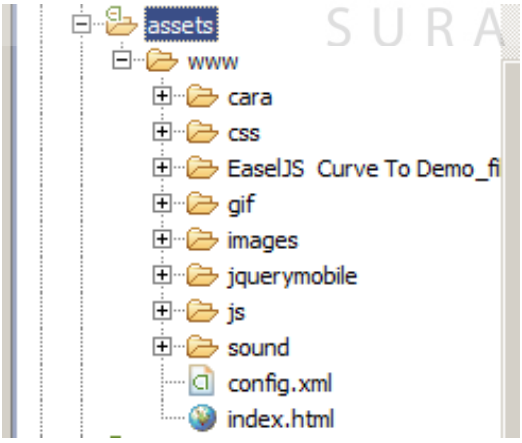
aplikasi menggunakan fitur *library* dari Android maka akan tampak pada bagian *Android 4.2.2*, *Android dependences* dan *referenced libraries* Seperti terlihat pada Gambar 4.7.



Gambar 4.7 *Library* yang Ada Pada Aplikasi Menulis Huruf Arab

c. *Asset*

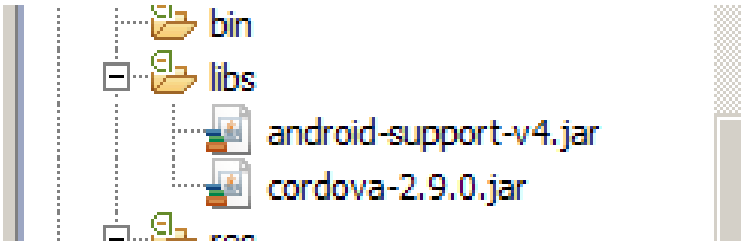
Pada aplikasi menulis huruf arab terdapat bagian Asset. Bagian ini berisi tentang modul, fungsi yang ada pada aplikasi menulis huruf arab, *index.html* (halaman *index* pada aplikasi menulis huruf arab) dan gambar-gambar yang diperlukan dalam aplikasi menulis huruf arab. Untuk lebih jelas isi struktur android bagian asset dapat dilihat pada Gambar 4.8.



Gambar 4.8 Isi Struktur Android pada Bagian Asset

d. *Libs*

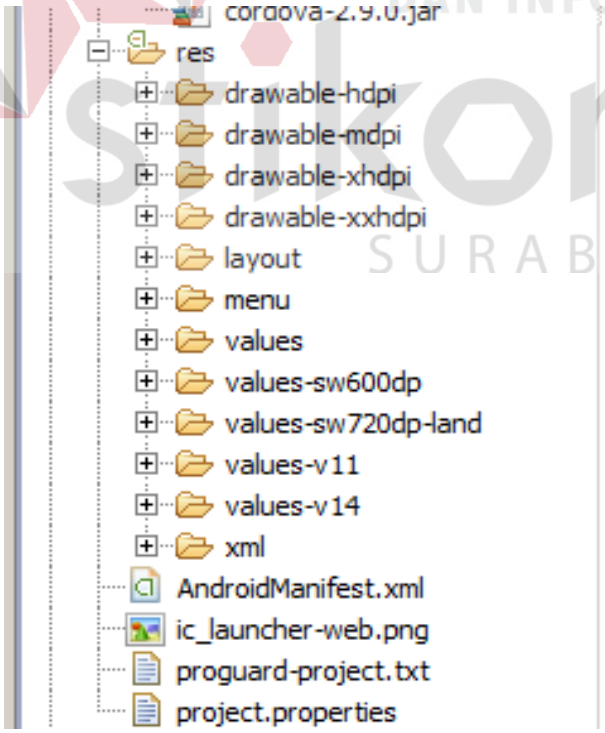
Pada bagian ini terdapat *library* Phonegap yaitu *cordova.js*. Semua fitur Phonegap yang dibutuhkan ada pada *cordova.js*. Untuk lebih jelas tampilan isi struktur pada bagian libs dapat dilihat pada Gambar 4.9.



Gambar 4.9 Isi Struktur pada Bagian libs

e. *Res*

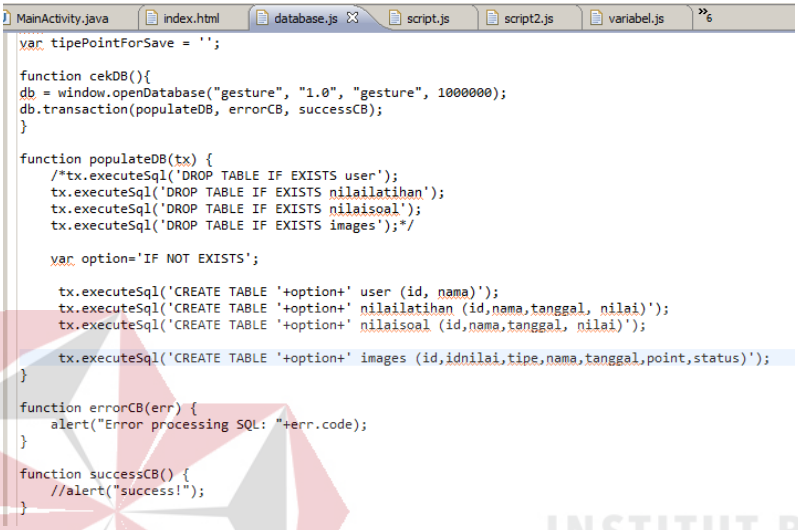
Pada bagian *res*, terdapat tampilan *layout* desain dari Android. Bagian ini berguna dalam mengatur gambar dan tata letak dalam desain. Untuk lebih jelas fitur *layout* pada android dapat dilihat pada Gambar 4.10.



Gambar 4.10 Fitur *layout* pada Android

4.3.4 Halaman Awal Aplikasi

Pada saat aplikasi pertama kali dijalankan, sistem akan mengecek terdapat apakah *database* sudah dibuat atau belum. Bila belum terdapat *database* maka sistem akan membuat sebuah *database* seperti pada Gambar 4.11.



Gambar 4.11 Cek *Database* dan Buat Tabel pada Aplikasi

Ketika aplikasi dijalankan maka fungsi cekdb() memproses, bila belum terdapat database "*If not Exist*" maka akan di eksekusi melakukan pembuatan *database* "*create table*".

Untuk masuk ke menu utama aplikasi, *user* harus mempunyai *user login.* Tanpa *user* login tidak bisa masuk ke halaman menu utama aplikasi. Masuk menu utama hanya ada pada fungsi tambah *user* dan *list user*, karena pada fungsi tambah *user* dan *list user* akan ada *link* ke halaman menu utama aplikasi. Berikut seperti terlihat pada Gambar 4.12.

Pada Gambar 4.12 terdapat funsi tambahUser terdapat *executed* "*insert table*", setelah itu ada perintah windows.location.href= #menu, maksudnya adalah setelah melakukan proses "*insert table*", maka kemudian akan diproses masuk ke

menu halaman utama “#menu”. Fungsi querySuccess adalah pengecekan pada *list user*. Bila *user* dapat memilih salah satu *user login* dari menu lanjut, maka langsung menuju ke halaman menu utama aplikasi “a href=”#menu”, kemudian ketika klik mengambil data *iduser*.

```
function tambahUser(tx){
    var u=$('#userbaru').val();
    var id = jmlID;
    tx.executeSql('INSERT INTO user (id, nama) VALUES ('+id+', "'+u+'")');
    localStorage.user=u;
    localStorage.iduser=id;
    window.location.href='#menu';
}

/* -------------- */

/* Ambil data user */
function queryDB(tx) {
    tx.executeSql('SELECT * FROM user', [], querySuccess, errorCB);
}

function querySuccess(tx, results) {
var len = results.rows.length;
    console.log("DEMO table: " + len + " rows found.");
    var xxx='';
    for (var i=0; i<len; i++){
        //xxx+="<br/>" + (i+1) + ". <a href=\"#menu\" onclick=\"goLogin('"+results.rows.item(i).data+"')\" sty
        xxx+='<li><a href="#menu" onclick="localStorage.iduser=\''+results.rows.item(i).id+'\';localStorage.us
    }
    $('#listuser').html(xxx);
    $('#listuser').listview('refresh');
}
/* -------------- */
```

Gambar 4.12 Fungsi Masuk ke Halaman Menu Utama

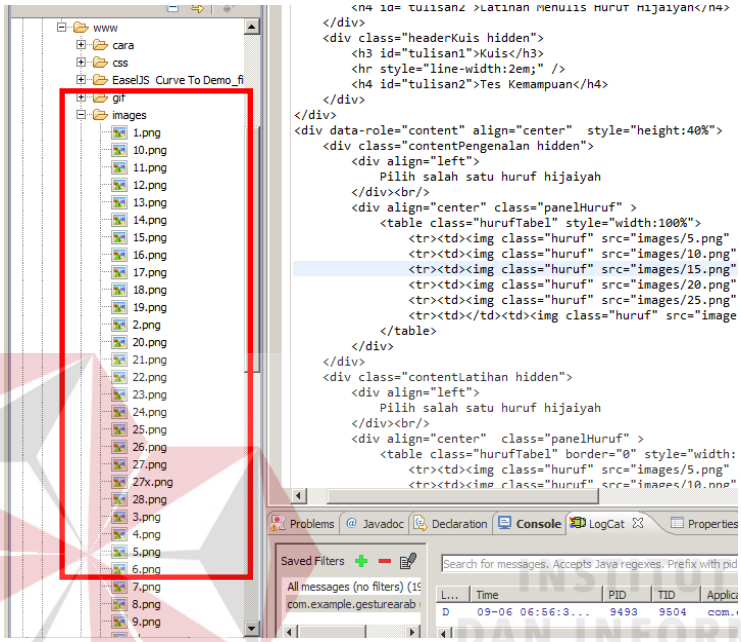
4.3.5 Halaman Pengenalan Aplikasi

Pada fungsi pengenalan hanya menampilkan gambar 28 huruf hijaiyah, tata cara penulisan huruf hijaiyah, gambar gerak yang berformat *gif dan mendengarkan suara pelafalan huruf hijaiyah. Berikut Gambar 4.13 adalah cara menampilkan gambar 28 huruf hijaiyah.

```
<div data-role="content" align="center"  style="height:40%">
    <div class="contentPengenalan hidden">
        <div align="left">
            Pilih salah satu huruf hijaiyah
        </div><br/>
        <div align="center" class="panelHuruf" >
            <table class="hurufTabel" style="width:100%">
                <tr><td><img class="huruf" src="images/5.png" /></td><td><img class="huruf" src="images/4.png'
                <tr><td><img class="huruf" src="images/10.png" /></td><td><img class="huruf" src="images/9.png
                <tr><td><img class="huruf" src="images/15.png" /></td><td><img class="huruf" src="images/14.pr
                <tr><td><img class="huruf" src="images/20.png" /></td><td><img class="huruf" src="images/19.pr
                <tr><td><img class="huruf" src="images/25.png" /></td><td><img class="huruf" src="images/24.pr
                <tr><td></td><td><img class="huruf" src="images/28.png" /></td><td><img class="huruf" src="ima
            </table>
        </div>
    </div>
```

Gambar 4.13 Cara Menampilkan 28 Huruf Hijaiyah

Source gambar 28 huruf hijaiyah disimpan pada folder “images”. Maka dari itu pada Gambar 4.13 memanggil gambar src=”images/5.png”. Berikut Gambar 4.14 adalah letak penyimpanan gambar 28 huruf hijaiyah.



Gambar 4.14 Letak Penyimpanan 28 Huruf Hijaiyah

Untuk fungsi *play* gambar gerak dan suara, sistemnya juga hampir sama dengan menampilkan 28 huruf hijaiyah yang telah dibahas sebelumnya. Gambar gerak dan suara juga disimpan satu paket dengan *project* aplikasi menulis huruf arab. *Button play* dan *sound* mengarahkan fungsi gambar gerak dan suara. Berikut pada Gambar 4.15 adalah gambaran fungsi gambar gerak dan Gambar 4.16 adalah suara.

```
:ontent" align="center"  style="height:40%">
:ontentPengenalan hidden">
gn="center">
 id="animasiDrawing" src="#" style="height:8em" />
'/>
gn="center" >
 src="images/play.png" onclick="playGif()" style="height:4em" /><img src="images/stop.png" onclick="stopGif()"
```

Gambar 4.15 Fungsi gambar gerak

```
ick="stopGif()" style="height:4em" /><img src="images/sound.png" onclick="playAudio()" style="height:4em" />
```

Gambar 4.16 Fungsi Suara

Setelah itu barulah gambar gerak menjalankan proses fungsi dari *playGif* dan fungsi suara menjalankan proses fungsi *playAudio*. Gambaran fungsi keduanya, huruf atau gambar yang dipilih akan disesuaikan dengan index yang dipilih. Misalkan yang dipilih adalah index ke pertama, maka gambar atau suara yang ditampilkan juga index ke-0 berdasarkan urutan nama penyimpanan, index pertama sistem akan mendeteksi index ke-0. Berikut gambaran fungsi seperti terlihat pada Gambar 4.17.

```
function playGif(){
    var file = hrf.indexOf(hurufPilihan);
    $("#animasiDrawing").attr("src", "gif/"+(file+1)+".gif");
    //alert(file);
}

function stopGif(){
    var file = hrf.indexOf(hurufPilihan);
    $("#animasiDrawing").attr("src", "cara/"+(file+1)+".png");
}

function playAudio() {
    var file = hrf.indexOf(hurufPilihan);
    var url=getPhoneGapPath()+'sound/'+(file+1)+'.amr';
    //var ur=getPhoneGapPath()
    //alert(ur);
        var media = new Media(url,
            // success callback
            function() {
                console.log("playAudio():Audio Success");
            },
            // error callback
            function(err) {
                alert(url);
                alert("playAudio():Audio Error: "+err);
        });
    //alert(file);
    //my_media[file].play();
    media.play();
}
```

