LAMPIRAN

Source Code Halaman Login

```
@section Scripts {
    @Scripts.Render("~/bundles/app")
}
<div class="row">
    <div class="col-sm-4">
        <form data-bind="submit: call Api">
            <h3>Test API</h3>
            <div class="form-group">
                <label>User</label>
                <input class="form-control" type="text" readonly data-bind="value: user" />
            </div>
            <div class="form-group error-messages" data-bind="foreach: errors">
                <p data-bind="text: $data" />
            </div>
        </form>
    </div>
    <div class="col-sm-4">
        <h3>Log In</h3>
        <form data-bind="submit: login">
            <div class="form-group">
                <label>Email</label>
                <input class="form-control" type="text" data-bind="value: loginEmail" />
            </div>
            <div class="form-group">
                <label>Password</label>
                <input class="form-control" type="password" data-bind="value: loginPassword" />
            </div>
            <div class="form-group">
                <button type="submit" class="btn btn-default">Log In</button>
                <button data-bind="click: logout" class="btn btn-default">Log Out</button>
            </div>
        </form>
    </div>
</div>
```

Source Code Login Access_Token AJAX

```
function ViewModel() {
    var self = this;

    var tokenKey = 'accessToken';

    self.result = ko.observable();
    self.user = ko.observable();

    self.registerEmail = ko.observable();
    self.registerPassword = ko.observable();
    self.registerPassword2 = ko.observable();
```

```
self.loginEmail = ko.observable();
self.loginPassword = ko.observable();
self.errors = ko.observableArray([]);

function showError(jqXHR) {

    self.result(jqXHR.status + ': ' + jqXHR.statusText);

    var response = jqXHR.responseJSON;
    if (response) {
        if (response.Message) self.errors.push(response.Message);
        if (response.ModelState) {
            var modelState = response.ModelState;
            for (var prop in modelState)
            {
                if (modelState.hasOwnProperty(prop)) {
                    var msgArr = modelState[prop]; // expect array here
                    if (msgArr.length) {
                        for (var i = 0; i < msgArr.length; ++i) self.errors.push(msgArr[i]);
                    }
                }
            }
        }
        if (response.error) self.errors.push(response.error);
        if (response.error_description) self.errors.push(response.error_description);
    }
}

self.callApi = function () {
    self.result('');
    self.errors.removeAll();

    var token = sessionStorage.getItem(tokenKey);
    var headers = {};
    if (token) {
        headers.Authorization = 'Bearer ' + token;
    }

    $.ajax({
        type: 'GET',
        url: '/api/values',

        headers: headers
    }).done(function (data) {
        self.result(data);
    }).fail(showError);
}

self.register = function () {
    self.result('');
    self.errors.removeAll();

    var data = {
        Email: self.registerEmail(),
        Password: self.registerPassword(),
        ConfirmPassword: self.registerPassword2()
```

```
};

        $.ajax({
            type: 'POST',
            url: '/api/Account/Register',
            contentType: 'application/json; charset=utf-8',


            data: JSON.stringify(data)
        }).done(function (data) {
            self.result("Done!");
        }).fail(showError);
    }

    self.login = function () {
        self.result('');
        self.errors.removeAll();

        var loginData = {
            grant_type: 'password',
            username: self.loginEmail(),
            password: self.loginPassword()
        };


        $.ajax({
            type: 'POST',
            url: '/Token',
            data: loginData
        }).done(function (data) {
            self.user(data.userName);
            // Cache the access token in session storage.
            sessionStorage.setItem(tokenKey, data.access_token);
            window.location.href = "/Guru/Home";
        }).fail(showError);
    }

    self.logout = function () {
        // Log out from the cookie based logon.
        var token = sessionStorage.getItem(tokenKey);
        var headers = {};
        if (token) {
            headers.Authorization = 'Bearer ' + token;
        }

        $.ajax({
            type: 'POST',
            url: '/api/Account/Logout',

            headers: headers
        }).done(function (data) {
            // Successfully logged out. Delete the token.
            self.user('');
            sessionStorage.removeItem(tokenKey);
        }).fail(showError);
```

```
            }
}

var app = new ViewModel();
ko.applyBindings(app);
```

Source Code API_DaTagurus.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using System.Web.Http.Description;
using FinalSkripsi.Models;
using System.Web.Http.OData;
using Newtonsoft.Json;
using System.Text;

namespace FinalSkripsi.Controllers
{
    public class API_DataGurusController : ApiController
    {
        private examdbEntities db = new examdbEntities();

        [EnableQuery]
        [Route("api/datagurus")]
        [HttpGet]
        // GET: api/API_DataGurus
        public HttpResponseMessage GetDataGuru()
        {
            try
            {
                return Request.CreateResponse(HttpStatusCode.Found, db.DataGurus.ToList());
            }
            catch (Exception)
            {
                return Request.CreateErrorResponse(HttpStatusCode.NotFound, "Data not found");
            }
        }

        [HttpGet]
        // GET: api/API_DataGurus/id
        public HttpResponseMessage Get(string guru_id, string nama_guru, int? mp_id, string jenis_kelamin,
string alamat, string nip, string sekolah)
        {
            try
            {
                return Request.CreateResponse(HttpStatusCode.Found, db.DataGurus.SingleOrDefault(p =>
p.Guru_Id == guru_id && p.Nama_Guru == nama_guru
```

