

Lampiran

Lampiran 1 *Listing Program*

- **Program di *module.js***

```
// File name: Module.js
// Created: 01.12.2005
// Copyright (c) Websoft, 2006. All rights reserved.

// InitModule
function InitModule()
{
}

// ShutdownModule
function ShutdownModule()
{
}

function NoID(){
    return "20140140003";
}

function ModulID(){
    return "HTMLChat";
}

function status(){
    return "VALID";
}

function url_update()
{
    var url_updt = "http://gflm.ummy.ac.id/_insert_text.php";
    return url_updt;
}

function url_validation()
{
    var uurl = "http://gflm.ummy.ac.id/_check_mhs.php";
    //var uurl = "http://gflm.ummy.ac.id/_pass_post.php";
    return uurl;
}

function url_send(){
    var uuurl="http://gflm.ummy.ac.id/_send_onlinetext.php";
    return uuurl;
}

function url_data(){
    var surl="http://gflm.ummy.ac.id/_data_mhs.php";
}
```

```

    return surl;
}

function callModuleID(){
    var id = "HTMLChat";
    return id;
}

var codeID = 68;
var moduleID = 'HTMLChat';

function read_file()
{
    var Scr8 = new
ActiveXObject("Scripting.FileSystemObject");
    var moduleIDs = moduleID+".txt";
    try
    {
        var CTF8 = Scr8.OpenTextFile(moduleIDs, 1,
true);
        encodedText = CTF8.ReadAll();
        CTF8.Close();
        decodedText = s_encode(encodedText,codeID);
        natureText = hexToString(decodedText);
        return natureText;
    }
    catch (err)
    {
        CTF8.Close();
        return 'Course ID is not found';
    }
}

function
create_newFile(moduleID,studentId,studentName,status)
{
    var moduleIDs = moduleID + '.txt';
    var Scr2 = new
ActiveXObject("Scripting.FileSystemObject");
    var score = g_encode(score);
    var string = generate_array(100);
    array = string.split(',');

array[0]=moduleID;array[1]=studentId;array[2]=studentName;ar
ray[3]=status;
    var Text = array_toText(array,100);
    var HText = stringToHex(Text);
    secureText = s_encode(HText,codeID);
    try
    {
        var CTF2 = Scr2.CreateTextFile(moduleIDs);

```

```

        CTF2.Write(secureText);
        CTF2.Close();
        return "OK";
    }
    catch (err)
    {
        CTF2.Close();
        alert('File saved error');
    }
}

function s_encode(str,code) {
    var encoded = "";
    for (i=0; i<str.length;i++) {
        var a = str.charCodeAt(i);
        var b = a ^ code;
        encoded = encoded+String.fromCharCode(b);
    }
    return encoded;
}

function g_encode(grade) {
    return (grade+174)*100;
}

function g_decode(higrade) {
    return ((higrade)/100 -174);
}

function generate_array(number){
    var text = "*";
    for (i=1;i<number;i++){
        text = text + ',';
        myArray[i] = '*';
        text = text + myArray[i];
    }
    return text;
}

var myArray = new Array();

function array_toText(myArray,length){
    var string = myArray[0];
    for (i=1;i<length;i++){
        string = string + "," + myArray[i];
    }
    return string;
}

function stringToHex (tmp) {
    var str = '',

```

```

        i = 0,
        tmp_len = tmp.length,
        c;

    for (; i < tmp_len; i += 1) {
        c = tmp.charCodeAt(i);
        str += d2h(c) + ' ';
    }
    return str;
}

function hexToString (tmp) {
    var arr = tmp.split(' '),
        str = '',
        i = 0,
        arr_len = arr.length,
        c;

    for (; i < arr_len; i += 1) {
        c = String.fromCharCode( h2d( arr[i] ) );
        str += c;
    }
    return str;
}

function d2h(d) {
    return d.toString(16);
}

function h2d (h) {
    return parseInt(h, 16);
}

function validation(url, json, object)
{
    var xmlhttp = readyAJAX();
    var result = null;
    if (xmlhttp != null)
    {
        ShowObject(object, "", "infinite");
        xmlhttp.open("POST", url, true);
        xmlhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
        xmlhttp.setRequestHeader("Content-length", "json.length");
        xmlhttp.setRequestHeader("Connection", "close");
        xmlhttp.send(json);
        xmlhttp.onreadystatechange = function()
        {
            if ((xmlhttp.readyState == 4) & (xmlhttp.status == 200))

```

```

    {
        var result = xmlHttp.responseText;
        if (result === "VALID")
            {
                response = updateFile();
                alert(response);
                HideObject(object, "", "infinite");
                ShowObject('OBJ_5');
                NextSlide();
            }
        if (result === "INVALID"){
            alert("Please, check your Student-ID and try
again");
            HideObject(object, "", "infinite");
        }
    }
    if ((xmlHttp.readyState == 4) & (xmlHttp.status !=
200)){
        alert("Validation error, please get internet
connection and try again");
        HideObject(object, "", "infinite");
    }
}
}
}

```

```
new XMLHttpRequest();
```

```

function updateFile()
{
    var text = read_file();
    var array = text.split(',');
    var moduleIDs = array[0] + '.txt';
    var Scr2 = new
ActiveXObject("Scripting.FileSystemObject");
    array[3]="VALID";
    var Text = array_toText(array,array.length);
    var HText = stringToHex(Text);
    secureText = s_encode(HText,codeID);
    try
    {
        var CTF2 = Scr2.CreateTextFile(moduleIDs);
        CTF2.Write(secureText);
        CTF2.Close();
        return "Validation is done";
    }
    catch (err)
    {
        CTF2.Close();
        alert('File saved error');
    }
}

```

```

}

function readyAJAX()
{
    try {
        return new XMLHttpRequest();
    } catch(e) {
        try {
            return new ActiveXObject("Msxml2.XMLHTTP");
        } catch(e) {
            try {
                return new
ActiveXObject("Microsoft.XMLHTTP");
            } catch(e) {
                return "A newer browser is needed.";
            }
        }
    }
}

function url_send(){
    var url_update =
"http://gflm.ummy.ac.id/_insert_text.php";
    return url_update;
}

function url_get(){
    var url =
"http://gflm.ummy.ac.id/_send_onlinetext.php";
    return url;
}

function url_check(){
    var url_n = "http://gflm.ummy.ac.id/_check_mhs.php";
    return url_n;
}

function nim(){
    return NIM;
}

function checkMhs(NIM){
    var url_NIM = url_check();
    var passvar= "username="+NIM;

    var url = "http://gflm.ummy.ac.id/_check_mhs.php";
    var xmlhttp = new XMLHttpRequest();
    xmlhttp.open("POST", url_NIM, true);
    xmlhttp.setRequestHeader("Content-type", "application/x-
www-form-urlencoded");
}

```

```

xmlHttpRequest.setRequestHeader("Content-length", "passvar.length
");
xmlHttpRequest.setRequestHeader("Connection", "close");
xmlHttpRequest.send(passvar);
xmlHttpRequest.onreadystatechange = function()
{
    if (xmlHttpRequest.readyState == 4)
    {
        if(xmlHttpRequest.status==200)
        {
            var resp = xmlHttpRequest.responseText;
            if(resp == "VALID"){
                alert
("Login Berhasil")

                NextSlide();}
            else { alert("NIM Tidak
Dikenal, ulangi");}
        }
        else
        {
            alert("PENYUSUP");
        }
    }
}

function sendDataPost(text)
{
    var url = url_send();
    var json = ModulID()+","+NIM+","+status() + ","+text;
    var xmlHttpRequest = readyAJAX();
    var passvar = "onlinetext="+json;
    if (xmlHttpRequest != null)
    {
        xmlHttpRequest.open("POST", url, true);
        xmlHttpRequest.setRequestHeader("Content-type","application/x-
www-form-urlencoded");
        xmlHttpRequest.setRequestHeader("Content-length",
"passvar.length");
        xmlHttpRequest.setRequestHeader("Connection","close");
        xmlHttpRequest.send(passvar);
        xmlHttpRequest.onreadystatechange = function()
        {
            if (xmlHttpRequest.readyState == 4)
            {
                if (xmlHttpRequest.status == 200)
                {
                    getStudentChat(NIM);
                }
            }
        }
    }
}