Gambar 4.17 Fungsi PlayGif dan PlayAudio

4.3.6 Halaman Menulis Huruf Hijaiyah

Dalam menulis huruf hijaiyah dibutuhkan media untuk menulis, pada aplikasi menggunakan *canvas*. *Canvas* terdapat fitur mendeteksi letak koordinat gambar yang ada pada *canvas*. Berikut fungsi dari *canvas*, seperti pada Gambar 4.18.

```
function init() {
    canvas = document.getElementById("myCanvas");
    index = 0;
    colors = ["#828b20", "#b0ac31", "#cbc53d", "#fad779", "#f9e4ad", "#faf2db", "#563512", "#9b4a0b",
    color="black";
    stroke = 10;
    //check to see if we are running in a browser with touch support
    stage = new createjs.Stage(canvas);
    stage.autoClear = false;
    stage.enableDOMEvents(true);

    createjs.Touch.enable(stage);
    createjs.Ticker.setFPS(24);

    drawingCanvas = new createjs.Shape();

    stage.addEventListener("stagemousedown", handleMouseDown);
    stage.addEventListener("stagemouseup", handleMouseUp);

    stage.addChild(drawingCanvas);
    stage.update();
    inislialisasi();
    //$.mobile.hidePageLoadingMsg();
}
```

Gambar 4.18 Pembuatan *Canvas*

Dalam menulis huruf hijaiyah terdapat fungsi ceknilai, yaitu sebagai penilaian benar atau salah huruf yang dipilih dengan tulisan *user*. Sehingga kalkulasi nilai sistem dari fungsi nilai berdasarkan huruf *gestures template* apakah sama dengan huruf *input user*, jika benar maka nilai benar bertambah satu, jika salah maka nilai salah bertambah satu. Berikut pada Gambar 4.19 fungsi dari ceknilai.

```
function acakHuruf(){
    hurufPilihan=hrf[Math.floor((Math.random()*hrf.length))];
    //hurufPilihan=hrf[Math.floor((Math.random()*3))];
}

function cekNilai(){
    var status='';
    var hasil='';
    if(hurufPilihan==hurufGesture){
        benar++; hasil = 'Benar';
    }
    else{
        salah++; hasil = 'Salah';
    }
    if(pilihanMenu=='Kuis'){
        previewPointData.push(pointCharacter);
        previewStatusData.push(hasil);
        //alert('ok');
        //addImage(pointCharacter,hasil,tipeKuis);
    }
```

Gambar 4.19 Fungsi CekNilai

4.3.7 Metode $N

a. Fungsi Inisialisasi

Fungi Inisialisasi adalah tahap awal dalam aplikasi, terdapat variable _points, _stroke dan _r kemudian memanggil fungsi loadcode(). Variable _points adalah fungsi untuk mengambil titik koordinat dari *input* user dan _strokes adalah fungsi untuk menentukan jumlah stroke dari *input* user. Kemudian Fungsi loadcode() adalah isi dari koordinat huruf arab yang sudah ditentukan, kemudian terdapat variable jumlah stroke, points sebagai pencocokannya. Berikut Fungsi Inisialisasi pada Gambar 4.20.

```
48  function inislialisasi(){
49      _points = new Array(); // point array for current stroke
50      _strokes = new Array(); // array of point arrays
51      _r = new NDollarRecognizer(true);
52
53      loadCode();
54  }
```

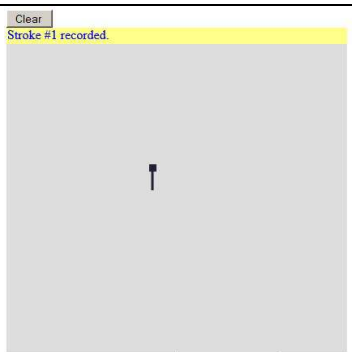
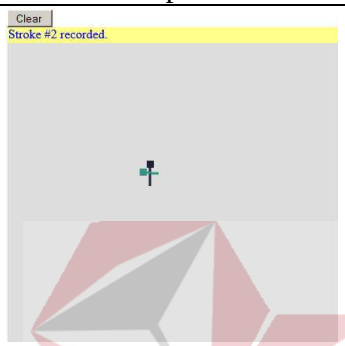
Gambar 4.20 Fungsi Inisialisasi

a. Fungsi LoadCode dan AddCustom

Fungsi loadcode terdapat titik koordinat huruf “alif” sampai dengan “ya” yang telah ditentukan atau yang disebut dengan *gestures template*. Penggunaan *gestures template* menggunakan *index* gambar huruf arab “alif” sampai “ya”, sehinga banyaknya *index* sebanyak 28. Pada fungsi loadcode terdapat fungsi untuk menentukan jumlah *stroke* dengan data koordinat yang menggunakan json seperti pada Gambar 4.21, kemudian diolah lagi pada fungsi AddCustom pada Gambar 4.23. Jumlah *stroke* yang disimpan nantinya dipakai sebagai *variable* pada fungsi addcustom yang di dalamnya terdapat fungsi addGestures, sebagai pencocokan pola huruf arab pada fungsi recognizes.

```
function loadCode(){
    //("Alif","Ba","Ta","Tsa","Ja","Ha","Kha","Da","Dza","Ro","Zaa","Sin","Syin","Saad","Dhad","Tho'","Dho'","'Ayn","Ghoyn","Faa","Qof","Kaaf","Lam
    /* Alif */ codeHrf.push('[[{"X":201,"Y":47},{"X":201,"Y":48},{"X":201,"Y":49},{"X":201,"Y":50},{"X":201,"Y":51},{"X":201,"Y":52},{"X":201,"Y":53}
    /* Ba' */ codeHrf.push('[[{"X":320,"Y":183},{"X":320,"Y":184},{"X":320,"Y":185},{"X":321,"Y":188},{"X":322,"Y":189},{"X":323,"Y":190},{"X":325,"Y
    /* Ta' */ codeHrf.push('[[{"X":292,"Y":162},{"X":293,"Y":164},{"X":295,"Y":164},{"X":296,"Y":166},{"X":297,"Y":166},{"X":299,"Y":166},{"X":300,"Y
    /* Tsa */ codeHrf.push('[[{"X":340,"Y":222},{"X":340,"Y":222},{"X":340,"Y":222},{"X":340,"Y":222},{"X":340,"Y":222},{"X":340,"Y":222},{"X":340,"Y":223},{"X":341,"Y
    /* Ja */ codeHrf.push('[[{"X":163,"Y":128},{"X":164,"Y":128},{"X":166,"Y":128},{"X":168,"Y":128},{"X":169,"Y":128},{"X":170,"Y":128},{"X":171,"Y"
    /* Ha */ codeHrf.push('[[{"X":133,"Y":103},{"X":133,"Y":103},{"X":133,"Y":103},{"X":133,"Y":103},{"X":133,"Y":103},{"X":134,"Y":103},{"X":135,"Y"
    /* Kha */ codeHrf.push('[[{"X":133,"Y":103},{"X":133,"Y":103},{"X":133,"Y":103},{"X":133,"Y":103},{"X":133,"Y":103},{"X":134,"Y":103},{"X":135,"Y
    /* Da */ codeHrf.push('[[{"X":214,"Y":124},{"X":214,"Y":124},{"X":214,"Y":124},{"X":214,"Y":124},{"X":214,"Y":124},{"X":214,"Y":124},{"X":214,"Y"
    /* Dza */ codeHrf.push('[[{"X":214,"Y":124},{"X":214,"Y":124},{"X":214,"Y":124},{"X":214,"Y":124},{"X":214,"Y":124},{"X":214,"Y":124},{"X":214,"Y
    /* Ro */ codeHrf.push('[[{"X":236,"Y":91},{"X":236,"Y":92},{"X":236,"Y":93},{"X":236,"Y":94},{"X":237,"Y":94},{"X":237,"Y":95},{"X":237,"Y":95},{
    /* Zaa */ codeHrf.push('[[{"X":236,"Y":91},{"X":236,"Y":92},{"X":236,"Y":93},{"X":236,"Y":94},{"X":237,"Y":94},{"X":237,"Y":95},{"X":237,"Y":95},
    /* Sin */ codeHrf.push('[[{"X":335,"Y":135},{"X":335,"Y":135},{"X":335,"Y":135},{"X":336,"Y":137},{"X":336,"Y":138},{"X":336,"Y":139},{"X":336,"
    /* Syin */ codeHrf.push('[[{"X":335,"Y":135},{"X":335,"Y":135},{"X":335,"Y":135},{"X":336,"Y":137},{"X":336,"Y":138},{"X":336,"Y":139},{"X":336,"
    /* Saad */ codeHrf.push('[[{"X":246,"Y":186},{"X":246,"Y":184},{"X":246,"Y":183},{"X":246,"Y":182},{"X":246,"Y":181},{"X":246,"Y":180},{"X":247,"
    /* Dhad */ codeHrf.push('[[{"X":246,"Y":186},{"X":246,"Y":184},{"X":246,"Y":183},{"X":246,"Y":182},{"X":246,"Y":181},{"X":246,"Y":180},{"X":247,"
    /* Tho' */ codeHrf.push('[[{"X":171,"Y":85},{"X":171,"Y":86},{"X":171,"Y":87},{"X":171,"Y":89},{"X":171,"Y":89},{"X":171,"Y":89},{"X":171,"Y":90}
    /* Dho' */ codeHrf.push('[[{"X":171,"Y":85},{"X":171,"Y":86},{"X":171,"Y":87},{"X":171,"Y":89},{"X":171,"Y":89},{"X":171,"Y":89},{"X":171,"Y":90}
    /* 'Ayn */ codeHrf.push('[[{"X":238,"Y":100},{"X":237,"Y":100},{"X":235,"Y":100},{"X":234,"Y":100},{"X":233,"Y":100},{"X":232,"Y":100},{"X":231,"
    /* Ghoyn */ codeHrf.push('[[{"X":215,"Y":108},{"X":215,"Y":108},{"X":215,"Y":108},{"X":214,"Y":107},{"X":214,"Y":107},{"X":214,"Y":106},{"X":213,
    /* Faa */ codeHrf.push('[[{"X":327,"Y":186},{"X":325,"Y":186},{"X":324,"Y":186},{"X":323,"Y":186},{"X":322,"Y":186},{"X":321,"Y":186},{"X":320,"Y
    /* Qof */ codeHrf.push('[[{"X":309,"Y":177},{"X":308,"Y":177},{"X":307,"Y":177},{"X":305,"Y":177},{"X":304,"Y":177},{"X":302,"Y":177},{"X":301,"Y
    /* Kaaf */ codeHrf.push('[[{"X":296,"Y":148},{"X":296,"Y":149},{"X":296,"Y":150},{"X":296,"Y":151},{"X":296,"Y":152},{"X":296,"Y":153},{"X":296,"
    /* Lam */ codeHrf.push('[[{"X":290,"Y":118},{"X":290,"Y":119},{"X":290,"Y":120},{"X":290,"Y":121},{"X":290,"Y":122},{"X":290,"Y":123},{"X":290,"Y
    /* Miim */ codeHrf.push('[[{"X":155,"Y":110},{"X":155,"Y":110},{"X":155,"Y":110},{"X":155,"Y":109},{"X":155,"Y":108},{"X":155,"Y":107},{"X":155,"
    /* Nun */ codeHrf.push('[[{"X":295,"Y":189},{"X":295,"Y":190},{"X":296,"Y":190},{"X":296,"Y":191},{"X":298,"Y":191},{"X":298,"Y":192},{"X":298,"Y
    /* Waw */ codeHrf.push('[[{"X":287,"Y":163},{"X":287,"Y":163},{"X":287,"Y":163},{"X":287,"Y":162},{"X":286,"Y":162},{"X":285,"Y":162},{"X":284,"Y
    /* Haa */ codeHrf.push('[[{"X":254,"Y":152},{"X":254,"Y":153},{"X":254,"Y":154},{"X":254,"Y":155},{"X":254,"Y":156},{"X":254,"Y":158},{"X":254,"Y
    /* Ya' */ codeHrf.push('[[{"X":311,"Y":176},{"X":310,"Y":176},{"X":309,"Y":176},{"X":308,"Y":175},{"X":307,"Y":175},{"X":307,"Y":174},{"X":306,"Y
    //alert('ok');
    for(var i = 0; i<codeHrf.length;i++){
        arr= JSON.parse(codeHrf[i]);
        onClickAddCustom(hrf[i]);
    }
```