```
&& p.MP_Id == mp_id && p.Jenis_kelamin == jenis_kelamin && p.Alamat == alamat && p.NIP == nip
&& p.Sekolah == sekolah));
            }
            catch (Exception)
            {
                return Request.CreateErrorResponse(HttpStatusCode.NotFound, " Data not found");
            }
        }

        // PUT: api/API_DataGurus/id
        [HttpPut]
        [Route("api/api_datagurus/{id}")]
        public HttpResponseMessage Put(string id, [FromBody] DataGuru dataGuru)
        {
            try
            {
                var entity = db.DataGurus.FirstOrDefault(p => p.Guru_Id == id);
                if (entity == null)
                    return Request.CreateErrorResponse(HttpStatusCode.BadRequest, " Data not found ");

                entity.Guru_Id = dataGuru.Guru_Id;
                entity.Nama_Guru = dataGuru.Nama_Guru;
                entity.MP_Id = dataGuru.MP_Id;
                entity.Jenis_kelamin = dataGuru.Jenis_kelamin;
                entity.Alamat = dataGuru.Alamat;
                entity.NIP = dataGuru.NIP;
                entity.Sekolah = dataGuru.Sekolah;
                db.SaveChanges();

                return Request.CreateResponse(HttpStatusCode.OK, "DataGuru Updated Successfully");
            }
            catch (Exception ex)
            {
                return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);
            }
        }

        [EnableQuery]
        [Route("api/datagurus/{id}")]
        [HttpPost]
        [ResponseType(typeof(DataGuru))]
        // POST: api/API_DataGurus
        public HttpResponseMessage Post([FromBody] DataGuru dataGuru)
        {
            try
            {
                db.DataGurus.Add(dataGuru);
                db.SaveChanges();
                var response = Request.CreateResponse(HttpStatusCode.Created, dataGuru);
                string uri = Url.Link("DefaultApi", new { id = dataGuru.Guru_Id });
                response.Headers.Location = new Uri(uri);

                return response;
            }
            catch(Exception)
```

```
            {
                return Request.CreateErrorResponse(HttpStatusCode.BadRequest, "Data not inserted");
            }
        }

        // DELETE: api/API_DataGurus/5
        [ResponseType(typeof(DataGuru))]
        [HttpDelete]
        public HttpResponseMessage DeleteDataGuru(string guru_id)
        {
            try
            {
                var dg = db.DataGurus.FirstOrDefault(p => p.Guru_Id == guru_id);
                if (dg == null)
                    return Request.CreateErrorResponse(HttpStatusCode.BadRequest, " not found to delete");
                db.DataGurus.Remove(dg);
                db.SaveChanges();

                return Request.CreateResponse(HttpStatusCode.OK, "DataGuru Deleted Successfully");
            }
            catch (Exception)
            {
                return Request.CreateErrorResponse(HttpStatusCode.BadRequest, "DataGuru not deleted");
            }
        }

        protected override void Dispose(bool disposing)
        {
            if (disposing)
            {
                db.Dispose();
            }
            base.Dispose(disposing);
        }