```

```

        else
        {
            alert("No Internet Connection");
        }
    }

}

}

}

function getStudentChat(NIM)
{
    var Course = ModulID();
    var htmlChat = '';
    var url = url_get();
    var passvar= "user_id=" + NIM + "&" + "course_code=" +
Course;
    var xmlhttp = readyAJAX();
    xmlhttp.open("POST", url, true);
    xmlhttp.setRequestHeader("Content-type", "application/x-
www-form-urlencoded");
    xmlhttp.setRequestHeader("Content-length", "passvar.length
");
    xmlhttp.setRequestHeader("Conection", "close");
    xmlhttp.send(passvar);
    xmlhttp.onreadystatechange = function()
    {
        if (xmlhttp.readyState == 4)
        {
            if(xmlhttp.status==200)
            {
                var text = xmlhttp.responseText;

                var array = text.split(",");
                var chat = array[3];
                g_arVars['Chat'] = chat;
                if (chat != undefined)
                {
                    htmlChat = newsDecode(chat);
                }
                //alert(chat);
                var view =
document.getElementById('Chatbox');
                view.style.background = "lightgray";
                view.style.overflow = 'scroll';
                view.innerHTML = "<HTML>"
+htmlChat+"</HTML>";
                view.scrollTop = 999999;

                // view.innerHTML = newsDecode (chat);
                // down = setTimeout(bottom, 100);

```



```

        }
        else
        {
            alert("No internet connection");
        }
    }
}

function newsCode(text){
    var buffer = "";
    var loop = "TRUE";
    while (loop){
        buffer = text.replace(',','^^');
        if (buffer == text)
        {
            break;
        }
        text = buffer;
    }
    return buffer;
}

function newsDecode(text){
    var buffer = "";
    var newText1 = "";
    var newText2 = "";
    var newText3 = "";
    var newText4 = "";
    var newText5 = "";
    var loop = "TRUE";

    while (loop){
        buffer = text.replace('$$',',');
        if (buffer == text) {break;}
        text = buffer;
    }
    newText1=text;
    var loop = "TRUE";
    while (loop){
        buffer = newText1.replace('#$', '<br>');
        if (buffer == newText1) {break;}
        newText1 = buffer;
    }
    newText2=newText1;
    var loop = "TRUE";
    while (loop){
        buffer = newText2.replace("@D$$", "<font color='red'><p
style='font-family:Verdana;', align = left><font
size='5'>");
        if (buffer == newText2) {break;}

```

```

        newText2 = buffer;
    }
    newText3=newText2;
    var loop = "TRUE";
    while (loop){
        buffer = newText3.replace("$@@", "</font>");
        if (buffer == newText3) {break;}
        newText3 = buffer;
    }
    newText4=newText3;
    var loop = "TRUE";
    while (loop){
        buffer = newText4.replace("@M$$", "<font color='blue'><p
style='font-family:Verdana;', align = right><font
size='5'>");
        if (buffer == newText4) {break;}
        newText4 = buffer;
    }
    newText5=newText4;
    var loop = "TRUE";
    while (loop){
        buffer = newText5.replace("$$@", "</font>");
        if (buffer == newText5) {break;}
        newText5 = buffer;
    }
    return newText5;
}

```

- **Program di Slide Login**

```

var text = read_file();
array = text.split(',');
if(array[3] == 'VALID')
{
    OpenSlideById('SLIDE_14');
}

```

- **Program di tombol Login**

```

var NIM = g_arVars['nim'];
var Name = g_arVars['name'];
//var PAS = g_arVars['pas'];
var url = url_update();
var moduleID = callModuleID();
var uurl = url_validation();

if (NIM !== '' )
{
    var status =
create_newFile(moduleID,NIM,Name, 'INVALID');
    if (status === 'OK')
    {
        alert('New data is saved');
    }
}

```

```

        var text = read_file();
        var array = text.split(',');
        var json = "username="+array[1];
        validation(uurl, json, 'OBJ_5');
    }
}
else
{
    alert('Enter NIM and First Name');
}
var data = read_file();
validation(url, data, "OBJ_5");

```

- **Program di *IFRAME***

```

text = read_file();
array = text.split(',');
NIM = array[1];
getStudentChat(NIM);
var view = document.getElementById('Chatbox');
view.style.background = "lightgray";
view.style.overflow = 'scroll';
view.scrollTop = 1000;

```

- **Program di tombol *Send***

```

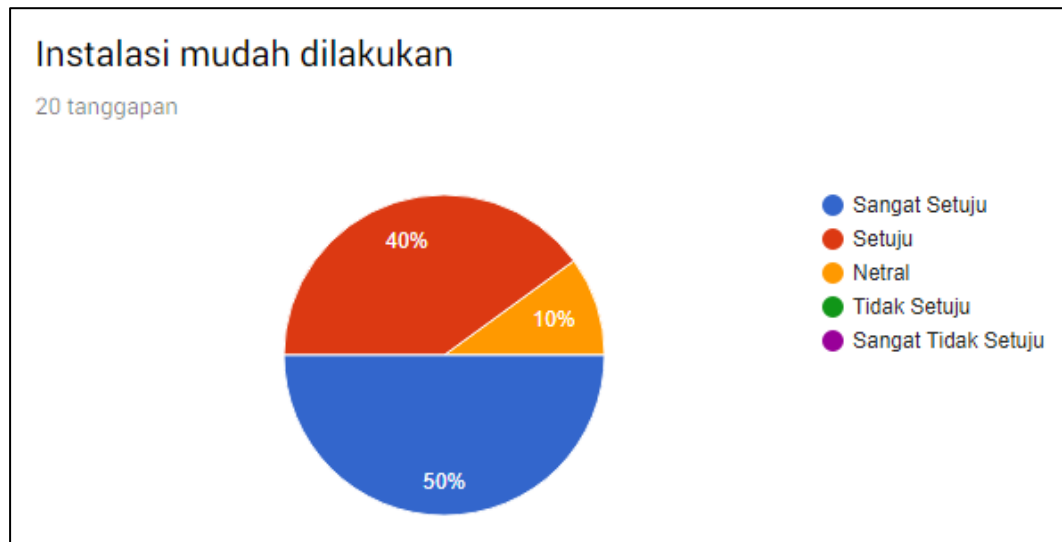
text = read_file();
array = text.split(',');
NIM = array[1];
var Chat = g_arVars['Chat'];
if (Chat == undefined)
{
    Chat="";
}
text = g_arVars['textA'];
var sendText = text.replace(", ", "$%$");

newText = Chat + "@M$$" + sendText + "#$%";
sendDataPost(newText);

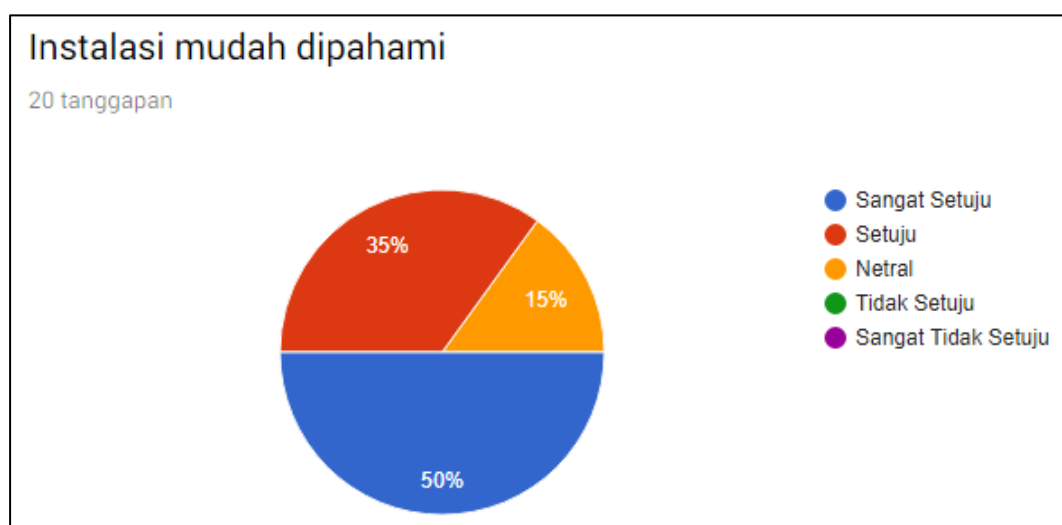
```