Gambar 4.21 Fungsi LoadCode

 Stroke pertama	[[{"X":180,"Y":171},{"X":180,"Y":172},{"X":180,"Y":173},{"X":180,"Y":174},{"X":180,"Y":176},{"X":180,"Y":177},{"X":180,"Y":179},{"X":180,"Y":180},{"X":180,"Y":182},{"X":180,"Y":183},{"X":180,"Y":185},{"X":180,"Y":186},{"X":180,"Y":188},{"X":180,"Y":189},{"X":180,"Y":191},{"X":180,"Y":192},{"X":180,"Y":194},{"X":180,"Y":195},{"X":180,"Y":197},{"X":180,"Y":198},{"X":180,"Y":200}]
 Stroke pertama dan kedua	[[{"X":180,"Y":171},{"X":180,"Y":172},{"X":180,"Y":173},{"X":180,"Y":174},{"X":180,"Y":176},{"X":180,"Y":177},{"X":180,"Y":179},{"X":180,"Y":180},{"X":180,"Y":182},{"X":180,"Y":183},{"X":180,"Y":185},{"X":180,"Y":186},{"X":180,"Y":188},{"X":180,"Y":189},{"X":180,"Y":191},{"X":180,"Y":192},{"X":180,"Y":194},{"X":180,"Y":195},{"X":180,"Y":197},{"X":180,"Y":198},{"X":180,"Y":200}],[{"X":171,"Y":181},{"X":172,"Y":181},{"X":172,"Y":182},{"X":173,"Y":182},{"X":174,"Y":182},{"X":175,"Y":183},{"X":177,"Y":183},{"X":178,"Y":183},{"X":179,"Y":183},{"X":180,"Y":183},{"X":181,"Y":184},{"X":182,"Y":184},{"X":183,"Y":184},{"X":184,"Y":184},{"X":185,"Y":184},{"X":186,"Y":184},{"X":187,"Y":184},{"X":188,"Y":184},{"X":189,"Y":184},{"X":190,"Y":184},{"X":191,"Y":184}]]

Gambar 4.22 Contoh *Stroke* Loadcode

Pada Gambar 4.22 adalah contoh pembacaan titik koordinat pada *gestures template*. Contoh penggunaan pembacaan *stroke*, pada tulisan koordinat berwarna biru adalah stroke awal, dan tulisan warna coklat adalah stroke yang kedua. Pada data titik koordinat menggunakan json. Pembatas *stroke* dibatasi oleh tanda “[..]”. Pada contoh diatas terdapat dua tanda pembatas, sehingga jumlah *stroke*-nya sebanyak dua *stroke* untuk pola yang dicontohkan pada Gambar 4.22.

```
function onClickAddCustom(str)
{
    var name = str; //'Ba\''; //document.getElementById('custom').value;
    //_strokes =
    //alert(str);
    var num = _r.AddGesture(name,true,arr);
    _points.length = 0;
    return false;
}
```

Gambar 4.23 Fungsi AddCustom

Pada fungsi AddCustom adalah mempunyai *variable* name, yang nantiya dicocokkan dengan fungsi $N yang didalamnya terdapat fungsi addGestures. Sehingga *variable* name pada fungsi addCustom mempunyai nilai berupa nama yang berguna dalam pencocokan pada fungsi *recognizes* dengan *input user*.

b. Fungsi AddGestures

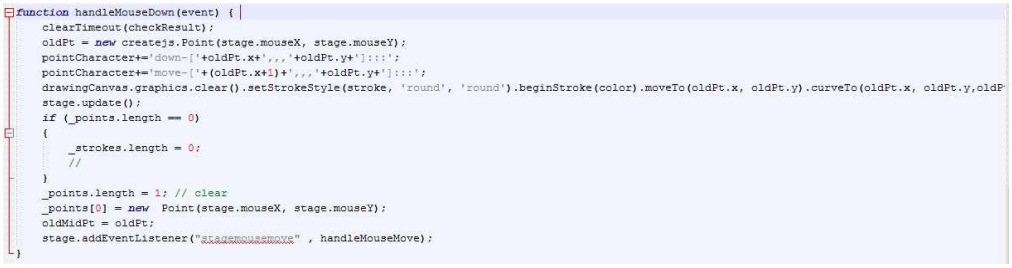
Fungsi AddGestures mempunyai *variable* jumlah *stroke* dan nama *gestures*. Fungsi ini nantinya akan memberikan nilai *variable* name pada fungsi addCustom. Berikut fungsi addgestures pada Gambar 4.24.

```
this.AddGesture = function(name, useBoundedRotationInvariance, strokes)
{
    this.Multistrokes[this.Multistrokes.length] = new Multistroke(name, useBoundedRotationInvariance, strokes);
    var num = 0;
    for (var i = 0; i < this.Multistrokes.length; i++) {
        if (this.Multistrokes[i].Name == name)
            num++;
    }
    return num;
}
```

Gambar 4.24 Fungsi AddGestures

c. Fungsi MouseDown

Fungsi MouseDown adalah kondisi ketika *user* pertama kali menyentuh layar, sehingga mendeteksi titik koordinat awal dalam penulisan *gestures* huruf arab. Fungsi MouseDown dapat dilihat pada Gambar 4.25.



```
function handleMouseDown(event) {
    clearTimeout(checkResult);
    oldPt = new createjs.Point(stage.mouseX, stage.mouseY);
    pointCharacter+='down-['+oldPt.x+',,,'+oldPt.y+']:::';
    pointCharacter+='move-['+(oldPt.x+1)+',,,'+oldPt.y+']:::';
    drawingCanvas.graphics.clear().setStrokeStyle(stroke, 'round', 'round').beginStroke(color).moveTo(oldPt.x, oldPt.y).curveTo(oldPt.x, oldPt.y,oldP
    stage.update();
    if (_points.length == 0)
    {
        _strokes.length = 0;
        //
    }
    _points.length = 1; // clear
    _points[0] = new  Point(stage.mouseX, stage.mouseY);
    oldMidPt = oldPt;
    stage.addEventListener("stagemousemove" , handleMouseMove);
}
```

Gambar 4.25 Fungsi MouseDown

d. Fungsi MouseMove

Fungsi MouseMove adalah pengenalan titik koordinat setelah gerakan titik awal atau mousedown. Setiap gerakan akan terdeteksi titik koordinat dan disimpan pada fungsi mousemove. Fungsi MuseMove dapat dilihat pada Gambar 4.26.

```
function handleMouseMove(event) {
    var midPt = new createjs.Point(oldPt.x + stage.mouseX>>1, oldPt.y+stage.mouseY>>1);
    pointCharacter+='move-['+(midPt.x)+',,,'+(midPt.y)+']:::';
    _points[_points.length] = new Point(stage.mouseX, stage.mouseY);
    drawingCanvas.graphics.clear().setStrokeStyle(stroke, 'round', 'round').beginStroke(color).moveTo(midPt.x, midPt.y).curveTo(oldPt.x, oldPt.y, old

    oldPt.x = stage.mouseX;
    oldPt.y = stage.mouseY;

    oldMidPt.x = midPt.x;
    oldMidPt.y = midPt.y;

    stage.update();
}
```

Gambar 4.26 Fungsi MouseMove

e. Fungsi MouseUp

Fungsi MouseUp adalah pengenalan ketika gerakan *gestures* telah berkahir atau tidak menyentuh layar. Apabila selama tiga detik tidak ada masukan *gestures*, maka akan dilakukan pengecekan nilai. Pada fungsi ini memberikan waktu sebanyak tiga detik, jika selama tiga detik tidak ada lagi *input*, maka proses pengecekan dan penilaian akan dilakukan. Pada Fungsi MouseUp terdapat pengecekan sebuah nilai. Fungsi MouseUp dapat dilihat pada Gambar 4.27.

```
function handleMouseUp(event) {
    pointCharacter+='up:::';
    _strokes[_strokes.length] = _points.slice(); // add new copy to set
    stage.removeEventListener("stagemousemove" , handleMouseMove);
    touchCount++;
    checkResult=setTimeout(function(){getResult()},3000);
}
```

Gambar 4.27 Fungsi MouseUp

f. Fungsi GetResult

Fungsi GetResult adalah fungsi dalam penilaian yang di dalamnya terdapat pengecekan benar dan salah tulisan dari *user*. Pada fungsi ini juga memanggil fungsi _r.recognizes yang ada di dalam metode $N. Setelah melakukan pengecekan salah atau benar, fungsi nilai dilakukan. Fungsi GetResult dapat dilihat pada Gambar 4.28.

```
function getResult(){
    try{
    //alert('ok');
    if (_strokes.length > 1 || (_strokes.length == 1 && _strokes[0].length >= 10))
    {
        //alert('ok');
        var result = _r.Recognize(_strokes, true, false, false);
        //alert(result.Name);
        hurufGesture=result.Name;
        //alert(hurufGesture);
        cekNilai();
    }

    _points.length = 0; // clear and signal to clear strokes on next mousedown
    }
    catch(e){
        //alert(e.message);
        hurufGesture='';
        //alert(hurufGesture);
        cekNilai();
    }
}
```

Gambar 4.28 Fungsi GetResult

Pada Fungsi GetResult, fungsi recognizes dilakukan, sehingga mendapatkan nilai huruf *gestures*. Huruf *gestures* nantinya akan sebagai *variable* penilaian pada fungsi nilai.

g. Fungsi Recognizes $N

Pada fungsi recognizes, adalah fungsi untuk model pencocokannya. Fungsi recognizes mempunyai beberapa *variable* sebagai model pencocokan metode $N, seperti *stroke*, *points*, *rotate*, *boundedrotation* dan fungsi lainnya yang mendukung dalam proses pencocokan $N. Inti dari metode $N terdapat pada fungsi ini. Sehingga nilai *variable* pada *input user* dengan data koordinat *gestures template* akan dicocokkan pada fungsi ini. *Variable* yang dipakai dalam pencocokan dengan menggunakan nama huruf arab berdasarkan *index* dari loadcode atau data titik koordinat *gestures template* dengan *index* penamaan huruf arab yang ada pada gambar huruf arab pada aplikasi. Seperti pada Gambar 4.29.

```
//
this.Recognize = function(strokes, useBoundedRotationInvariance, requireSameNoOfStrokes, useProtractor)
{
    var points = CombineStrokes(strokes); // make one connected unistroke from the given strokes
    points = Resample(points, NumPoints);
    var radians = IndicativeAngle(points);
    points = RotateBy(points, -radians);
    points = ScaleDimTo(points, SquareSize, OneDThreshold);
    if (useBoundedRotationInvariance)
        points = RotateBy(points, +radians); // restore
    points = TranslateTo(points, Origin);
    var startv = CalcStartUnitVector(points, StartAngleIndex);
    var vector = Vectorize(points, useBoundedRotationInvariance); // for Protractor

    var b = +Infinity;
    var u = -1;


    //alert(touchCount);
    var tc = 1;
    if(touchCount > 1){
        //tc = touchCount-1;
    }
    //alert(touchCount);
    var touchHurufCount = hrfByTouchCount[hrfTouchCount.indexOf(touchCount)];
    /*---------------------------*/

    for (var i = 0; i < this.Multistrokes.length; i++) // for each multistroke
    {
        var nm = this.Multistrokes[i].Name;
        if(touchHurufCount.indexOf(nm) != -1 && hrf.indexOf(nm) !=-1){
            if (!requireSameNoOfStrokes || strokes.length == this.Multistrokes[i].NumStrokes) // optional -- only attempt match when same # of s
            {
                for (var j = 0; j < this.Multistrokes[i].Unistrokes.length; j++) // each unistroke within this multistroke
                {
                    if (AngleBetweenUnitVectors(startv, this.Multistrokes[i].Unistrokes[j].StartUnitVector) <= AngleSimilarityThreshold) // str
                    {
                        var d;
                        if (useProtractor) // for Protractor
                            d = OptimalCosineDistance(this.Multistrokes[i].Unistrokes[j].Vector, vector);
                        else // Golden Section Search (original $N)
                            d = DistanceAtBestAngle(points, this.Multistrokes[i].Unistrokes[j], -AngleRange, +AngleRange, AnglePrecision);
                        if (d < b) {
                            b = d; // best (least) distance
                            u = i; // multistroke owner of unistroke
                        }
                    }
                }
            }
        }
    }
    return (u == -1) ? new Result("No match.", 0.0) : new Result(this.Multistrokes[u].Name, useProtractor ? 1.0 / b : 1.0 - b / HalfDiagonal);
};
```

Gambar 4.29 Fungsi Recognizes

Inti dari fungsi recognizes adalah untuk memberikan nilai dari *variable* huruf *gestures* pada fungsi GetResult.

h. Fungsi Nilai

Pada fungsi Nilai adalah kondisi dimana *variable* namagestures sudah ada, kemudian pada fungsi nilai di cocokkan dengan hurufpilihan. *Variable* huruf pilihan adalah soal dari kuis. Ketika hurufgestures==hurufpilihan maka akan bernilai benar. Berikut gambaran fungsi nilai pada Gambar 4.30

```
function cekNilai(){
    var status='';
    var hasil='';
    if(hurufPilihan==hurufGesture){
        benar++; hasil = 'Benar';
    }
    else{
        salah++; hasil = 'Salah';
    }
    if(pilihanMenu=='Kuis'){
        previewPointData.push(pointCharacter);
        previewStatusData.push(hasil);
        previewSoalData.push(hurufPilihan);
        //alert('ok');
        //addImage(pointCharacter,hasil,tipeKuis);
    }
    onClickClearStrokes();
```

Gambar 4.30 Fungsi Nilai

A. Implementasi *Interfaces*

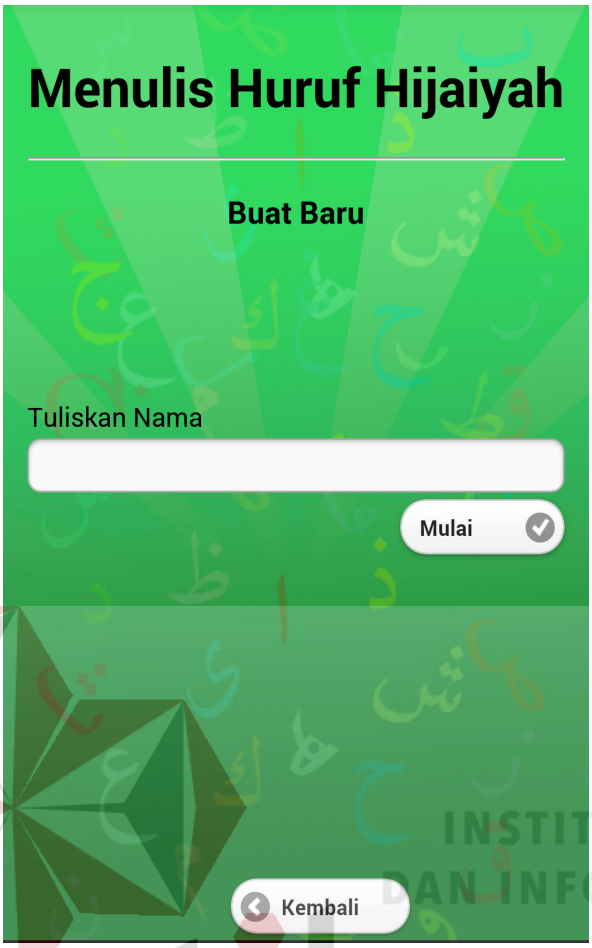
Implementasi *interfaces* menggambarkan tampilan dari aplikasi yang dibangun yaitu aplikasi menulis huruf arab berbasis Android. Berikut ini adalah penjelasan implementasi *interfaces* dari aplikasi menulis huruf arab.