        private bool DataGuruExists(string id)
        {
            return db.DataGurus.Count(e => e.Guru_Id == id) > 0;
        }
    }
}
```

Source Code API ODATA DaTagurus.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using System.Web.Http.ModelBinding;
using System.Web.Http.OData;
using System.Web.Http.OData.Routing;
```

```
using FinalSkripsi.Models;

namespace FinalSkripsi.Controllers


{
    public class DaTagurusController : ODataController
    {
        private examdbEntities db = new examdbEntities();

        // GET: odata/DaTagurus
        [EnableQuery]
        [Route("api/api_daTagurus")]
        public IQueryable<DaTaguru> GetDaTagurus()
        {
            return db.DaTagurus;
        }

        // GET: odata/DaTagurus(5)
        [EnableQuery]
        [Route("api/api_daTagurus")]
        public SingleResult<DaTaguru> GetDaTaguru([FromODataUri] string key)
        {
            return SingleResult.Create(db.DaTagurus.Where(daTaguru => daTaguru.Guru_Id == key));
        }

        // PUT: odata/DaTagurus(5)
        public IHttpActionResult Put([FromODataUri] string key, Delta<DaTaguru> patch)
        {
            Validate(patch.GetEntity());

            if (!ModelState.IsValid)
            {
                return BadRequest(ModelState);
            }

            DaTaguru daTaguru = db.DaTagurus.Find(key);

            if (daTaguru == null)
            {
                return NotFound();
            }

            patch.Put(daTaguru);

            try
            {
                db.SaveChanges();
            }
            catch (DbUpdateConcurrencyException)
            {
                if (!DaTaguruExists(key))
                {
                    return NotFound();
                }
                else
```

```
            {
                throw;
            }
        }

        return Updated(da*Tag*uru);
    }

    // POST: odata/Da*Tag*urus
    [Route("*api*/*api*_da*Tag*urus/{id}")]
    public I*Http*ActionResult Post(Da*Tag*uru da*Tag*uru)
    {
        if (!*Model*State.IsValid)
        {
            return BadRequest(*Model*State);
        }

        db.Da*Tag*urus.Add(da*Tag*uru);

        try
        {
            db.SaveChanges();
        }
        catch (DbUpdateException)
        {
            if (Da*Tag*uruExists(da*Tag*uru.Guru_Id))
            {
                return Conflict();
            }
            else
            {
                throw;
            }
        }

        return Created(da*Tag*uru);
    }

    // PATCH: odata/Da*Tag*urus(5)
    [AcceptVerbs("PATCH", "MERGE")]
        public I*Http*ActionResult Patch([FromODataUri] string key,
        Delta<Data*Guru*> patch)
    {
        Validate(patch.GetEntity());

        if (!*Model*State.IsValid)
        {
            return BadRequest(*Model*State);
        }

        Da*Tag*uru da*Tag*uru = db.Da*Tag*urus.Find(key);
        if (da*Tag*uru == null)
        {
            return NotFound();
        }

        patch.Patch(da*Tag*uru);
```

```
            try
            {
                db.SaveChanges();
            }
            catch (DbUpdateConcurrencyException)
            {
                if (!DaTaguruExists(key))
                {
                    return NotFound();

                }
                else
                {
                    throw;
                }
            }

            return Updated(daTaguru);
        }

        // DELETE: odata/DaTagurus(5)
        public IHttpActionResult Delete([FromODataUri] string key)
        {
            DaTaguru daTaguru = db.DaTagurus.Find(key);
            if (daTaguru == null)
            {
                return NotFound();
            }

            db.DaTagurus.Remove(daTaguru);
            db.SaveChanges();

            return StatusCode(HttpStatusCode.NoContent);
        }

        // GET: odata/DaTagurus(5)/Grups
        [EnableQuery]
        public IQueryable<Grup> GetGrups([FromODataUri] string key)
        {
            return db.DaTagurus.Where(m => m.Guru_Id == key).SelectMany(m => m.Grups);
        }

        // GET: odata/DaTagurus(5)/Soals
        [EnableQuery]
        public IQueryable<Soal> GetSoals([FromODataUri] string key)
        {
            return db.DaTagurus.Where(m => m.Guru_Id == key).SelectMany(m => m.Soals);
        }

        // GET: odata/DaTagurus(5)/StandarNilais
        [EnableQuery]
        public IQueryable<StandarNilai> GetStandarNilais([FromODataUri] string key)
        {


            return db.DaTagurus.Where(m => m.Guru_Id == key).SelectMany(m => m.StandarNilais);
        }

        // GET: odata/DaTagurus(5)/Ujians
        [EnableQuery]
        public IQueryable<Ujian> GetUjians([FromODataUri] string key)
        {
            return db.DaTagurus.Where(m => m.Guru_Id == key).SelectMany(m => m.Ujians);
```

```
 }

      // GET: odata/DaTagurus(5)/MataPelajaran

      [EnableQuery]
      public SingleResult<MataPelajaran> GetMataPelajaran([FromODataUri] string key)
      {
         return SingleResult.Create(db.DaTagurus.Where(m => m.Guru_Id == key).Select(m =>
m.MataPelajaran));
      }

      protected override void Dispose(bool disposing)
      {
         if (disposing)
         {
            db.Dispose();
         }
         base.Dispose(disposing);
      }

      private bool DaTaguruExists(string key)
      {
         return db.DaTagurus.Count(e => e.Guru_Id == key) > 0;
      }
   }
}
```

Source Code API_DataSiswas.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using System.Web.Http.Description;
using FinalSkripsi.Models;
using System.Web.Http.OData;

namespace FinalSkripsi.Controllers
{
    public class API_DataSiswasController : ApiController
    {
        private examdbEntities db = new examdbEntities();

        // GET: api/API_DataSiswas
        [EnableQuery]
        [Route("api/api_datasiswas")]
        [HttpGet]
        public HttpResponseMessage GetDataSiswa()
        {
            try
            {
```

```
return Request.CreateResponse(HttpStatusCode.Found, db.DataSiswas.ToList());
            }
            catch (Exception)
            {
                return Request.CreateErrorResponse(HttpStatusCode.NotFound,
"Data not found");
            }
        }

        [HttpGet]
        // GET: api/API_DataSiswas/id
        public HttpResponseMessage Get(string siswa_id, string nama_siswa,
string jenis_kelamin, string sekolah, string alamat, string nisn, string
kelas)
        {
            try
            {
                return Request.CreateResponse(HttpStatusCode.Found,
db.DataSiswas.SingleOrDefault(p => p.Siswa_Id == siswa_id && p.Nama_Siswa ==
nama_siswa
                && p.Jenis_kelamin == jenis_kelamin && p.Sekolah == sekolah
&& p.Alamat == alamat && p.NISN == nisn && p.Kelas == kelas));
            }
            catch (Exception)
            {
                return Request.CreateErrorResponse(HttpStatusCode.NotFound,
" Data not found");
            }
        }