Lampiran 2 Hasil Kuesioer

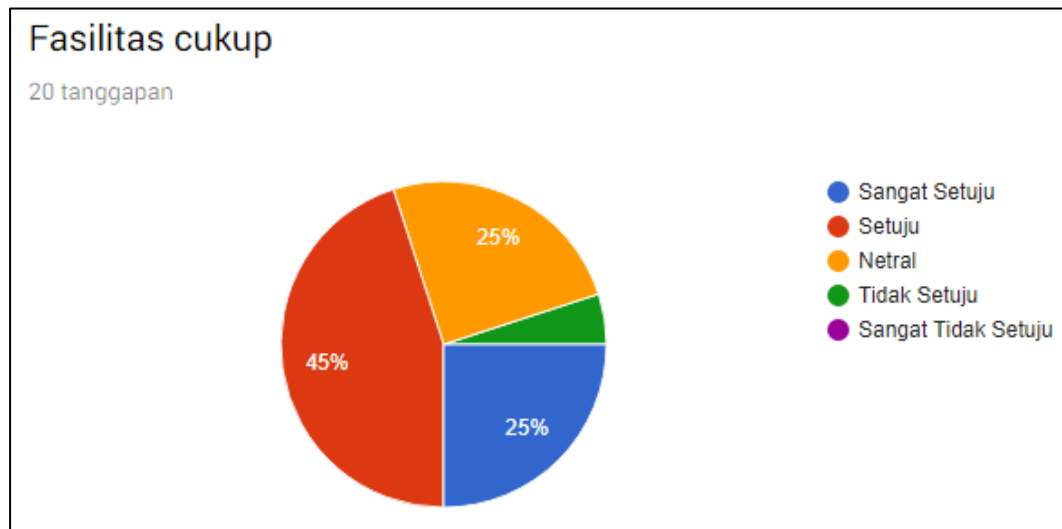
1. Instalasi mudah dilakukan



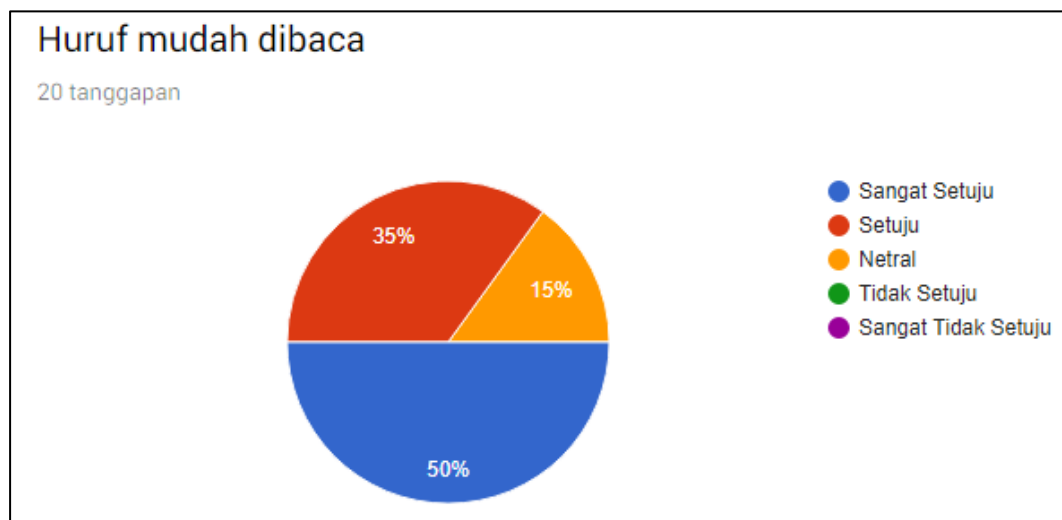
2. Instalasi mudah dipahami



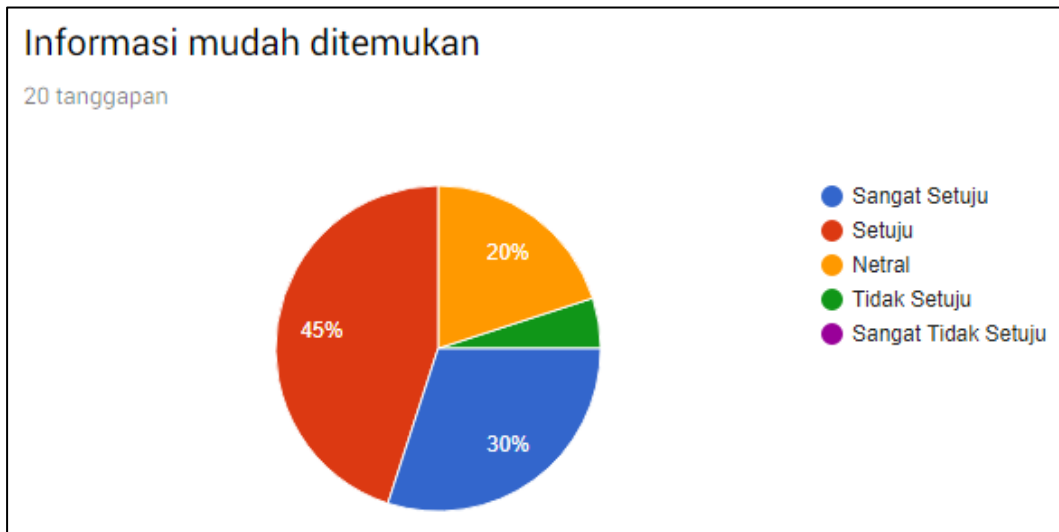
3. Fasilitas cukup



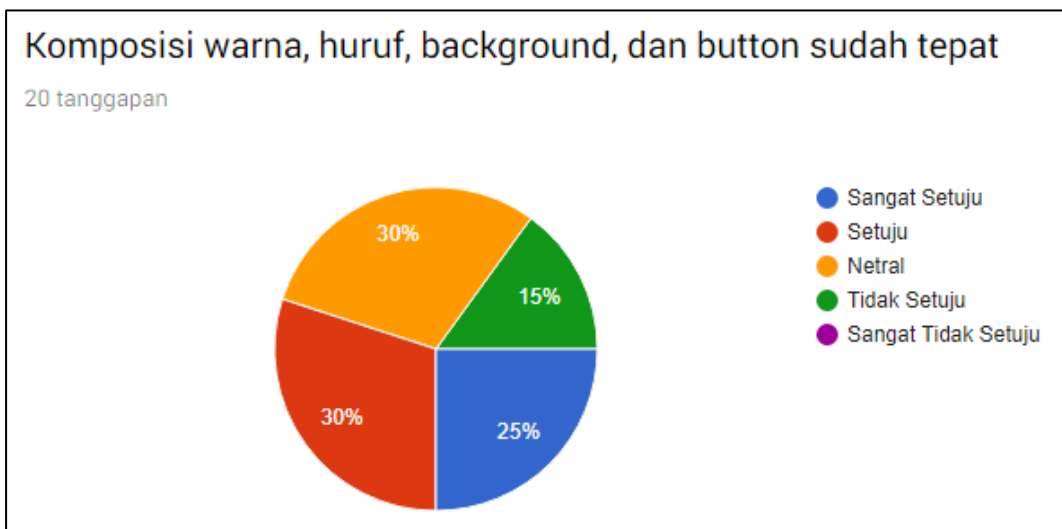
4. Huruf mudah dibaca



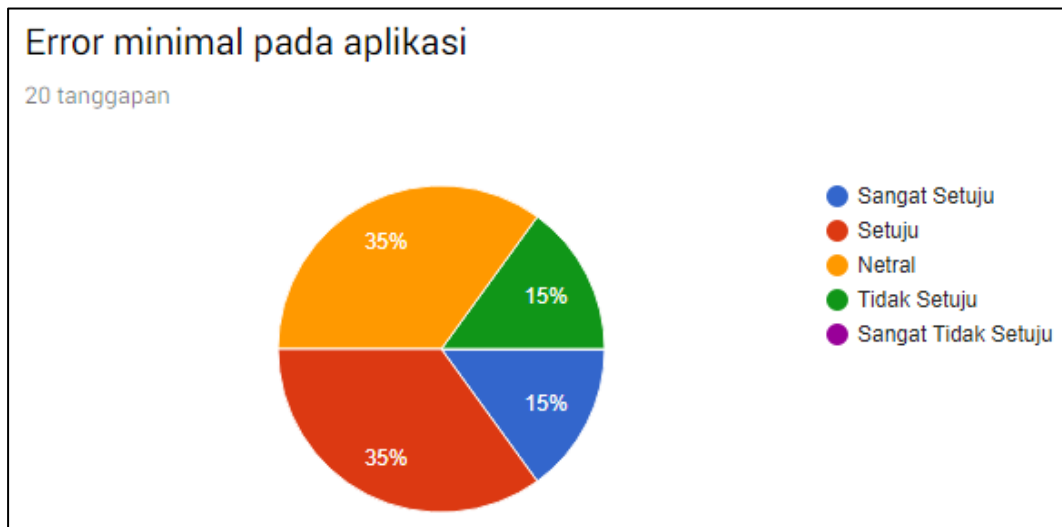
5. Informasi mudah ditemukan



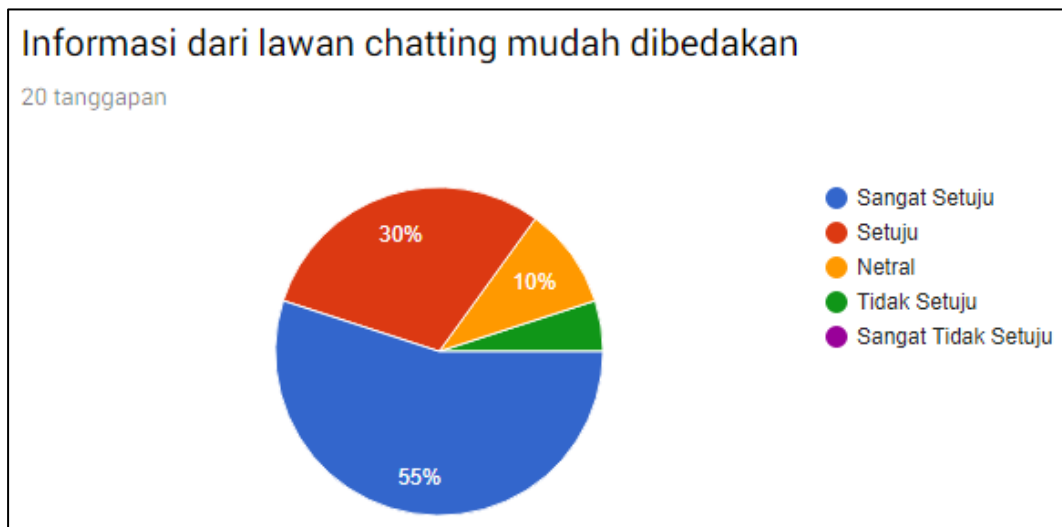