1. Tampilan Menu Pembuka

Pada saat membuka aplikasi, *user* akan masuk pada menu pembuka, seperti pada Gambar 4.31.

Pada menu pembuka, sistem akan menampilkan dua menu, yaitu menu “buat baru” dan “menu “lanjut”. Pada menu “buat baru” beguna untuk ketika *user* pertama kali menjalankan aplikasi, *user* diwajibkan untuk membuat *user id* baru untuk masuk ke dalam Aplikasi Menulis Huruf Arab. Kemudian menu “lanjut” digunakan ketika *user* sudah mempunyai atau pernah membuat *user* untuk masuk ke dalam Aplikasi Menulis Huruf Arab. Untuk lebih jelasnya, berikut pada Gambar 4.32 adalah tampilan menu “pembuka”.

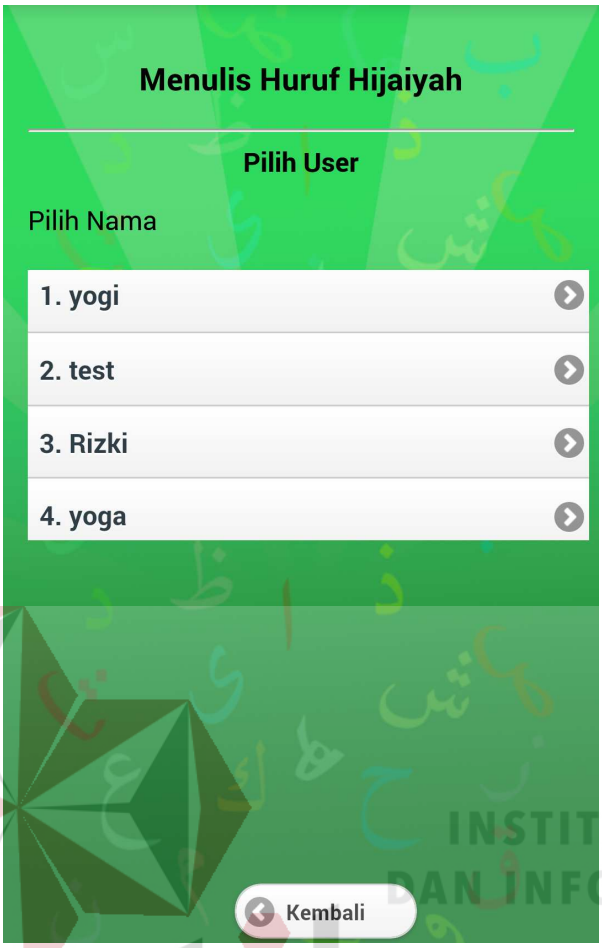
Gambar 4.31 Tampilan Menu Pembuka



Gambar 4.32 Tampilan Menu Buat Baru User

2. Tampilan *Login*

Tampilan menu *login* berfungsi untuk mengidentifikasi *user* dalam menggunakan aplikasi. Sistem harus mengetahui identitas *user*, karena didalam sistem nantinya akan menyimpan histori nilai. Bila *user* belum mempunyai *user id* untuk *login*, maka pada tampilan login ada fitur untuk membuat *user id* baru dan memilih menu "buat baru". Bagi *user* yang sudah memiliki *user id*, maka *user* hanya tinggal memilih *user id* yang ditampilkan oleh sistem pada menu "lanjut". Tampilan *login* terlihat pada Gambar 4.33.



Gambar 4.33 Menu Daftar User

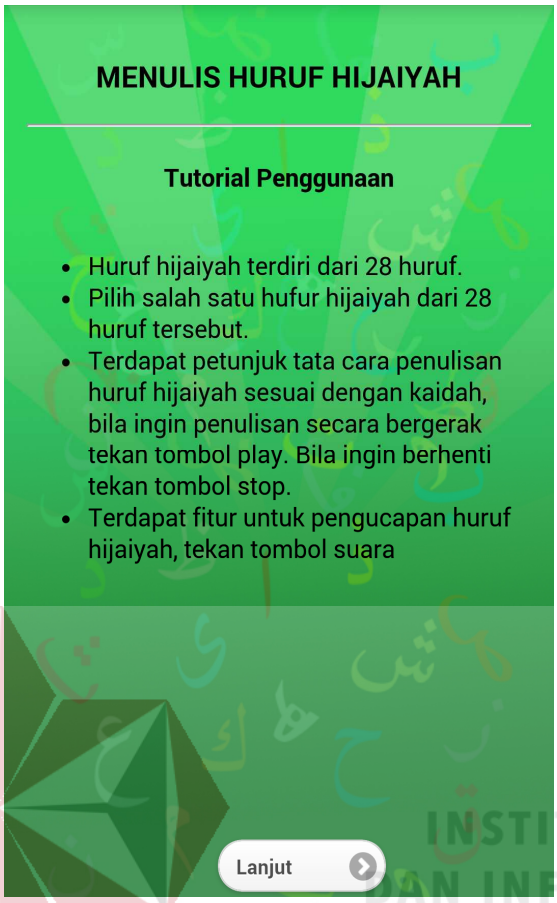
3. Tampilan Menu Utama

Setelah memilih *user*, berikutnya tampil pilih menu utama yang dapat dilihat pada Gambar 4.34, karena bila *user* tidak memiliki *user login* maka tidak akan dapat masuk ke dalam menu utama pada Aplikasi Menulis Huruf Arab. Pada menu ini juga menampilkan nama *user* yang pernah *login* atau pernah membuat. Setelah itu, sistem akan masuk pada menu "utama" pada Aplikasi Menulis Huruf Arab, sehingga *user* dapat memilih fitur pembelajaran yang akan dipelajari terlebih dahulu pada menu utama, seperti pengenalan huruf arab, latihan menulis huruf arab, kuis (soal evaluasi dan tes kemampuan) dan nilai.

Gambar 4.34 Tampilan Menu Utama

4. Tampilan Pengenalan Huruf Arab

Tampilan menu pengenalan huruf arab berisi tentang tutorial atau tata cara penulisan huruf arab yang benar sesuai dengan kaidah. Setelah *user* memilih menu "pengenalan hijaiyah", pertama nanti akan muncul seperti tata cara penggunaan atau tutorial seperti pada Gambar 4.35. Kedua *user* harus memilih salah satu dari 28 huruf hijaiyah, seperti pada Gambar 4.36. Ketiga, *user* masuk ke dalam menu tata cara penulisan huruf hijaiyah yang sesuai dengan kaidah seperti pada Gambar 4.37.



Gambar 4.35 Tata Cara Penggunan Menu yang dipilih

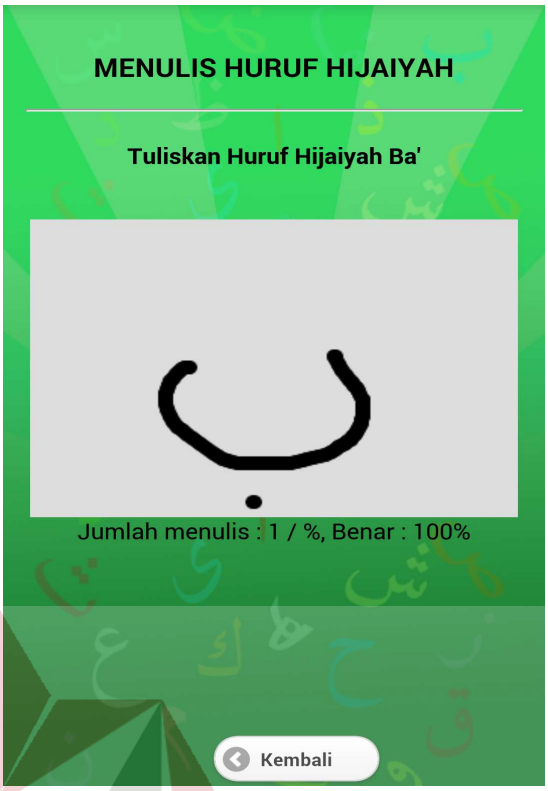
Gambar 4.36 Daftar 28 Huruf Hijaiyah yang harus di pilih user

Gambar 4.37 Tata Cara Penulisan Huruf Hijaiayah Sesuai Kaidah.

Pada Gambar 4.37 terdapat beberapa fitur yaitu tombol play, stop dan suara. Masing-masing fitur mempunyai fungsi tersendiri. Fungsi tombol play berfungsi untuk melihat tata cara penulisan denga tulisan bergerak dan tombol stop bilan ingin menghentikan gambar tulisan bergerak. Fungsi tombol suara adalah untuk mendengarkan pelafalan huruf hijaiyah yang dipilih.

5. Tampilan Latihan Menulis Huruf Arab

Pada menu latihan menulis huruf hijaiyah, awalnya *user* akan diberikan info tata cara penggunaan menu “latihan menullis hijaiyah”. Setelah itu *user* harus memilih salah satu huruf hijaiyah seperti pada Gambar 4.36. Setelah itu masuk ke dalam menu inti dari “latihan menulis hijaiyah” pada Gambar 4.38.



Gambar 4.38 Tampilan Menu Latihan Menulis Hijaiyah

Pada fitur ini *user* akan mencoba belajar menulis huruf hijaiyah yang telah dipilih pada menu sebelumnya. Kemudian hasil tulisan *user* (*gestures*) akan di cocokkan keberhasilan *user* dalam menulis hijaiyah. Sistem akan memberikan informasi jumlah menulis yang telah dilakukan *user* dan juga prosentase benar. Dapat dilihat pada Gambar 4.38.

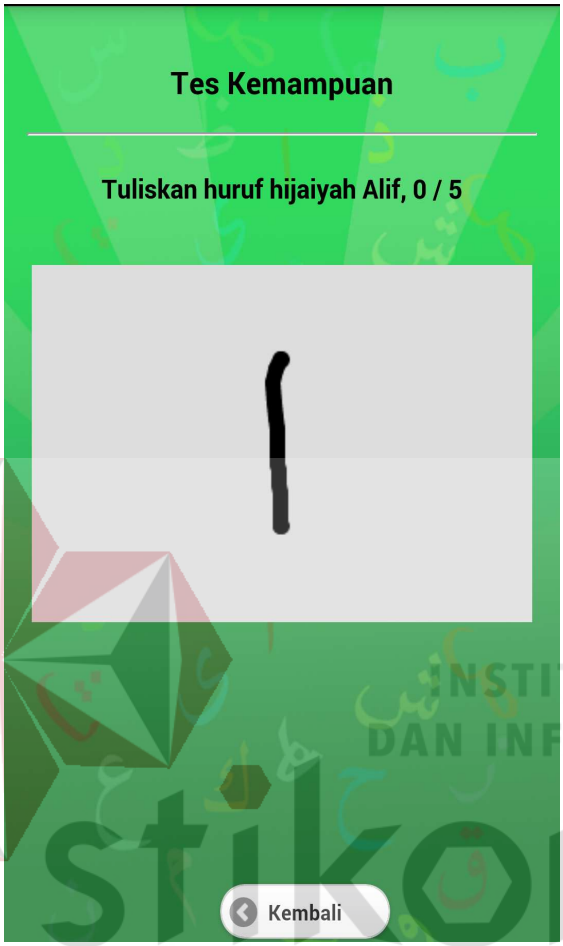
6. Tampilan Kuis

Pada menu kuis di harapkan dapat melatih *user* dalam mengenali dan menghafal 28 huruf hijaiyah. Pada aplikasi terdapat dua macam kuis yaitu:

a. Tampilan Tes Kemampuan

Pada kuis tes kemampuan, *user* awalnya harus memilih salah satu huruf hijaiyah, kemudian *user* harus menuliskan jawaban sesuai huruf yang dipilih,

misalkan *user* memilih huruf alif, maka *user* memasukkan tulisan huruf arab alif sebanyak lima kali. Untuk lebih jelasnya seperti terlihat pada Gambar 4.39.