        // PUT: api/API_DataSiswas/5
        [HttpPut]
        [Route("api/api_datasiswas/{id}")]
        public HttpResponseMessage Put(string id, [FromBody] DataSiswa
dataSiswa)
        {
            try
            {
                var entity = db.DataSiswas.FirstOrDefault(p => p.Siswa_Id ==
id);
                if (entity == null)
                    return
Request.CreateErrorResponse(HttpStatusCode.BadRequest, " Data not found ");

                entity.Siswa_Id = dataSiswa.Siswa_Id;
                entity.Nama_Siswa = dataSiswa.Nama_Siswa;
                entity.Jenis_kelamin = dataSiswa.Jenis_kelamin;
                entity.Sekolah = dataSiswa.Sekolah;
                entity.Alamat = dataSiswa.Alamat;
                entity.NISN = dataSiswa.NISN;
                entity.Kelas = dataSiswa.Kelas;
                db.SaveChanges();

                return Request.CreateResponse(HttpStatusCode.OK, "DataSiswa
Updated Successfully");
            }
            catch (Exception ex)
            {
```

```
return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);
            }
        }

        // POST: api/API_DataSiswas
        [EnableQuery]
        [Route("api/api_datasiswas/{id}")]
        [ResponseType(typeof(DataSiswa))]
        [HttpPost]
        public HttpResponseMessage Post([FromBody] DataSiswa dataSiswa)
        {
            try
            {
                db.DataSiswas.Add(dataSiswa);
                db.SaveChanges();
                var response =
Request.CreateResponse(HttpStatusCode.Created, dataSiswa);
                string uri = Url.Link("DefaultApi", new { id =
dataSiswa.Siswa_Id });
                response.Headers.Location = new Uri(uri);

                return response;
            }
            catch (Exception)
            {
                return
Request.CreateErrorResponse(HttpStatusCode.BadRequest, "Data not inserted");
            }
        }


        // DELETE: api/API_DataSiswas/5
        [ResponseType(typeof(DataSiswa))]
        public IHttpActionResult DeleteDataSiswa(string id)
        {
            DataSiswa dataSiswa = db.DataSiswas.Find(id);
            if (dataSiswa == null)
            {
                return NotFound();
            }

            db.DataSiswas.Remove(dataSiswa);
            db.SaveChanges();

            return Ok(dataSiswa);
        }

        protected override void Dispose(bool disposing)
        {
            if (disposing)
            {
                db.Dispose();
            }
            base.Dispose(disposing);
        }

        private bool DataSiswaExists(string id)
        {
            return db.DataSiswas.Count((e => e.Siswa_Id == id)) > 0;
```

```
            }
        }
}
```

Source Code API ODATA DataSiswas.cs

```
using System;
using System.Collections.Generic;

using System.Data;
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using System.Web.Http.ModelBinding;
using System.Web.Http.OData;
using System.Web.Http.OData.Routing;
using FinalSkripsi.Models;

namespace FinalSkripsi.Controllers
{
    public class DataSiswasController : ODataController
    {
        private examdbEntities db = new examdbEntities();

        // GET: odata/DataSiswas
        [EnableQuery]
        [Route("api/api_datasiswas")]
        public IQueryable<DataSiswa> GetDataSiswas()
        {
            return db.DataSiswas;
        }

        // GET: odata/DataSiswas(5)
        [EnableQuery]
        [Route("api/api_datasiswas")]
              public SingleResult<DataSiswa> GetDataSiswa([FromODataUri] string key)
        {
            return SingleResult.Create(db.DataSiswas.Where(dataSiswa => dataSiswa.Siswa_Id == key));
        }

        // PUT: odata/DataSiswas(5)
        public IHttpActionResult Put([FromODataUri] string key, Delta<DataSiswa> patch)
        {
            Validate(patch.GetEntity());

            if (!ModelState.IsValid)
            {
                return BadRequest(ModelState);
            }

            DataSiswa dataSiswa = db.DataSiswas.Find(key);
```

```
    if (dataSiswa == null)
    {
        return NotFound();
    }



    patch.Put(dataSiswa);

    try
    {
        db.SaveChanges();
    }
    catch (DbUpdateConcurrencyException)
    {
        if (!DataSiswaExists(key))
        {
            return NotFound();
        }
        else
        {
            throw;
        }
    }

    return Updated(dataSiswa);
}

// POST: odata/DataSiswas
[Route("api/api_datasiswas/{id}")]
public IHttpActionResult Post(DataSiswa dataSiswa)
{
    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }

    db.DataSiswas.Add(dataSiswa);

    try
    {
        db.SaveChanges();
    }
    catch (DbUpdateException)
    {
        if (DataSiswaExists(dataSiswa.Siswa_Id))
        {
            return Conflict();
        }
        else
        {
            throw;
        }
    }

    return Created(dataSiswa);
```

```
}

// PATCH: odata/DataSiswas(5)
[AcceptVerbs("PATCH", "MERGE")]
        public IHttpActionResult Patch([FromODataUri] string key, Delta<DataSiswa> patch)
{
   Validate(patch.GetEntity());



   if (!ModelState.IsValid)
   {
      return BadRequest(ModelState);
   }

   DataSiswa dataSiswa = db.DataSiswas.Find(key);
   if (dataSiswa == null)
   {
      return NotFound();
   }

   patch.Patch(dataSiswa);

   try
   {
      db.SaveChanges();
   }
   catch (DbUpdateConcurrencyException)
   {
      if (!DataSiswaExists(key))
      {
         return NotFound();
      }
      else
      {
         throw;
      }
   }

   return Updated(dataSiswa);
}

// DELETE: odata/DataSiswas(5)
public IHttpActionResult Delete([FromODataUri] string key)
{
   DataSiswa dataSiswa = db.DataSiswas.Find(key);
   if (dataSiswa == null)
   {
      return NotFound();
   }

   db.DataSiswas.Remove(dataSiswa);
   db.SaveChanges();

   return StatusCode(HttpStatusCode.NoContent);
```

```
}

        // GET: odata/DataSiswas(5)/Nilais
        [EnableQuery]
        public IQueryable<Nilai> GetNilais([FromODataUri] string key)
        {
                        return db.DataSiswas.Where(m => m.Siswa_Id == key).SelectMany(m => m.Nilais);
        }

        protected override void Dispose(bool disposing)
        {
          if (disposing)