6. Komposisi warna, huruf, *background*, dan *button* sudah tepat



7. *Error* minimal pada aplikasi



8. Informasi dari lawan *chatting* mudah dibedakan

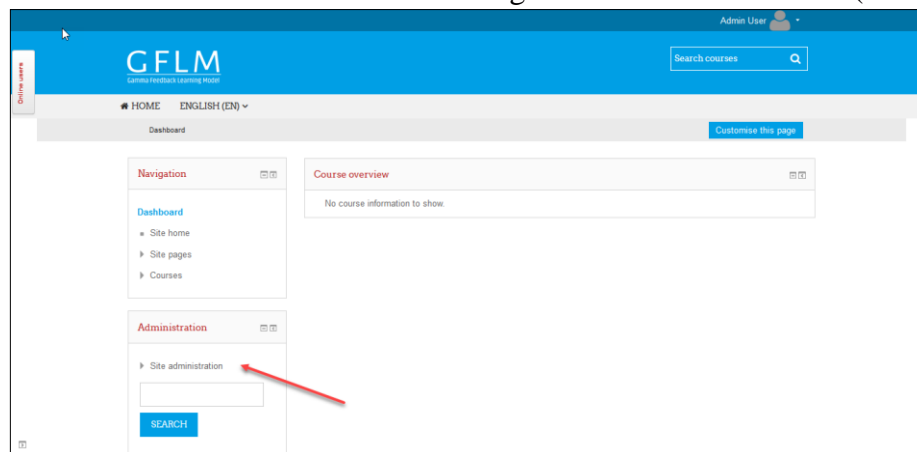


Lampiran 3 GFLM Moodle

- **Langkah pertama: Membuat *Category***

Langkah pertama adalah *Category setting*, langkah ini diperlukan untuk menyiapkan *Moodle* supaya dapat digunakan beberapa program studi.