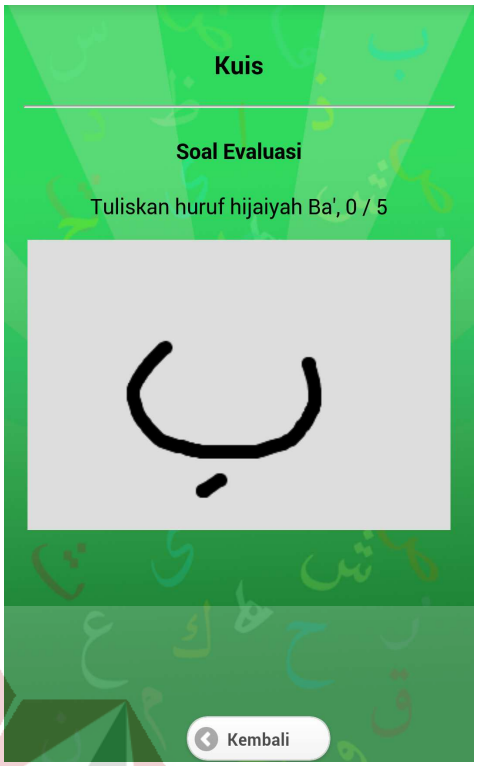
Gambar 4.39 Tampilan Kuis Tes Kemampuan

Setelah *user* menuliskan sebanyak lima kali, sistem akan memberikan info kepada *user* atas hasil tes kemampuan yang dilakukan benar dan salahnya, karena pada proses ini sistem melakukan kalkulasi nilai salah dan benar berdasarkan *input gestures* yang dilakukan oleh *user*, kemudian oleh sistem akan menyimpan pada *database* nilai. Untuk jelasnya, seperti yang terlihat pada Gambar 4.40.

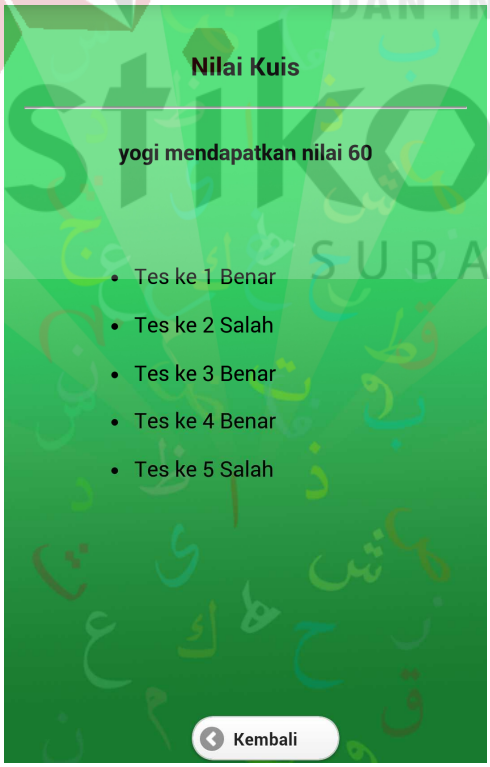
Gambar 4.40 Info Hasil user melakukan tes kemampuan

b. Tampilan Soal Evaluasi

Pada kuis soal evaluasi, *user* dapat melakukan latihan dan uji coba dalam menulis huruf arab. Pada fitur soal evaluasi, nantinya *user* mendapatkan perintah dari sistem secara acak, kemudian *user* harus memberikan jawaban atas perintah sistem seperti pada Gambar 4.41. Setelah *user* memasukkan jawaban, sistem akan mengecek dan memberikan nilai serta pemberitahuan jawaban salah atau benar. Seperti terlihat pada Gambar 4.42.



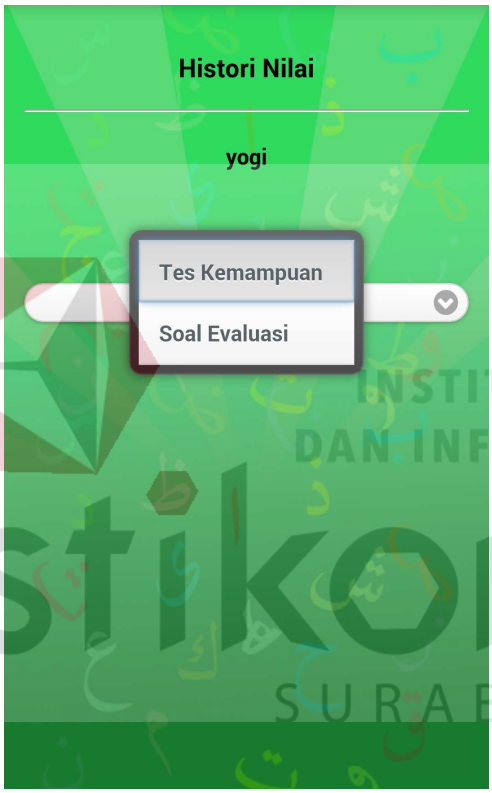
Gambar 4.41 Tampilan Kuis Soal Evaluasi



Gambar 4.42 Hasil user melakukan tes soal evaluasi

7. Tampilan Nilai

Pada aplikasi menulis huruf arab, *user* dapat melihat *history* nilai dari uji coba atau latihan yang dilakukan oleh *user*. Ketika *user* menggunakan menu kuis, sistem menyimpan hasil pengecekan sistem pada waktu *user* melakukan latihan kuis. Pada penilaian terdapat dua macam nilai kuis yaitu nilai tes kemampuan dan soal evaluasi, seperti yang terlihat pada Gambar 4.43.



Gambar 4.43 Tampilan Menu Nilai

Pada menu ini *user* dapat melihat nilai tes kemampuan dan soal evaluasi. Apabila *user* ingin melihat nilai pada kuis tes kemampuan, maka *user* tinggal memilih nilai tes kemampuan, seperti yang terlihat pada Gambar 4.44, dan bila ingin melihat nilai pada soal evaluasi *user* memilih nilai soal evaluasi seperti pada gambar 4.45.

Gambar 4.44 Tampilan Nilai Kuis Tes Kemampuan

Gambar 4.45 Tampilan Nilai Kuis Soal Evaluasi

4.4 Evaluasi Sistem

Setelah melakukan impelementasi sistem, tahap selanjutnya adalah melakukan uji coba dan evaluasi terhadap sistem. Tujuan evaluasi ini adalah untuk mengetahui apakah aplikasi yang telah dibuat ini sudah berjalan dengan baik dan sesuai dengan tujuan atau output yang diharapkan. Uji coba ini akan

dilakukan dengan menggunakan metode *black box testing.* Adapun evaluasi yang dilakukan adalah sebagai berikut:

4.4.1 Uji Coba Fungsi Aplikasi

Uji coba fungsi aplikasi dilakukan dengan memberikan aplikasi kepada *user* untuk menjalankan aplikasi menulis huruf arab. Uji coba kepada *user* dilakukan untuk melakukan penilaian terhadap aplikasi yang dibangun, apakah sudah dibuat sesuai tujuan dan berhasil sesuai harapan.

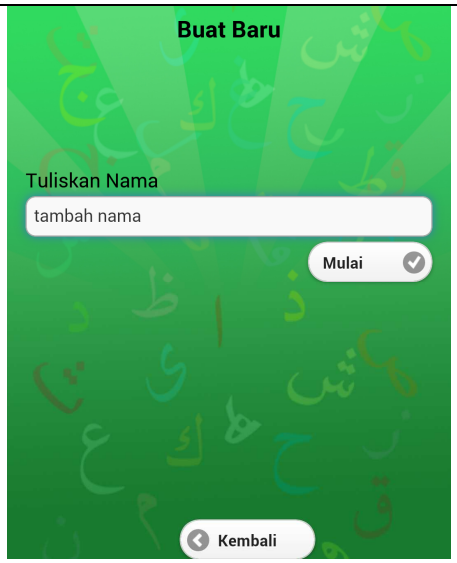
A. Hasil Uji Coba Daftar *User*

Halaman daftar user berfungsi untuk menambah nama *user*, menampilkan nama *user*, dan mengirimkan pesan ke database agar aplikasi dapat memuat hasil kerja dari *user* yang terpilih. Desain uji coba daftar *user* dapat dilihat pada Tabel 4.1.

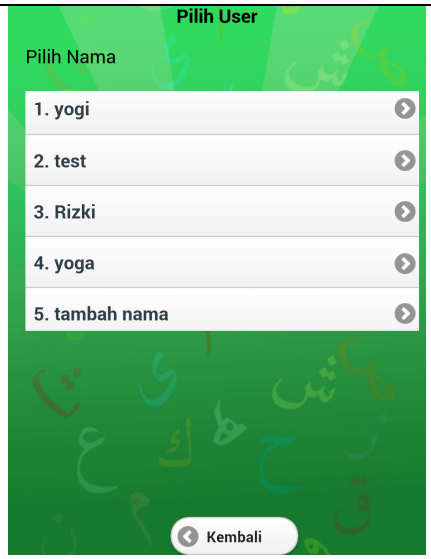
Tabel 4.1 Keterangan *Test Case ID* 1 Fungsi Buat *User Login*

Test Case ID	**Tujuan**	***Input***	***Output* yang diharapkan**	**Status**
1.	Menambah Nama *User*	Pilih menu buat baru, ke kemudian masukkan nama *user* dan pilih tombol mulai	Nama *user* tampil pada menu utama aplikasi.	Sukses

Pada tabel 4.1 merupakan fungsi uji coba terhadap *user login*. Pembuktian ini berstatus sukses karena dapat ditambahkan *user* baru, karena pada menu lanjut akan tampil nama *user* baru. Berikut pembuktian sukses dalam menambah *user* pada Gambar 4.46.



Tampilan buat *user* baru



Tampilan ketika *user* sudah disimpan

Gambar 4.46 Pembuktian Test Case ID 1 Fungsi Buat *User Login*

Pada Gambar 4.46 adalah tampilan membuat *user* baru. *User* yang telah dibuat akan disimpan dan kemudian bisa ditampilkan pada menu lanjut yang berisi *user* yang sudah pernah membuat atau yang disimpan oleh aplikasi.

Uji berikutnya adalah ketika *user* memilih *userid*-nya, dapat masuk ke dalam menu utama aplikasi menulis huruf arab. Setelah masuk ke dalam menu utama, apakah *user* yang masuk sesuai dengan *user* yang dipilih. Berikut hasil uji pada tabel 4.2.

Tabel 4.2 Keterangan *Test Case ID* 2 Fungsi *User Login*

Test Case ID	**Tujuan**	***Input***	***Output* yang diharapkan**	**Status**
2.	Masuk menu lanjut dengan nama *user* yang dipilih	Pilih Nama *User*	Tampil nama *User* yang dipilih pada halaman menu utama	Sukses

Setelah dilakukan pembuktian pada tabel 4.2 ternyata hasilnya adalah sukses, *user* dapat masuk ke dalam menu utama dan *user* yang masuk sesuai dengan *user* yang dipilih. Berikut pembuktiannya pada Gambar 4.47.

Tampilan *list user* yang tersimpan

Tampilan Menu Utama

Gambar 4.47 Pembuktian Test Case ID 2 Fungsi *User Login*

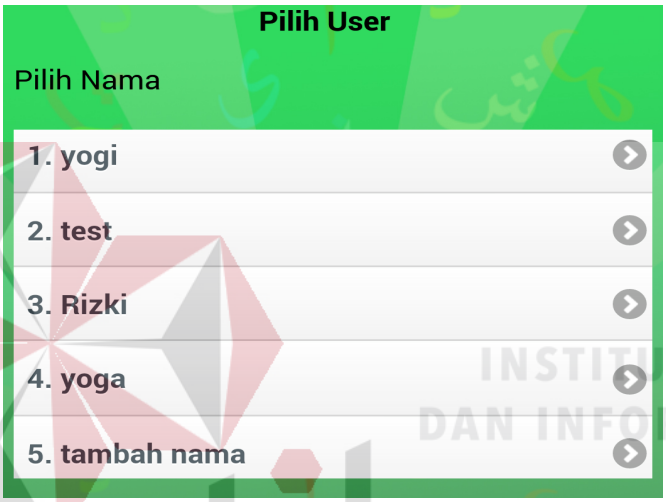
Pada Gambar 4.47 adalah pembuktian untuk masuk ke menu utama. Pertama *user* harus mempunyai idlogin pada menu lanjut. Setelah *user* memilih nama, maka aplikasi akan mengarah ke menu utama.

Berikutnya adalah uji menampilkan nama yang sudah di buat sebelumnya. *User* yang telah dibuat, di uji apakah masih ada di dalam daftar *user* yaitu pada menu lanjut, atau tidak tampil pada menu lanjut data *user* yang tersimpan oleh sistem. Berikut adalah daftar *user* yang sudah dibuat sebelumnya pada tabel 4.3.

Tabel 4.3 Keterangan *Test Case ID* 3 Fungsi *List User*

Test Case ID	Tujuan	*Input*	*Output* yang diharapkan	Status
3.	Menampilkan nama *user* yang tersimpan	Masuk menu lanjut	Tampil nama *user* yang sudah ada.	Sukses

Hasil uji coba pada tabel 4.3 adalah sukses. Nama atau daftar *user* yang telah dibuat sebelumnya tetap masuk ke dalam menu lanjut. Berikut pembuktian pada Gambar 4.48.