          {
            db.Dispose();
          }
          base.Dispose(disposing);
        }

        private bool DataSiswaExists(string key)
        {
          return db.DataSiswas.Count(e => e.Siswa_Id == key) > 0;
        }
    }
}
```

Source Code API_MataPelajarans.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using System.Web.Http.Description;
using FinalSkripsi.Models;
using System.Web.Http.OData;

namespace FinalSkripsi.Controllers
{
    public class API_MataPelajaransController : ApiController
    {
        private examdbEntities db = new examdbEntities();


        [EnableQuery]
        [Route("api/api_matapelajarans")]
        [HttpGet]
        // GET: api/API_MataPelajarans
        public HttpResponseMessage GetMataPelajaran()
        {
          Try
```

```
            {
                return Request.CreateResponse(HttpStatusCode.Found, db.MataPelajarans.ToList());
            }
            catch (Exception)
            {
                return Request.CreateErrorResponse(HttpStatusCode.NotFound, "Data not found");
            }
        }

        [HttpGet]
        // GET: api/API_MataPelajarans/id
        public HttpResponseMessage Get(int id, string nama_mp)
        {
            try
            {
                return Request.CreateResponse(HttpStatusCode.Found, db.MataPelajarans.SingleOrDefault(p =>
p.MP_Id == id && p.Nama_MP == nama_mp));
            }
            catch (Exception)
            {
                return Request.CreateErrorResponse(HttpStatusCode.NotFound, " Data not found");
            }
        }

        // PUT: api/API_MataPelajarans/id
        [HttpPut]
        [Route("api/api_matapelajarans/{id}")]
        public HttpResponseMessage Put(int id, [FromBody] MataPelajaran mataPelajaran)
        {
            try
            {
                var entity = db.MataPelajarans.SingleOrDefault(p => p.MP_Id == id );
                if (entity == null)
                    return Request.CreateErrorResponse(HttpStatusCode.BadRequest, " Data not found ");

                entity.MP_Id = mataPelajaran.MP_Id;
                entity.Nama_MP = mataPelajaran.Nama_MP;
                db.SaveChanges();

                return Request.CreateResponse(HttpStatusCode.OK, "MataPelajaran Updated Successfully");
            }
            catch (Exception ex)
            {
                return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);
            }
        }

        // POST: api/API_MataPelajarans
        [EnableQuery]
        [Route("api/api_matapelajarans/{id}")]
        [HttpPost]
        [ResponseType(typeof(MataPelajaran))]
        public HttpResponseMessage Post([FromBody] MataPelajaran mataPelajaran)
        {
            try
```

```
{
            db.MataPelajarans.Add(mataPelajaran);
            db.SaveChanges();
            var response = Request.CreateResponse(HttpStatusCode.Created, mataPelajaran);
            response.Headers.Location = Request.RequestUri;

            return response;
        }
        catch (Exception)
        {
            return Request.CreateErrorResponse(HttpStatusCode.BadRequest, "Data not inserted");
        }
    }

    // DELETE: api/API_MataPelajarans/5
    [ResponseType(typeof(MataPelajaran))]
    [HttpDelete]
    public HttpResponseMessage DeleteMataPelajaran(int id)
    {
        try
        {
            var mp = db.MataPelajarans.SingleOrDefault(p => p.MP_Id == id);
            if (mp == null)
                return Request.CreateErrorResponse(HttpStatusCode.BadRequest, " Data not found to delete");
            db.MataPelajarans.Remove(mp);
            db.SaveChanges();

            return Request.CreateResponse(HttpStatusCode.OK, "MataPelajaran Deleted Successfully");
        }
        catch (Exception)
        {
            return Request.CreateErrorResponse(HttpStatusCode.BadRequest, "MataPelajaran not deleted");
        }
    }

    protected override void Dispose(bool disposing)
    {
        if (disposing)
        {
            db.Dispose();
        }
        base.Dispose(disposing);
    }

    private bool MataPelajaranExists(int id)
    {
        return db.MataPelajarans.Count(e => e.MP_Id == id) > 0;
    }
     }
}
```