- Login sebagai *Administrator*
- *Click* menu *Home* pada *toolbar menu*
- *Click* menu *Site Administration* di bagian *Administration Block* (Gambar 1)



Gambar 1 *Administration Block*

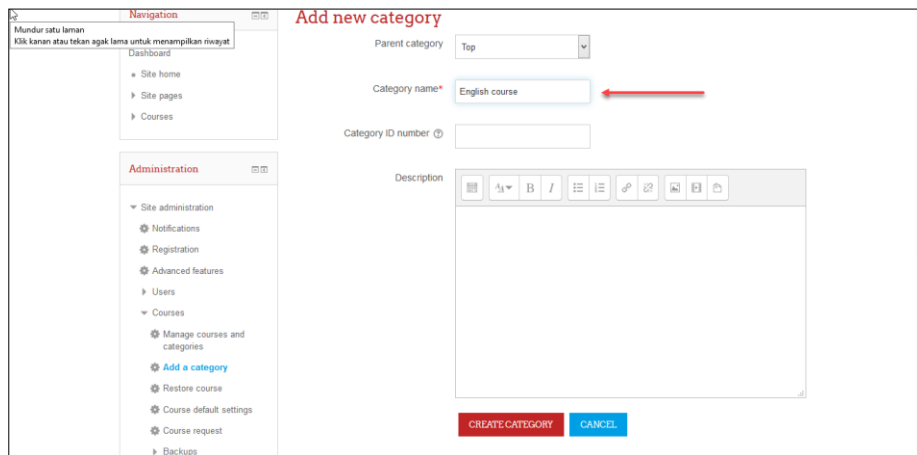
- Pilih menu *New Category*



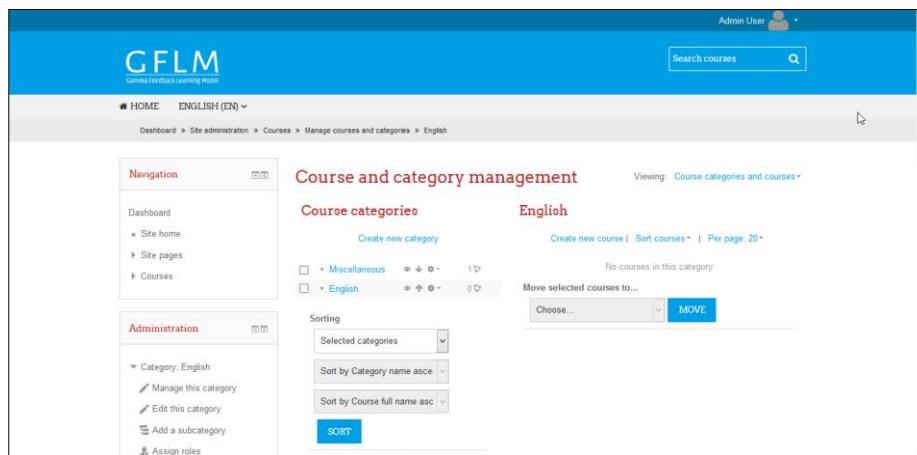
Gambar 2 Pilih menu untuk membuat *Category*

- Masukkan nama Program Studi sebagai *Category name* dan klik tombol *Create Category* di sisi bawah warna merah (Gambar 2). Jika direncanakan *Moodle* hanya untuk satu program studi, maka kategori digunakan sebagai nama blok kuliah. Misalkan, blok teori, blok praktikum, blok tugas akhir, dll.

- Tampilan penambahan *category* terlihat pada Gambar 3.



Gambar 3 Memasukkan *Category name*

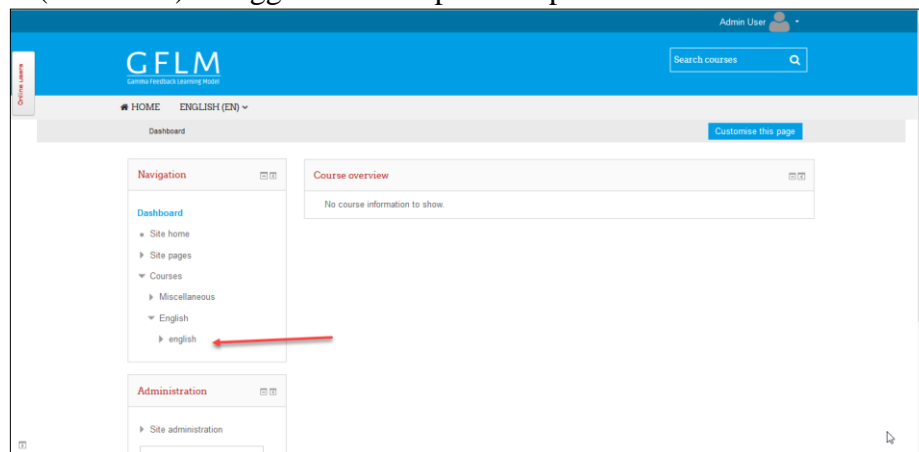


Gambar 4 Kategori *English* berhasil dibuat

Selesai membuat *Category*, dilanjutkan membuat matakuliah.

Langkah-langkah:

- Pada *Dashboard block* klik *Courses*, klik *Category*, kemudian klik *English* (Gambar 4). Hingga muncul seperti tampilan Gambar 5.



Gambar 5 Link untuk membuat matakuliah pada kategori tertentu

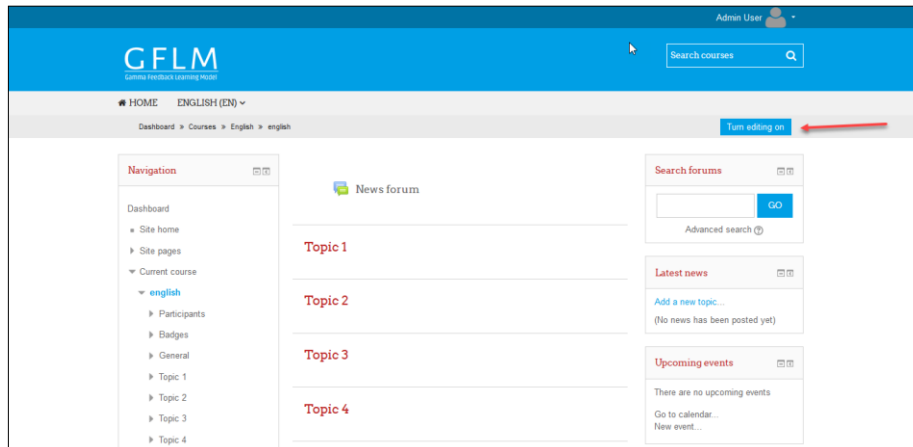
- Setelah tampil seperti Gambar 5, di *Input Form* ketik nama matakuliah, dalam contoh ini nama matakuliah adalah “*english*”, sama dengan nama kategorinya. Setelah pengisian data cukup, data disimpan dengan menekan tombol “*SAVE AND RETURN*” yang terdapat di posisi paling bawah Gambar 6.

Gambar 6 Isian untuk membuat matakuliah baru

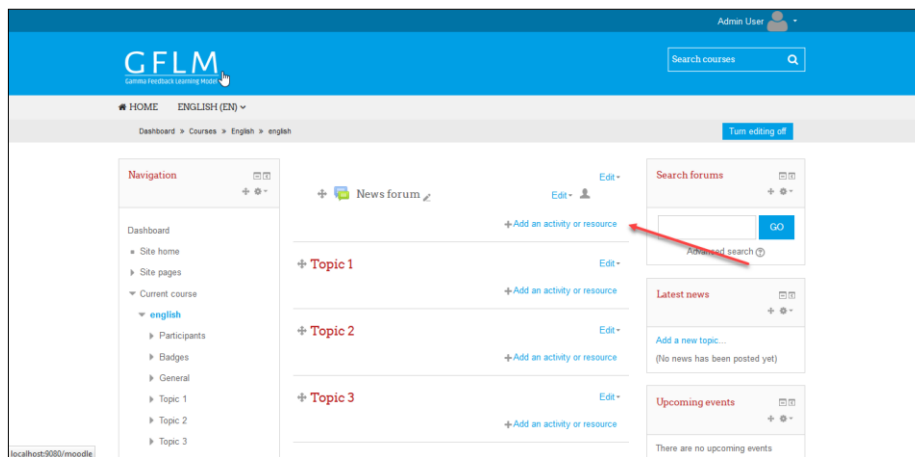
Gambar 7 Tombol *Save and Return* di posisi paling bawah

• Langkah kedua: Membuat *Assignment*

- Dimulai dari *Dashboard*, click *Site Administration*, klik *Courses*, dan klik *english* maka akan muncul tampilan seperti tampak pada Gambar 7. Tampilan ini menunjukkan sedang berada di ruang kuliah maya “*ENGLISH*”.
- Kemudian tombol *Turn edit on* (diisi atas kanan warna biru) ditekan sehingga antarmuka berubah menjadi Gambar 8. Pada layar terlihat ada tampilan *link* untuk *Add activity or resources*.