Gambar 4.48 Pembuktian Test Case ID 3 *List User*

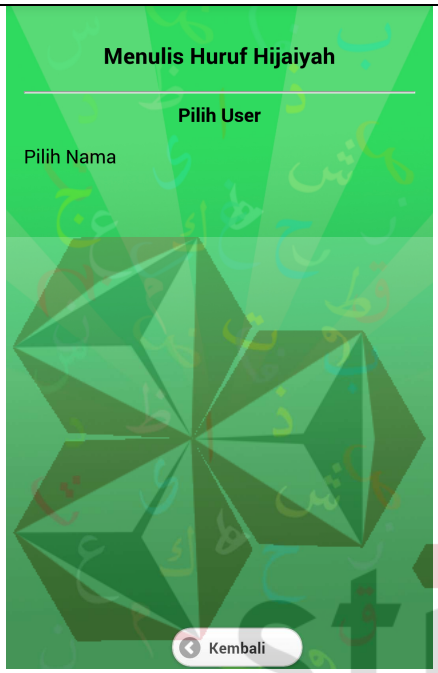
Pada Gambar 4.48 adalah pembuktian bahwa *user* yang pernah membuat akan ditampilkan pada menu lanjut.

Pada tabel 4.4 adalah uji coba ketika aplikasi tidak ada *user login.* Di dalam menu lanjut tidak ada nama *user* sama sekali, seperti pada tabel 4.4.

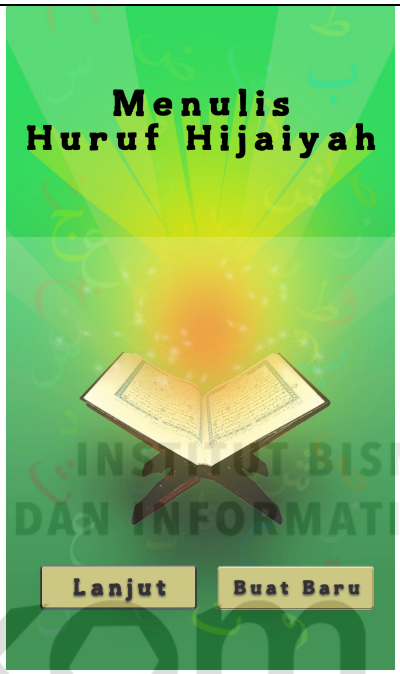
Tabel 4.4 Keterangan *Test Case ID* 4 Fungsi Menu Utama

Test Case ID	Tujuan	*Input*	*Output* yang diharapkan	Status
4.	Masuk aplikasi tanpa ada *user login*	Pilih menu lanjut *user*.	Tidak dapat masuk ke dalam menu utama aplikasi.	Sukses

Hasil coba pada tabel 4.4 adalah sukses. *User* tidak dapat masuk ke dalam menu utama karena tidak ada *user login, user* akan selalu berada dalam menu pembuka, karena di dalam menu lanjut hanya ada tombol kembali nantinya akan ke menu pembuka. Sehingga *user* diharuskan membuat *user login*. Berikut pembuktiannya seperti terlihat pada Gambar 4.49.



Tampilan *List User* apabila tidak ada yang tersimpan



Menu yang ada pada tampilan awal aplikasi

Gambar 4.49 Pembuktian Test Case ID 4 Fungsi Menu Utama

Pada Gambar 4.49 adalah uji coba, ketika *user* tidak mempunyai idlogin, maka *user* tidak bisa masuk ke menu utama, sehingga *user* akan ditampilkan pada menu pembuka pada aplikasi.

B. Hasil Uji Coba Pengenalan Huruf Arab

Pada menu pengenalan huruf hijaiyah terdapat tingkatan/level pengenalan huruf hijaiyah dan juga terdapat fitur untuk mengetahui cara penulisan dan pelafalan huruf hijaiyah tersebut. Uji coba pengenalan huruf hijaiyah ini bertujuan

untuk mengetahui apakah fungsi pengenalan huruf hijaiyah berjalan dengan baik, khususnya dalam uji coba *touchevent*, dalam penggambaran pola huruf hijaiyah sesuai dengan kaidah. Desain uji coba pengenalan huruf arab bisa dilihat pada tabel 4.5.

Tabel 4.5 Keterangan *Test Case ID* 5 Fungsi Huruf Arab

Test Case ID	**Tujuan**	***Input***	***Output* yang diharapkan**	**Status**
5.	Mengecek kesesuaian pilihan *user* memilih huruf arab.	Sentuhan jari pada salah satu pilihan dari 28 huruf arab .	Muncul tata cara menulis huruf arab sesuai dengan pilihan *user*.	Sukses

Pada hasil coba pada tabel 4.5 adalah sukses. Kesesuaian pilihan *user* memilih huruf arab, maka yang muncul adalah tata cara penulisan huruf yang dipilih oleh *user*, dicontohkan *user* memilih huruf "baa", maka akan tampil tata cara penulisan huruf "baa". Berikut adalah pembuktian seperti yang terlihat pada Gambar 4.50.

Gambar 4.50 Pembuktian Test Case ID 5 Fungsi Huruf Arab

Pada Gambar 4.50 adalah uji coba pada menu pengenalan. Pada uji coba memilih huruf ba. Setelah itu *user* akan masuk ke menu pengenalan yang terdapat beberapa fitur yaitu tombol, stop dan suara.

Tabel 4.6 adalah tabel uji coba fungsi tombol play pada menu pengenalan hijaiyah. Diharapkan uji coba ini fungsi yang di uji sesuai dengan fungsi seharusnya.

Tabel 4.6 Keterangan Test Case ID 6 Fungsi Petunjuk

Test Case ID	**Tujuan**	***Input***	***Output* yang diharapkan**	**Status**
6.	Cek fungsi lihat tata cara penulisan huruf baa’	Pilih tombol *play*	Muncul tata cara penulisan huruf arab yang dipilih *user* dengan gambar bergerak.	Sukses

Hasil uji coba pada tabel 4.6 adalah sukses. Fungsi tombol *play* pada menu pengenalan hijaiyah dapat berjalan sesuai dengan fungsinya yaitu dapat menampilkan tata cara penulisan huruf arab dengan gambar bergerak. Dicontohkan *user play* pada huruf “baa”, maka tampil gambar gerak huruf “baa”. Berikut adalah pembuktian dari uji coba seperti pada Gambar 4.51.

Gambar 4.51 Pembuktian Test Case ID 6 Fungsi Petunjuk

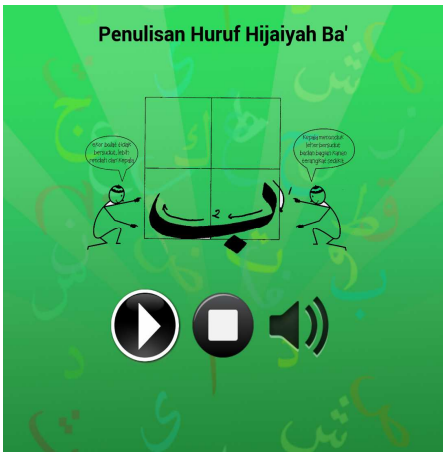
Pada Gambar 4.51 adalah uji coba fitur *play* pada menu pengenalan huruf arab. *User* dapat melihat gambar gerak tentang tata cara penulisan huruf arab. Pada uji coba ini, dicontohkan huruf ba, jadi akan muncuk tata cara penulisan huruf ba.

Tabel 4.7 adalah uji coba fungsi tombol *stop* ketika dalam kondisi gambar bergerak berjalan. Diharapkan tombol *stop* berfungsi menghentikan gambar bergerak (kembali ke posisi awal) gambar tata cara penulisan huruf arab dengan gambar tidak bergerak.

Tabel 4.7 Keterangan Test Case ID 7 Fungsi Petunjuk

Test Case ID	**Tujuan**	***Input***	***Output* yang diharapkan**	**Status**
7.	Cek fungsi lihat tata cara penulisan tanpa gambar gerak	Pilih tombol play kemudian pilih tombol stop.	Muncul tata cara penulisan huruf arab berupa gambar tapi tidak bergerak.	Sukses

Hasil coba dari tabel 4.7 adalah sukses. Tombol *stop* dapat berfungsi menghentikan gambar bergerak dan kembali ke posisi awal berupa gambar dengan petunjuk. Berikut adalah pembuktian seperti pada Gambar 4.52.



Gambar 4.52 Pembuktian Test Case ID 7 Fungsi Petunjuk

Pada Gambar 4.52 adalah uji coba tombol *stop*. Apabila sebelumnya *user* melihat gambar gerak, tombol *stop* berfungsi untuk menampilkan gambar ba dengan garis dan panah petunjuk penulisan huruf arab. Pada uji coba ini, menggunakan huruf ba. Tombol suara adalah untuk mendengarkan pelafalan dari huruf ba.

C. Hasil Uji Coba Latihan Menulis Huruf Arab

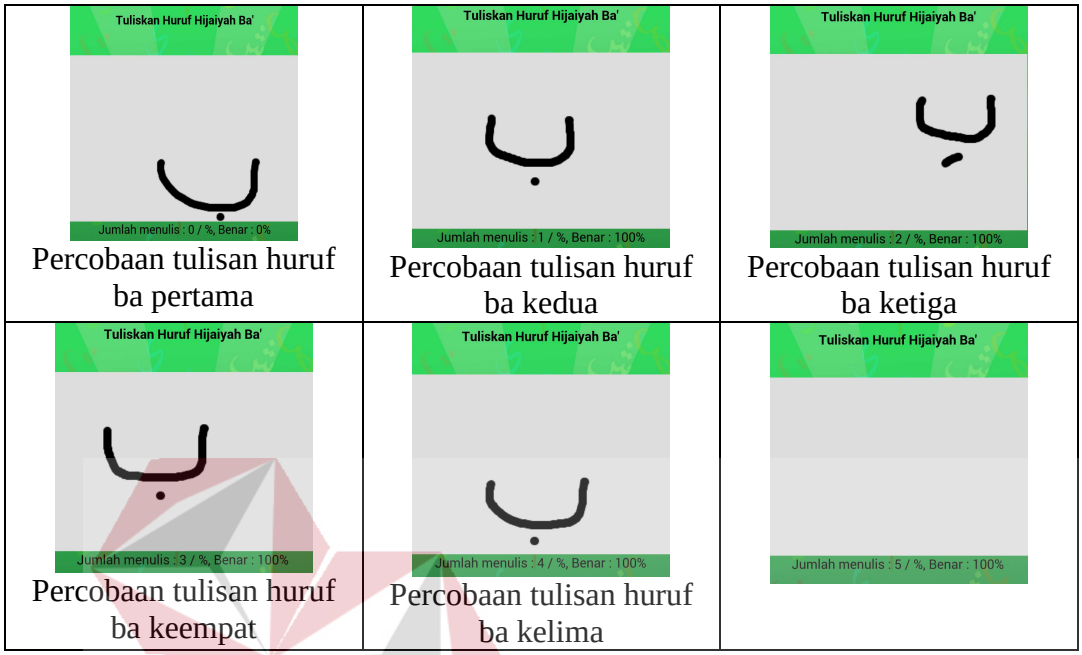
Setiap *user* yang akan melakukan uji coba pada menu latihan menulis hijaiyah sesuai dengan huruf yang dipilih oleh *user*. Uji coba latihan menulis huruf arab ini bertujuan untuk mengetahui apakah fungsi ini dapat berjalan dengan baik agar nantinya bila aplikasi sampai pada tangan *user* semua fungsi dapat berjalan sesuai harapan, yaitu dapat mengecek benar atau salah hasil *input user*. Berikut pada tabel 4.8 desain uji coba menulis huruf hijaiyah.

Tabel 4.8 Keterangan Test Case ID 8 Fungsi Latihan Menulis

Test Case ID	**Tujuan**	***Input***	***Output* yang diharapkan**	**Status**
8.	Cek fungsi latihan menulis huruf arab bila benar.	Masuk pada menu latihan menulis hijaiyah, lalau memasukkan tulisan huruf arab/ *gestures* dengan sengaja benar, sebanyak lima kali.	Prosentase benar 100% *input* dari *user*.	Sukses

Hasil uji coba dari tabel 4.8 adalah sukses. Dalam uji coba ini disengaja menulis huruf arab dengan benar. Dari jumlah menulis huruf arab yang dipilih *user* sebanyak lima kali, prosentase benar adalah tetap 100%. Hal ini menunjukkan dari jumlah tulisan *user* sebanyak lima kali semua tulisan adalah

benar Berikut pembuktian dari uji coba yang dilakukan yang terlihat pada Gambar 4.53.



Gambar 4.53 Pembuktian Test Case ID 8 Fungsi Latihan Menulis

Pada Gambar 4.53 adalah uji coba menu latihan menulis huruf arab. pada uji coba ini menggunakan huruf ba. Setiap kali *user* menuliskan huruf, aplikasi akan memberikan info prosentase benar dan jumlah dari *user* menulis. Pada uji coba ini menggunakan huruf ba yang benar.