Source Code API ODATA MataPelajarans.cs

```
using System;
using System.Collections.Generic;
using System.Data;
```

```
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using System.Web.Http.ModelBinding;
using System.Web.Http.OData;
using System.Web.Http.OData.Routing;
using FinalSkripsi.Models;

namespace FinalSkripsi.Controllers
{
   /*
   public class MataPelajaransController : ODataController
   {

      private examdbEntities db = new examdbEntities();

      // GET: odata/MataPelajarans
      [EnableQuery]
      public IQueryable<MataPelajaran> GetMataPelajarans()
      {
         return db.MataPelajarans;
      }

      // GET: odata/MataPelajarans(5)
      [EnableQuery]
      public SingleResult<MataPelajaran> GetMataPelajaran([FromODataUri] int key)
      {
         return SingleResult.Create(db.MataPelajarans.Where(mataPelajaran => mataPelajaran.MP_Id ==
key));
      }

      // PUT: odata/MataPelajarans(5)
      public IHttpActionResult Put([FromODataUri] int key, Delta<MataPelajaran> patch)
      {
         Validate(patch.GetEntity());

         if (!ModelState.IsValid)
         {
            return BadRequest(ModelState);
         }

         MataPelajaran mataPelajaran = db.MataPelajarans.Find(key);
         if (mataPelajaran == null)
         {
            return NotFound();
         }

         patch.Put(mataPelajaran);

         try
         {
            db.SaveChanges();
         }
```

```
        catch (DbUpdateConcurrencyException)
        {
            if (!MataPelajaranExists(key))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }

        return Updated(mataPelajaran);
    }

    // POST: odata/MataPelajarans
    public IHttpActionResult Post(MataPelajaran mataPelajaran)
    {

        if (!ModelState.IsValid)
        {
            return BadRequest(ModelState);
        }

        db.MataPelajarans.Add(mataPelajaran);
        db.SaveChanges();

        return Created(mataPelajaran);
    }

    // PATCH: odata/MataPelajarans(5)
    [AcceptVerbs("PATCH", "MERGE")]
    public IHttpActionResult Patch([FromODataUri] int key, Delta<MataPelajaran> patch)
    {
        Validate(patch.GetEntity());

        if (!ModelState.IsValid)
        {
            return BadRequest(ModelState);
        }

        MataPelajaran mataPelajaran = db.MataPelajarans.Find(key);
        if (mataPelajaran == null)
        {
            return NotFound();

        }

        patch.Patch(mataPelajaran);

        try
        {
            db.SaveChanges();
        }
        catch (DbUpdateConcurrencyException)
        {
```

```
            if (!MataPelajaranExists(key))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }

        return Updated(mataPelajaran);
    }

    // DELETE: odata/MataPelajarans(5)
    public IHttpActionResult Delete([FromODataUri] int key)
    {
        MataPelajaran mataPelajaran = db.MataPelajarans.Find(key);
        if (mataPelajaran == null)
        {
            return NotFound();
        }


        db.MataPelajarans.Remove(mataPelajaran);
        db.SaveChanges();

        return StatusCode(HttpStatusCode.NoContent);
    }

    // GET: odata/MataPelajarans(5)/DaTagurus
    [EnableQuery]
    public IQueryable<DaTaguru> GetDaTagurus([FromODataUri] int key)
    {
        return db.MataPelajarans.Where(m => m.MP_Id == key).SelectMany(m => m.DaTagurus);
    }

    // GET: odata/MataPelajarans(5)/Ujians
    [EnableQuery]
    public IQueryable<Ujian> GetUjians([FromODataUri] int key)
    {
        return db.MataPelajarans.Where(m => m.MP_Id == key).SelectMany(m => m.Ujians);
    }

    protected override void Dispose(bool disposing)
    {
        if (disposing)
        {
            db.Dispose();
        }
        base.Dispose(disposing);
    }

    private bool MataPelajaranExists(int key)
    {
        return db.MataPelajarans.Count(e => e.MP_Id == key) > 0;
    }
```

```
}
}
```

Source Code API_Nilais.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using System.Web.Http.Description;
using FinalSkripsi.Models;
using System.Web.Http.OData;

namespace FinalSkripsi.Controllers

{
    public class API_NilaisController : ApiController
    {
        private examdbEntities db = new examdbEntities();

        // GET: api/API_Nilais
        [EnableQuery]
        [Route("api/api_nilais")]
        public IQueryable<Nilai> GetNilais()
        {
            return db.Nilais;
        }

        // GET: api/API_Nilais/5
        [ResponseType(typeof(Nilai))]
        public IHttpActionResult GetNilai(int id)
        {
            Nilai nilai = db.Nilais.Find(id);
            if (nilai == null)
            {
                return NotFound();
            }

            return Ok(nilai);
        }

        // PUT: api/API_Nilais/5
        [ResponseType(typeof(void))]
        public IHttpActionResult PutNilai(int id, Nilai nilai)
        {
            if (!ModelState.IsValid)
            {
                return BadRequest(ModelState);
            }
```

```
        if (id != nilai.Nilai_Id)
        {
            return BadRequest();
        }

        db.Entry(nilai).State = EntityState.Modified;

        try
        {
            db.SaveChanges();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!NilaiExists(id))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }

        return StatusCode(HttpStatusCode.NoContent);