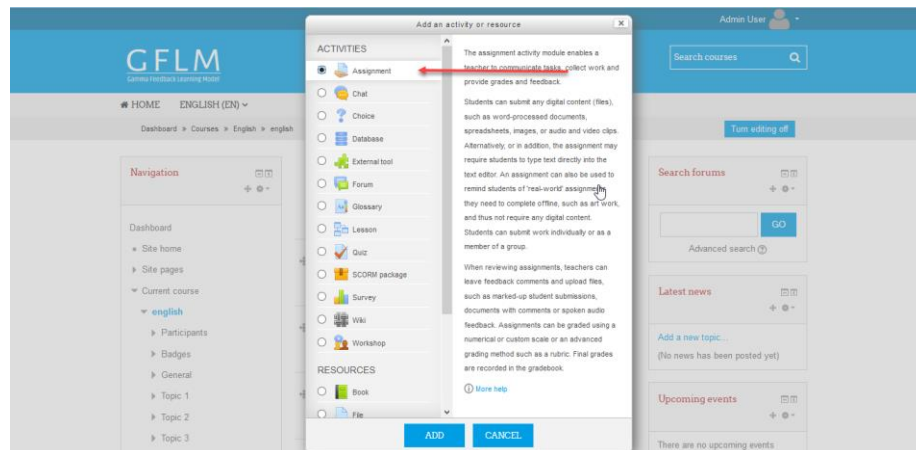
Gambar 8 Tampilan untuk membuat *Assignment*



Gambar 9 Tampilan antarmuka matakuliah pada *mode editing on*

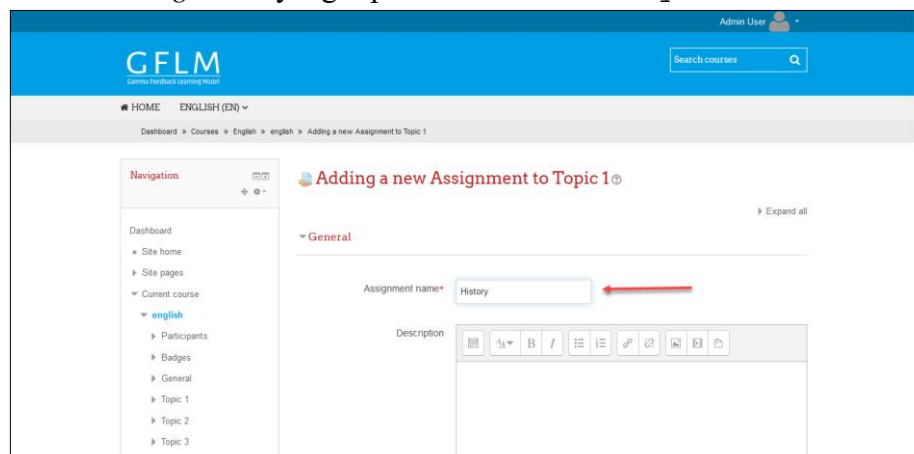
- Langkah berikutnya, menekan tombol *Add on activity or resource* sehingga muncul tampilan seperti Gambar 9.
- Kemudian klik *Assignment*.

Assignment ini dipilih karena menyesuaikan dengan data yang akan dikirim dari modul *CAI* yang berbentuk teks.



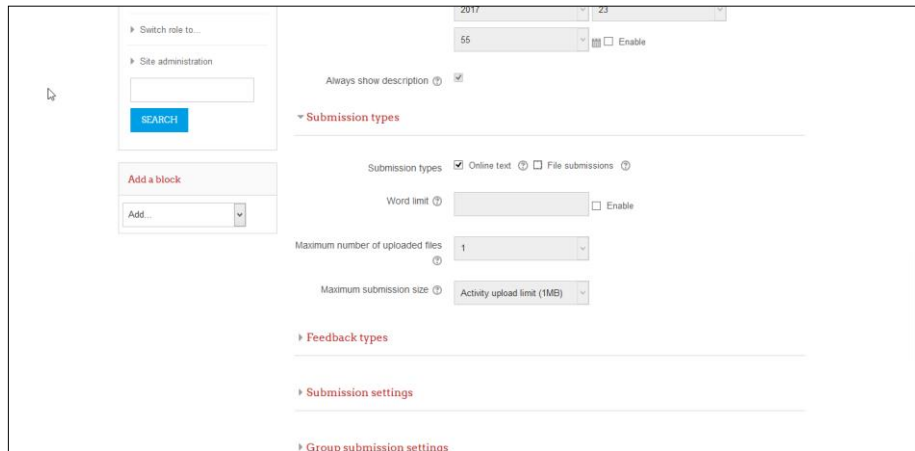
Gambar 10 Tampilan *activity* dengan pilihan *Assignment*

- Kemudian tombol *ADD* (warna biru) ditekan sehingga muncul tampilan seperti Gambar 10.
- Nama *Assignment* yang dipilih adalah “History”.



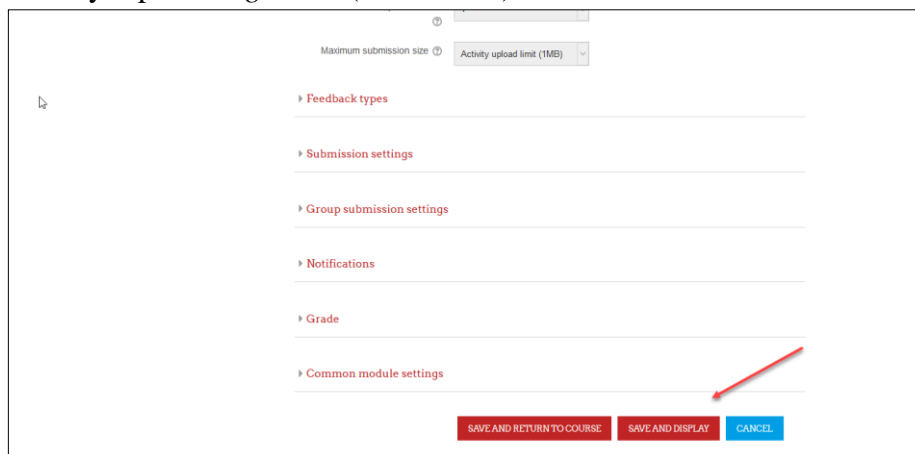
Gambar 11 Tampilan untuk memasukkan nama *Assignment*

- Pada Gambar 11 terdapat pilihan cara pengiriman *Assignment*, *online* atau *submission* (unggah *file* dokumen). Dipilih cara *online* karena data teks dikirim secara *online* yang akan diatur oleh *API*.



Gambar 12 Pilihan cara pengiriman *online* atau *upload file*

- Diisi paling bawah terdapat tombol “*SAVE AND DISPLAY*” untuk menyimpan *Assignment* (Gambar 13).