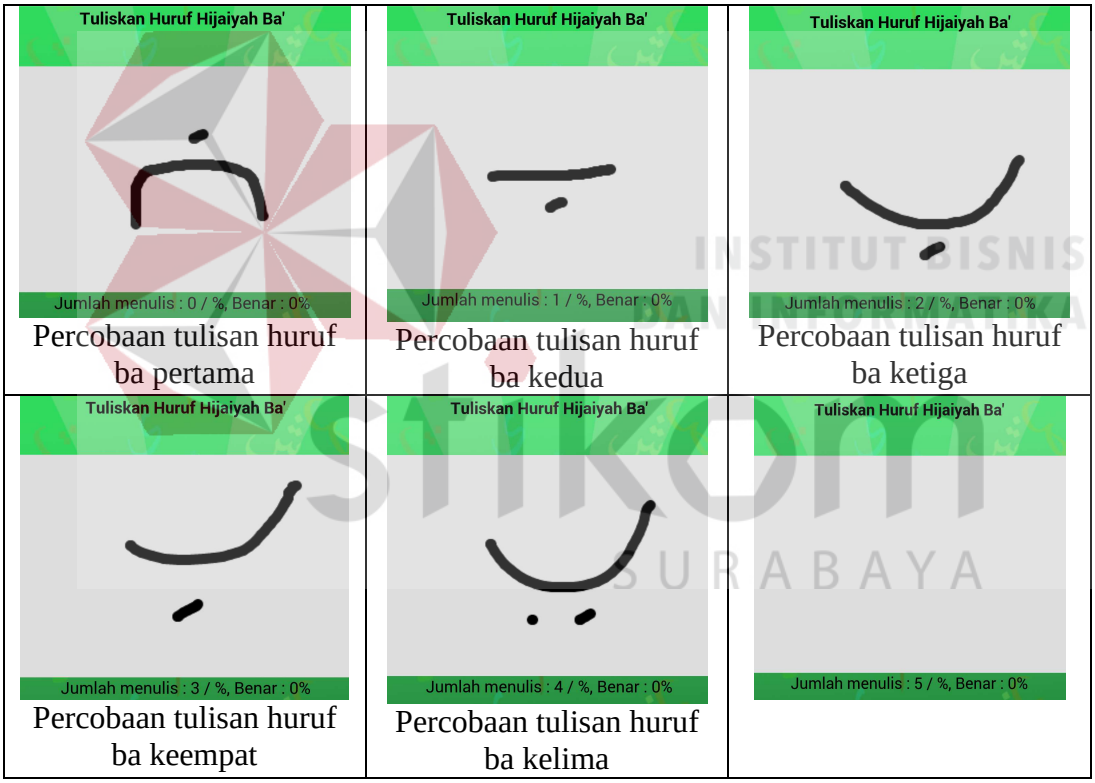
Tabel 4.9 adalah uji coba dalam latihan menulis huruf arab. Hal yang di uji coba adalah kesesuaian fungsi *gestures* pada menu latihan menulis hijaiyah. Diharapkan fungsi berjalan sesuai dengan harapan.

Tabel 4.9 Keterangan *Test Case ID* 9 Fungsi Latihan Menulis

Test Case ID	**Tujuan**	***Input***	***Output* yang diharapkan**	**Status**
9.	Cek fungsi latihan menulis huruf arab bila salah.	Memilih menu latihan menulis hijaiyah, lalu memasukkan tulisan	Prosentase benar 0% *input* dari *user*.	Sukses

Test Case ID	Tujuan	*Input*	*Output* yang diharapkan	Status
		huruf arab/ *gestures* dengan sengaja salah, sebanyak lima kali.		

Hasil uji coba pada tabel 4.9 adalah sukses. Fungsi pencocokan dalam latihan berjalan dengan cukup baik. Pada uji coba kali ini disengaja tulisan huruf arab tidak sesuai dengan huruf ba'. Hasil uji coba, dari lima kali menulis huruf ba' prosentase benar adalah 0%. Berikut pembuktian yang ada pada Gambar 4.54.



Gambar 4.54 Pembuktian Test Case ID 9 Fungsi Latihan Menulis

Pada Gambar 4.54 adalah uji coba menu latihan menulis huruf arab. pada uji coba ini menggunakan huruf ba. Setiap kali *user* menuliskan huruf, aplikasi akan memberikan info prosentase benar dan jumlah dari *user* menulis. Pada uji coba ini menggunakan huruf ba yang salah.

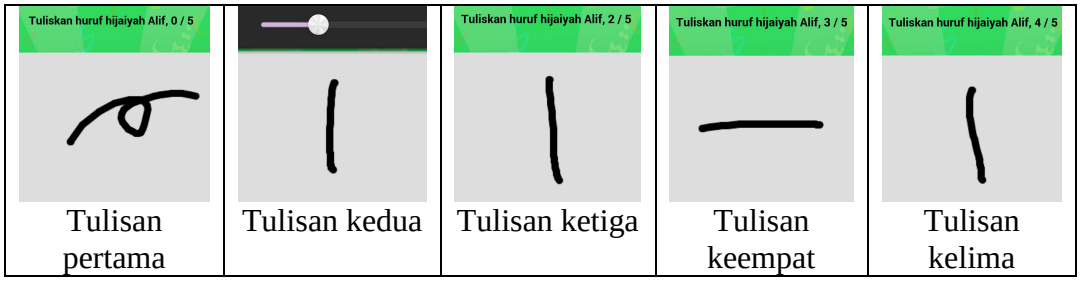
D. Hasil Uji Coba Kuis Menulis Huruf Hijaiyah/ Arab

Pada uji coba kuis menulis huruf arab adalah menguji sistem ketika *user* mencoba melakukan latihan mengerjakan soal kuis. Sistem akan memberikan info nilai ketika *user* telah menuliskan jawaban sebanyak lima kali. Berikut uji coba yang dilakukan pada tabel 4.10.

Tabel 4.10 Keterangan *Test Case ID* 10 Fungsi Kuis

Test Case ID	**Tujuan**	***Input***	***Output* yang diharapkan**	**Status**
10.	Cek fungsi kuis menulis huruf arab pada tes kemampuan sebanyak lima kali.	Memilih menu kuis soal tes kemampuan kemudian memasukkan tulisan huruf arab/ *gestures* sesuai dengan pilihan *user*, sebanyak lima kali.	*User* dapat menuliskan huruf pada *canvas* sebanyak lima kali.	Sukses

Hasil uji coba yang dilakukan pada tabel 4.10. adalah sukses. Setelah memasukkan tulisan *gestures* huruf arab sebanyak llima kali, pada akhir soal keluar informasi nilai. Pada Gambar 4.55 adalah hasil pengerjaan *user* melakukan kuis tes kemampuan.



Tulisan pertama	Tulisan kedua	Tulisan ketiga	Tulisan keempat	Tulisan kelima

Gambar 4.55 Pembuktian Test Case ID 10 Fungsi Kuis

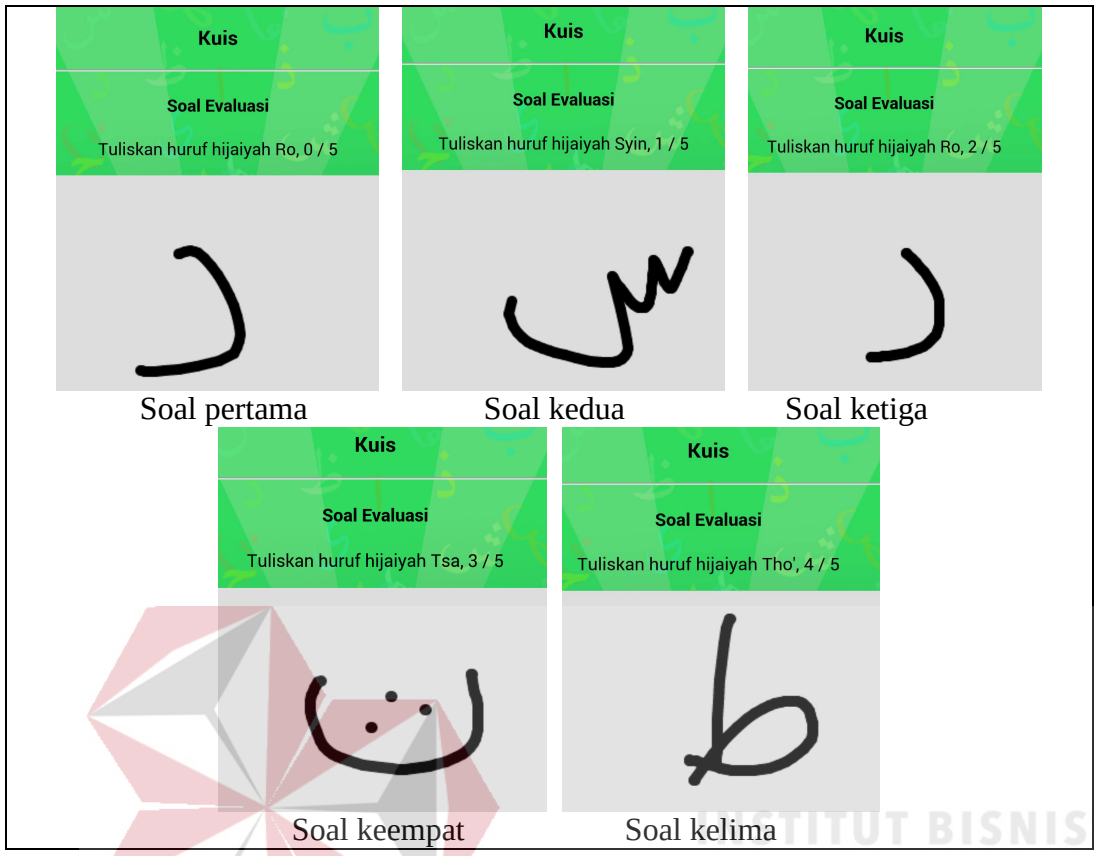
Pada Gambar 4.55 adalah uji coba fungsi tes kemampuan untuk pembuktian *user* dapat menuliskan atau memasukkan jawaban sebanyak lima kali. Uji coba yang dilakukan adalah memilih huruf arab alif.

Tabel 4.11 adalah uji coba dalam latihan kuis menulis huruf arab pada menu soal evaluasi. Hal yang di uji coba adalah sistem dapat memberikan soal secara acak sebanyak lima kali kemudian *user* dapat menuliskan jawaban. Diharapkan fungsi berjalan sesuai dengan harapan.

Tabel 4.11 Keterangan *Test Case ID* 11 Fungsi Kuis

Test Case ID	**Tujuan**	***Input***	***Output* yang diharapkan**	**Status**
11.	Cek fungsi kuis menulis huruf arab pada soal evaluasi sebanyak lima kali.	Memilih menu kuis soal evaluasi kemudian memasukkan tulisan huruf arab/ *gestures* sesuai dengan soal yang diberikan oleh sistem, sebanyak lima kali.	Sistem dapat memberikan soal secara acak kemudian *user* dapat menuliskan huruf pada *canvas* sebanyak lima kali.	Sukses

Hasil uji coba yang dilakukan pada tabel 4.11. adalah sukses. Setelah masuk pada menu kuis soal evaluasi, *user* kemudian memasukkan tulisan *gestures* huruf arab sebanyak lima kali. Pada Gambar 4.56 adalah hasil pengerjaan *user* melakukan kuis soal evaluasi.



Gambar 4.56 Pembuktian Test Case ID 11 Fungsi Kuis

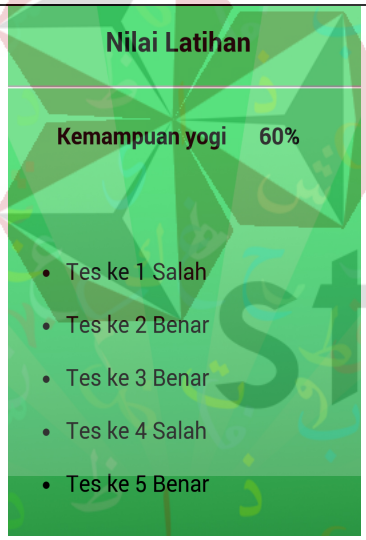
Pada Gambar 4.54 adalah uji coba terhadap fungsi menu kuis soal evaluasi. *User* dapat memasukkan jawaban sebanyak lima kali dan soal dapat dari sistem secara acak. Uji coba ini berfungsi untuk melatih *user* dalam mengingat bentuk 28 huruf arab, karena sistem memberikan soal secara acak.

Tabel 4.12 adalah uji coba dalam memberikan nilai pada menu kuis tes kemampuan maupun soal evaluasi. Hal yang di uji coba adalah sistem dapat memberikan info nilai setelah *user* menulis jawaban sebanyak lima kali, karena pada sistem terdapat proses untuk menghitung nilai atau kalkulasi nilai dari *input gestures user*. Diharapkan fungsi berjalan sesuai dengan harapan.

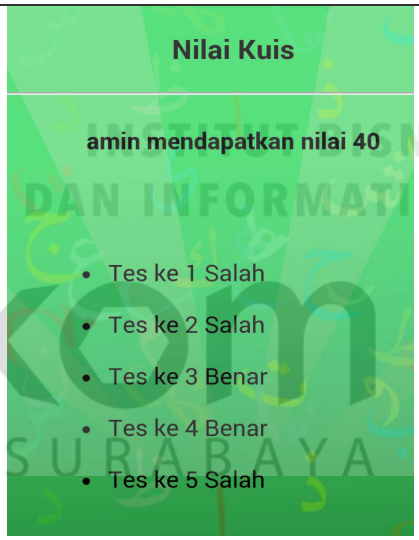
Tabel 4.12 Keterangan *Test Case ID* 12 Fungsi Nilai Kuis

Test Case ID	Tujuan	*Input*	*Output* yang diharapkan	Status
12.	Cek hasil nilai setelah *user* melakukan kuis.	Memasukkan tulisan huruf arab pada soal kuis.	Nilai dari latihan *user* dalam mengerjakan soal kuis	Sukses

Hasil uji coba yang dilakukan pada tabel 4.12. adalah sukses. Setelah *user* menuliskan jawaban pada soal kuis sebanyak lima kali, sistem memberikan info nilai dari hasil pengerjaan kuis. Pada Gambar 4.57 adalah info nilai pengerjaan *user* melakukan kuis.



Info Nilai dari Tes Kemampuan



Info Nilai dari Soal Evaluasi

Gambar 4.57 Pembuktian Test Case ID 12 Fungsi Nilai Kuis

Pada Gambar 4.57 adalah uji coba dari hasil kalkulasi nilai latihan kuis. Ketika *user* melakukan latihan kuis tes kemampuan atau soal evaluasi, sistem akan memberikan info berupa nilai. Uji coba ini dilakukan agar *user* tahu hasil dari latihan yang dilakukan dan nantinya sebagai bahan pembetulan dalam menulis huruf arab.