    }

    // POST: api/API_Nilais

    [EnableQuery]
    [Route("api/api_nilais/{id}")]
    [ResponseType(typeof(Nilai))]
    public IHttpActionResult PostNilai(Nilai nilai)
    {
        if (!ModelState.IsValid)
        {
            return BadRequest(ModelState);
        }

        db.Nilais.Add(nilai);
        db.SaveChanges();

        return CreatedAtRoute("DefaultApi", new { id = nilai.Nilai_Id }, nilai);
    }

    // DELETE: api/API_Nilais/5
    [ResponseType(typeof(Nilai))]
    public IHttpActionResult DeleteNilai(int id)
    {
        Nilai nilai = db.Nilais.Find(id);
        if (nilai == null)
        {
            return NotFound();
        }

        db.Nilais.Remove(nilai);
```

```
db.SaveChanges();

            return Ok(nilai);
        }

        protected override void Dispose(bool disposing)
        {
            if (disposing)
            {
                db.Dispose();
            }
            base.Dispose(disposing);
        }

        private bool NilaiExists(int id)
        {
            return db.Nilais.Count(e => e.Nilai_Id == id) > 0;
        }
```

Source Code API ODATA Nilais.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using System.Web.Http.ModelBinding;
using System.Web.Http.OData;
using System.Web.Http.OData.Routing;
using FinalSkripsi.Models;

namespace FinalSkripsi.Controllers
{

    public class NilaisController : ODataController
    {
        private examdbEntities db = new examdbEntities();

        // GET: odata/Nilais
        [EnableQuery]
        public IQueryable<Nilai> GetNilais()
        {
            return db.Nilais;
        }

        // GET: odata/Nilais(5)
        [EnableQuery]
```

```
public SingleResult<Nilai> GetNilai([FromODataUri] int key)
{
    return SingleResult.Create(db.Nilais.Where(nilai => nilai.Nilai_Id == key));
}

// PUT: odata/Nilais(5)
public IHttpActionResult Put([FromODataUri] int key, Delta<Nilai> patch)
{
    Validate(patch.GetEntity());

    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }

    Nilai nilai = db.Nilais.Find(key);
    if (nilai == null)
    {
        return NotFound();
    }

    patch.Put(nilai);

    try
    {
        db.SaveChanges();
    }
    catch (DbUpdateConcurrencyException)
    {
        if (!NilaiExists(key))
        {
            return NotFound();
        }
        else
        {
            throw;
        }
    }

    return Updated(nilai);
}

// POST: odata/Nilais
public IHttpActionResult Post(Nilai nilai)
{
    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }

    db.Nilais.Add(nilai);
    db.SaveChanges();

    return Created(nilai);
}
```

```
// PATCH: odata/Nilais(5)
[AcceptVerbs("PATCH", "MERGE")]
public IHttpActionResult Patch([FromODataUri] int key, Delta<Nilai> patch)
{
    Validate(patch.GetEntity());

    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }

    Nilai nilai = db.Nilais.Find(key);
    if (nilai == null)
    {
        return NotFound();
    }

    patch.Patch(nilai);

    try
    {
        db.SaveChanges();
    }

    catch (DbUpdateConcurrencyException)
    {
        if (!NilaiExists(key))
        {
            return NotFound();
        }
        else
        {
            throw;
        }
    }

    return Updated(nilai);
}

// DELETE: odata/Nilais(5)
public IHttpActionResult Delete([FromODataUri] int key)
{
    Nilai nilai = db.Nilais.Find(key);
    if (nilai == null)
    {
        return NotFound();
    }

    db.Nilais.Remove(nilai);
    db.SaveChanges();

    return StatusCode(HttpStatusCode.NoContent);
}

// GET: odata/Nilais(5)/DataSiswa
[EnableQuery]
```

```
        public SingleResult<DataSiswa> GetDataSiswa([FromODataUri] int key)
        {
            return SingleResult.Create(db.Nilais.Where(m => m.Nilai_Id == key).Select(m => m.DataSiswa));
        }

        protected override void Dispose(bool disposing)
        {
            if (disposing)
            {
                db.Dispose();
            }
            base.Dispose(disposing);
        }

        private bool NilaiExists(int key)
        {
            return db.Nilais.Count(e => e.Nilai_Id == key) > 0;
        }
    }
}
```

Source Code API_Nilais.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using System.Web.Http.Description;
using FinalSkripsi.Models;
using System.Web.Http.OData;

namespace FinalSkripsi.Controllers
{
    public class API_SoalsController : ApiController
    {
        private examdbEntities db = new examdbEntities();

        // GET: api/API_Soals
        [EnableQuery]
        [Route("api/api_soals")]
        public IQueryable<Soal> GetSoals()
        {
            return db.Soals;
        }

        // GET: api/API_Soals/5
        [ResponseType(typeof(Soal))]
        public IHttpActionResult GetSoal(int id)
        {
            Soal soal = db.Soals.Find(id);
```