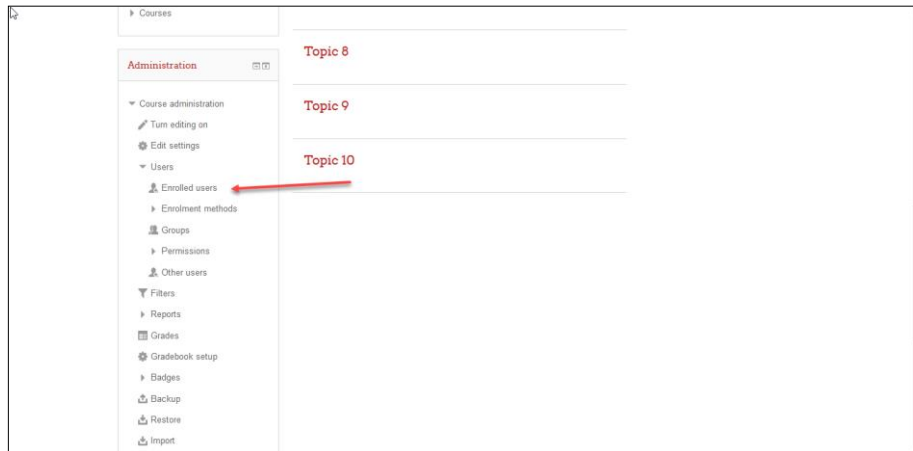
Gambar 13 Tombol untuk menyimpan *Assignment*

Langkah berikutnya adalah mendaftarkan mahasiswa ke dalam matakuliah. Tatacara *upload* data peserta matakuliah dapat dilihat pada buku panduan *LMS Moodle*.

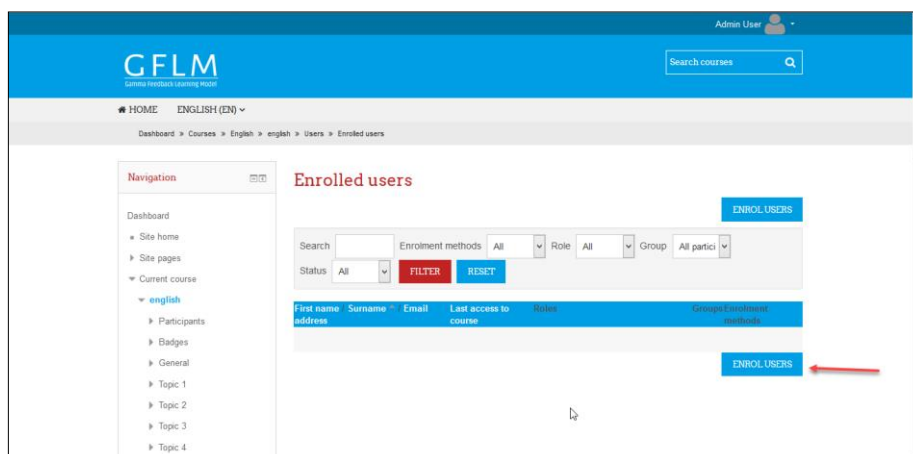
- **Langkah ketiga: *Student enrollment***

Pendaftaran mahasiswa sebagai peserta matakuliah berguna sebagai *trigger* untuk tabel *Assignment*. Langkah-langkah:

- Klik *Home*, pada *Navigation block* klik *Course*
- Jika sudah muncul *Course Administration*, klik *Users* kemudian klik *Enrolled users* (Gambar 14), sehingga tampil seperti Gambar 15.

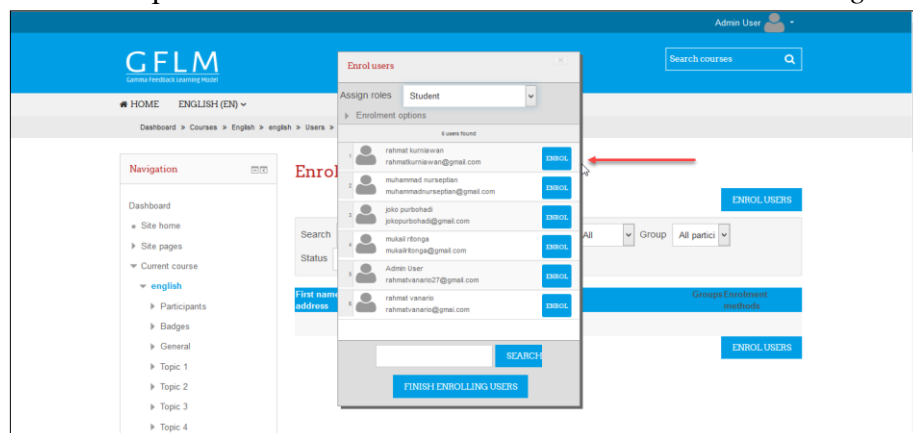


Gambar 14 Antarmuka untuk *Enrolled users*

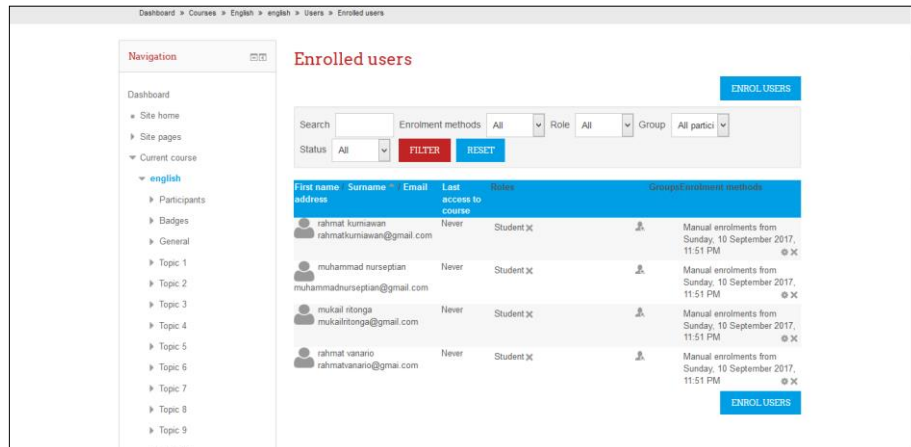


Gambar 15 Antarmuka untuk mendaftarkan mahasiswa

- Klik tombol “*Enrol user*” sehingga muncul tampilan seperti Gambar 16. Pilih mahasiswa yang ingin didaftarkan ke matakuliah “*ENGLISH*” dengan cara klik pada tombol “*Enroll*”. Kemudian klik “*Finish enrolling user*”.



Gambar 16 Mendaftarkan mahasiswa pada matakuliah



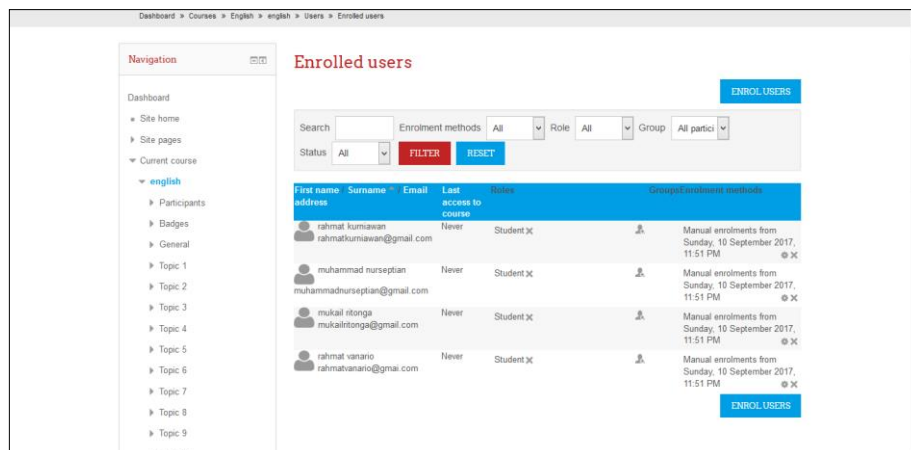
Gambar 17 Mahasiswa peserta matakuliah *English*

Gambar 17 memperlihatkan ada sembilan mahasiswa yang telah didaftarkan pada matakuliah English. Berikutnya memberi *trigger* untuk tabel *Assignment*.