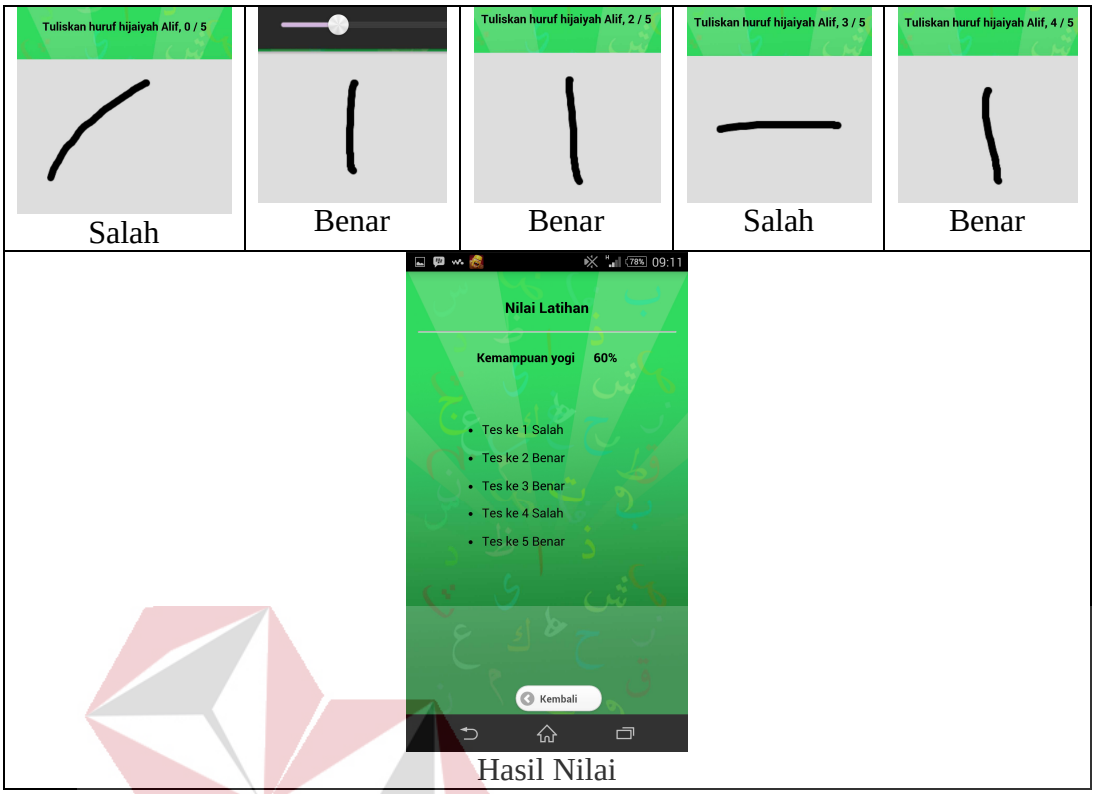
E. Hasil Uji Coba Penilaian

Pada uji coba penilaian, adalah menguji sistem dalam menilai masukan/ *input* dari *user*. Ketika *user* melakukan latihan di kuis, sistem mencocokkan dan memberikan total nilai dari hasil pengerjaan, setelah *user* memasukkan tulisan jawaban sebanyak lima kali. Berikut uji coba yang dilakukan seperti pada tabel 4.13.

Tabel 4.13 Keterangan *Test Case ID* 13 Fungsi Nilai

Test Case ID	**Tujuan**	***Input***	***Output* yang diharapkan**	**Status**
13.	Cek fungsi penilaian pada kuis tes kemampuan	Memasukkan tulisan huruf arab/ *gestures* huruf alif secara acak benar/ salah.	Muncul kalkulasi nilai yang telah dilakukan di kuis	Sukses

Hasil uji coba yang dilakukan pada tabel 4.13. adalah sukses. Setelah memasukkan tulisan *gestures* huruf arab sebanyak lima kali, pada akhir soal keluar info nilai dari hasil pengerjaan. Pada aplikasi kuis terdapat dua macam yaitu kuis tes kemampuan dan kuis soal evaluasi. Kedua kuis mempunyai beberapa fungsi yang sama dalam menampikan nilai, yaitu proses nilai dilakukan setelah *user* melakukan tulisan jawaban lima kali. Di contohkan pada uji coba adalah huruf “alif”. *User* memasukkan huruf “alif” sebanyak lima kali, kemudian sistem melakukan proses perhitungan atau kalkulasi nilai benar dan salah. Kemudian sistem memberikan hasil atau info total nilai hasil pengerjaan user melakukan latihan kuis. Untuk jelasnya seperti yang terlihat pada Gambar 4.58.



Gambar 4.58 Pembuktian Test Case ID 13 Fungsi Nilai

Pada tabel 4.14 ada uji coba kesesuain nilai yang disimpan. Diharapkan nilai yang disimpan sesuai dengan hasil nilai yang telah dilakukan *user* ketika mengerjakan soal kuis tes kemampuan.

Tabel 4.14 Keterangan *Test Case ID* 14 Funsgi Nilai

Test Case ID	**Tujuan**	***Input***	***Output* yang diharapkan**	**Status**
14.	Cek fungsi kesesuaian nilai yang tersimpan	Memilih Menu nilai	Muncul kalkulasi nilai tes kemampuan dan hasilnya sama dengan nilai yang telah dilakukan.	Sukses

Hasil uji coba yang dilakukan pada tabel 4.14 adalah sukses. Hasil nilai yang dilakukan pada kuis tes kemampuan sama dengan nilai yang tersimpan pada fitur nilai. Berikut pembuktian pada Gambar 4.59.

Hasil Nilai ketika Kuis Tes Kemampuan

Daftar nilai yang ada pada fitur nilai

Gambar 4.59 Pembuktian Test Case ID 14 Fungsi Nilai

Tabel 4.15 adalah uji coba kesesuan terhadap *gestures* yang tersimpan pada fitur nilai. Hal ini dilakukan agar hasilnya *valid* dengan apa yang dilakukan oleh *user*.

Tabel 4.15 Keterangan *Test Case ID* 15 Fungsi Nilai

Test Case ID	**Tujuan**	***Input***	***Output* yang diharapkan**	**Status**
15.	Cek fungsi kesesuaian nilai *gestures* yang tersimpan	Memilih Menu nilai	*Gestures* yang tersimpan pada fitur nilai sesuai dengan *input gestures* yang dilakukan *user* ketika melakukan kuis tes kemampuan.	Sukses

Hasil coba yang dilakukan pada tabel 4.15 adalah sukses. *Gestures* yang tersimpan pada fitur nilai sesuai dengan *input gestures* yang dilakukan oleh *user* pada waktu melakukan kuis tes kemampuan. Berikut pembuktian seperti pada Gambar 4.60.

Gambar 4.60 Pembuktian Test Case ID 15 Fungsi Nilai

Pada Gambar 4.60 terdapat dua macam gambar (dua baris) atas dan bawah. Pada gambar baris pertama adalah hasil *screenshoot* ketika *user* memasukkan *gestures* pada kuis tes kemampuan. Pada baris yang paling bawah, baris kedua adalah *gestures* yang tersimpan pada fitur nilai. Ini adalah pembuktian bahwa hasil tulisan *user* dengan hasil tulisan yang ditampilkan di fitur nilai sama. Hal ini berguna untuk melihat histori tulisan, bila terdapat tulisan yang salah maka bisa menjadi acuan pebaikan. Uji yang digunakan adalah huruf alif.

4.4.2 Uji Coba Fungsi Pencocokan

Pada proses uji coba ini, panelis mencoba tes aplikasi dalam hal ketepatan pencocokan *gestures* huruf arab. Uji coba ini dilakukan dengan cara mencoba sebanyak sepuluh kali dari masing-masing 28 huruf arab.

Pada Tabel 4.16 adalah uji coba pencocokan 28 huruf arab sebanyak sepuluh kali dan terdapat nilai prosentase benar dan salah dari tiap masing-masing 28 huruf arab. Prosentase benar adalah ketika bentuk pola benar kemudian pada sistem juga memberikan nilai benar. Pada prosentase salah adalah ketika pola yang ditulis benar tetapi sistem mendeteksi salah ataupun sebeliknya pola yang seharusnya salah tetapi oleh sistem memberikan nilai benar.

Tabel 4.16 Hasil Uji Coba Pencocokan 28 Huruf Arab

No	Huruf Arab	Prosentase Pencocokan	
		Benar	Salah
1.	‘Alif	100	0
2.	Baa’	90	10
3.	Taa’	90	10
4.	Tsaa’	90	10
5.	Jiim	100	0
6.	Haa’	100	0
7.	Kho	100	0
8.	Daal	90	10
9.	(Dzh)aal	90	10
10.	Raa’	90	10
11.	Zaayn	90	10
12.	Siin	60	40
13.	Shiin	60	40
14.	Saad	80	20
15.	Daad	80	20
16.	Taa’	60	40
17.	(Zh)aa’	60	40
18.	‘Ayn	80	20
19.	Ghayn	80	20
20.	Faa’	90	10
21.	Qaaf	90	10
22.	Kaaf	90	10
23.	Laam	100	0
24.	Miim	60	40
25.	Nuun	90	10
26.	Haa’	60	40
27.	Waaw’	90	10
28.	Yaa’	80	20
Jumlah		2340	460

Dari uji coba ketepatan dalam pencocokan 28 huruf arab, mempunyai nilai kebenaran atau *valid* sebesar 2340 dan kesalahan sebesar 460. Sehingga bisa dihitung prosentase keberhasilan dari 28 huruf arab adalah jumlah kebenaran yang diuji/ jumlah total kebenaran dari 28 huruf arab dikalikan 100%, jadi nilai keberhasilannya 2340/ 2800 x 100% hasilnya adalah 83,6% atau bisa dibulatkan menjadi 84%. Dapat disimpulkan bahwa aplikasi berjalan dengan baik.

4.4.3 Uji Coba Fungsi Aplikasi Terhadap Pengguna

Pada proses uji coba ini, *user* bisa melihat tata cara penulisan huruf arab sesuai dengan kaidah, kemudian *user* dapat melakukan latihan kuis dan melihat histori nilai dari hasil latihan mengerjakan kuis. Uji coba ini dilakukan dengan cara mencoba aplikasi menulis huruf arab langsung kepada *user*.

Pada Tabel 4.17 adalah uji coba tes kemampuan pada 10 *user*. Masing-masing melakukan latihan sebanyak lima kali. Berikut daftar *user* yang melakukan uji coba.

Tabel 4.17 Daftar *User* yang melakukan uji coba

Nama *User*	Status
Arie	Mahasiswa
Dika	Sekolah
Yuli	Sekolah
Inggit	Sekolah
Atik	Sekolah
Ismail	Sekolah
Izzah	Mahasiswa
Fadila	Kerja
Deni	Kerja
Ardi	Mahasiswa

Pada tabel 4.18 adalah hasil dari uji coba pada 10 *user* melakukan latihan tes kemampuan. Berikut adalah hasil nilai dari masing-masing 10 *user*.

Tabel 4.18 Hasil Nilai Uji Coba Tes Kemampuan

User	Huruf	Nilai				
		I	II	III	IV	V
Arie	Ba	80	60	100	60	100
Dika	Alif	80	100	100	100	100
Yuli	Kha	100	80	100	60	100
Inggit	Dal	100	80	80	80	100
Atik	Roy	100	80	100	100	100
Ismail	Wauw	80	100	100	100	100
Izzah	Dho	60	100	60	80	80
Fadila	Ya	80	80	80	100	100
Deni	Tho	60	100	100	100	100
Ardi	Dzal	60	100	100	100	100

Tabel 4.18 adalah hasil tes dari beberapa *user* dalam melakukan tes kemampuan. Hasil nilai tersebut dapat berguna untuk mengetahui perkembangan dari *user* melakukan latihan kuis dalam menulis huruf arab.

4.5. Evaluasi

Dari proses uji coba sebelumnya maka, hasil dari uji coba di rangkum dalam tahap evaluasi ini.

4.5.1. Evaluasi Fitur

Dari uji coba fitur dapat disimpulkan bahwa semua fitur berjalan dengan baik, Fitur menampilkan gambar gerak dapat berfungsi, serta fitur latihan dalam menulis huruf arab sudah bersifat dua arah karena sistem memberikan info salah dan benar tulisan dari *input user*, serta fitur nilai adalah hasil latihan *user* dalam melakukan kuis soal evaluasi atau tes kemampuan, sehingga *user* dapat melihat histori nilai. Evaluasi fitur pada sistem sudah berjalan dengan baik.

4.5.2. Evaluasi Penilaian

Dari hasil uji coba pada 10 *user* dapat diperoleh bahwa aplikasi menulis huruf arab ini dapat membantu *user* dalam melakukan latihan menulis huruf arab. *User* dapat melihat histori nilai dan hasil tulisan dari uji coba yang dilakukan, sehingga dapat memantau perkembangan dari hasil latihan menulis huruf arab.

4.5.3. Evaluasi Pencocokan Arab

Dari hasil uji coba menulis masing-masing 28 huruf arab sebanyak sepuluh kali, 84% aplikasi dapat mengenali atau mencocokkan hasil *input user* dengan baik. Sehingga dalam proses mencocokkan *gestures*, aplikasi menulis huruf arab sudah berjalan dengan baik.