- **Langkah keempat: Menginisiasi *Assignment data***

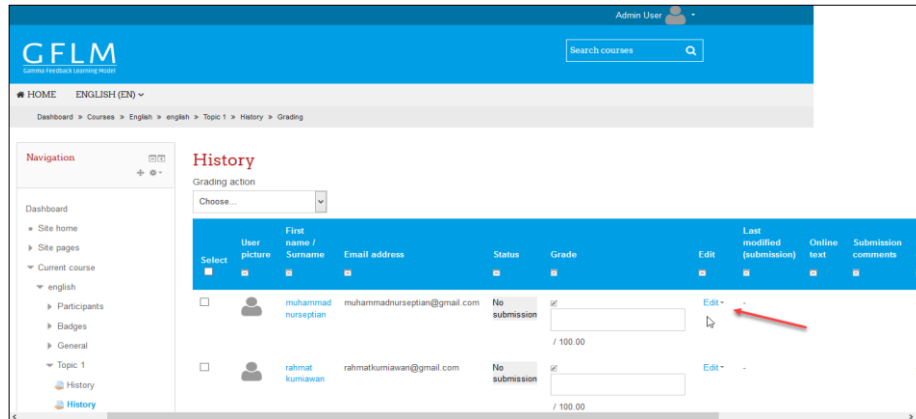
Langkah ini diperlukan untuk inisiasi nilai tugas. Seperti telah dibahas sebelumnya, bahwa *Assignment* telah ditetapkan sebagai tugas mahasiswa yang dikumpulkan secara *online*. Dalam sistem *Extended-CAI*, tugas dikirim melalui modul *CAI* yang dijalankan pada komputer atau piranti *mobile* milik mahasiswa. Tugas tersebut dikirim juga dalam bentuk teks. Namun tugas dapat dikirim setelah terlebih dahulu dilakukan proses inisiasi dengan mengisi tugas setiap mahasiswa dengan teks, minimalnya sebuah karakter (bukan spasi). Selain nilai tugas juga perlu diinisiasi dengan mengisi angka minimal 0. Langkah ini hanya dapat dilakukan oleh dosen dan *administrator*.

- Pada *Navigation block*, klik *Current Course*, klik *English*, kemudian klik *Topic 1*, dan terakhir klik *History* sehingga tampil jendela seperti Gambar 18.



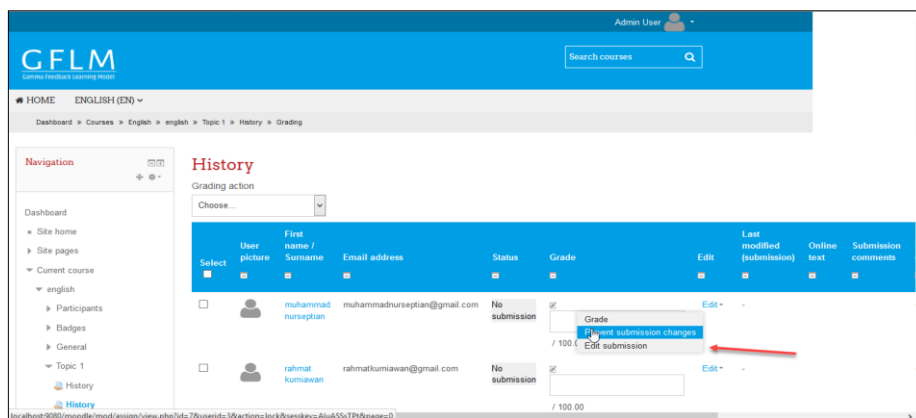
Gambar 18 Antarmuka untuk menampilkan teks terkirim CAI

- Pada antarmuka *History*, klik *link View grade all submissions* sehingga tampil *Grade action* seperti Gambar 19.



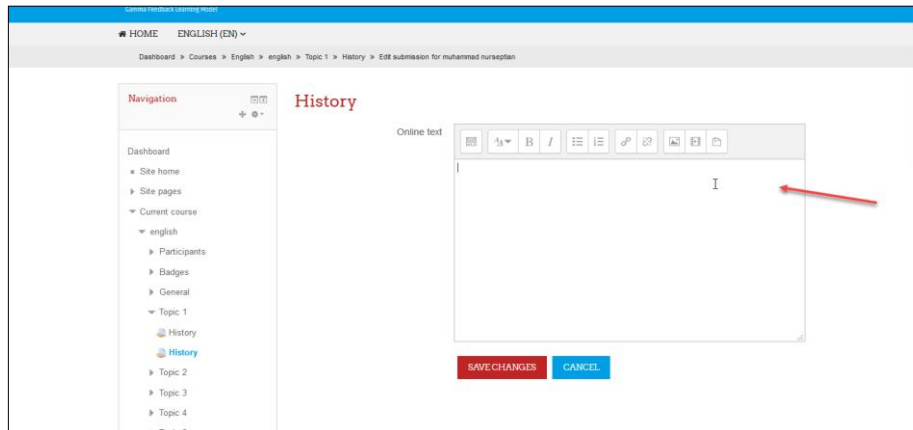
Gambar 19 Tampilan antarmuka *Grading Action*

- Dari antarmuka *Grade Action*, klik menu *Edit* sehingga tampil seperti Gambar 20.

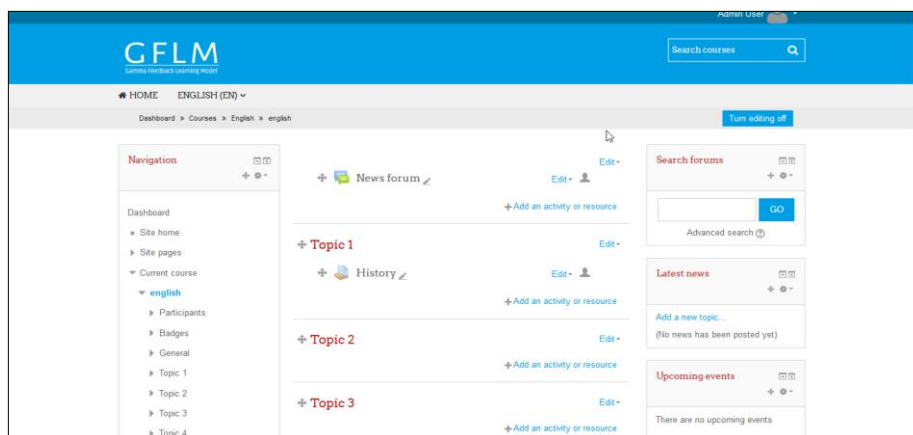


Gambar 20 Edit tugas secara *online*

- Masukkan sembarang *character* pada *text box* (Gambar 20), minimal 1 buah *character*, kemudian klik tombol “SAVE CHANGE” sehingga tampil seperti Gambar 21.



Gambar 21 Inisiasi teks tugas secara *online*



Gambar 22 Tampilan akhir matakuliah “*English*”

Gambar 22 memperlihatkan tampilan akhir matakuliah “*English*” berikut sebuah *Assignment* yang diberi nama “*History*”.