```
            if (soal == null)
            {
                return NotFound();
            }

            return Ok(soal);
        }

        // PUT: api/API_Soals/5
        [ResponseType(typeof(void))]
        public IHttpActionResult PutSoal(int id, Soal soal)
        {
            if (!ModelState.IsValid)
            {

                return BadRequest(ModelState);
            }

            if (id != soal.Soal_Id)
            {
                return BadRequest();
            }

            db.Entry(soal).State = EntityState.Modified;

            try
            {
                db.SaveChanges();
            }
            catch (DbUpdateConcurrencyException)
            {
                if (!SoalExists(id))
                {
                    return NotFound();
                }
                else
                {
                    throw;
                }
            }

            return StatusCode(HttpStatusCode.NoContent);
        }

        [EnableQuery]
        [Route("api/api_soals/{id}")]
        // POST: api/API_Soals
        [ResponseType(typeof(Soal))]
        public IHttpActionResult PostSoal(Soal soal)
        {
            if (!ModelState.IsValid)
            {
                return BadRequest(ModelState);
            }

            db.Soals.Add(soal);
```

```
            db.SaveChanges();

            return CreatedAtRoute("DefaultApi", new { id = soal.Soal_Id }, soal);
        }

        // DELETE: api/API_Soals/5
        [ResponseType(typeof(Soal))]
        public IHttpActionResult DeleteSoal(int id)
        {
            Soal soal = db.Soals.Find(id);
            if (soal == null)
            {
                return NotFound();
            }

            db.Soals.Remove(soal);
            db.SaveChanges();

            return Ok(soal);
        }
```

```
        protected override void Dispose(bool disposing)
        {
            if (disposing)
            {
                db.Dispose();
            }

            base.Dispose(disposing);
        }

        private bool SoalExists(int id)
        {
            return db.Soals.Count(e => e.Soal_Id == id) > 0;
        }
    }
}
```

Source Code API ODATA Soals.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using System.Web.Http.ModelBinding;
using System.Web.Http.OData;
using System.Web.Http.OData.Routing;
using FinalSkripsi.Models;

namespace FinalSkripsi.Controllers
```

```
    {
        public class SoalsController : ODataController
        {
            private examdbEntities db = new examdbEntities();

            // GET: odata/Soals
            [EnableQuery]
            public IQueryable<Soal> GetSoals()
            {
                return db.Soals;
            }

            // GET: odata/Soals(5)
            [EnableQuery]
            public SingleResult<Soal> GetSoal([FromODataUri] int key)
            {
                return SingleResult.Create(db.Soals.Where(soal => soal.Soal_Id == key));
            }

            // PUT: odata/Soals(5)
            public IHttpActionResult Put([FromODataUri] int key, Delta<Soal> patch)
            {
                Validate(patch.GetEntity());

    if (!ModelState.IsValid)

    {
        return BadRequest(ModelState);
    }

    Soal soal = db.Soals.Find(key);
    if (soal == null)
    {
        return NotFound();
    }

    patch.Put(soal);

    try
    {
        db.SaveChanges();
    }
    catch (DbUpdateConcurrencyException)
    {
        if (!SoalExists(key))
        {
            return NotFound();
        }
        else
        {
            throw;
        }
    }

    return Updated(soal);
}

// POST: odata/Soals
public IHttpActionResult Post(Soal soal)
{
    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
```

```
        }

        db.Soals.Add(soal);
        db.SaveChanges();

        return Created(soal);
    }

    // PATCH: odata/Soals(5)
    [AcceptVerbs("PATCH", "MERGE")]
    public IHttpActionResult Patch([FromODataUri] int key, Delta<Soal> patch)
    {
        Validate(patch.GetEntity());

        if (!ModelState.IsValid)
        {
            return BadRequest(ModelState);
        }

        Soal soal = db.Soals.Find(key);
        if (soal == null)
        {
            return NotFound();
        }




        patch.Patch(soal);

        try
        {

            db.SaveChanges();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!SoalExists(key))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }

        return Updated(soal);
    }

    // DELETE: odata/Soals(5)
    public IHttpActionResult Delete([FromODataUri] int key)
    {
        Soal soal = db.Soals.Find(key);
        if (soal == null)
        {
            return NotFound();
        }

        db.Soals.Remove(soal);
        db.SaveChanges();
```

```
            return StatusCode(HttpStatusCode.NoContent);
        }

        // GET: odata/Soals(5)/DaTaguru
        [EnableQuery]
        public SingleResult<DaTaguru> GetDaTaguru([FromODataUri] int key)
        {
            return SingleResult.Create(db.Soals.Where(m => m.Soal_Id == key).Select(m => m.DaTaguru));
        }

        protected override void Dispose(bool disposing)
        {
            if (disposing)
            {
                db.Dispose();
            }
            base.Dispose(disposing);
        }

        private bool SoalExists(int key)
        {
            return db.Soals.Count(e => e.Soal_Id == key) > 0;
        }
    }
}
```

Source Code API_StandarNilaisController.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using System.Web.Http.Description;
using FinalSkripsi.Models;
using System.Web.Http.OData;

namespace FinalSkripsi.Controllers
{
    public class API_StandarNilaisController : ApiController
    {
        private examdbEntities db = new examdbEntities();

        // GET: api/API_StandarNilais
        [EnableQuery]
        [Route("api/api_standarnilais")]
        public IQueryable<StandarNilai> GetStandarNilais()
        {
            return db.StandarNilais;
        }
```

```
        // GET: api/API_StandarNilais/5
        [ResponseType(typeof(StandarNilai))]
        public IHttpActionResult GetStandarNilai(int id)
        {
            StandarNilai standarNilai = db.StandarNilais.Find(id);
            if (standarNilai == null)
            {
                return NotFound();
            }

            return Ok(standarNilai);
        }

        // PUT: api/API_StandarNilais/5
        [ResponseType(typeof(void))]
        public IHttpActionResult PutStandarNilai(int id, StandarNilai standarNilai)
        {
            if (!ModelState.IsValid)
            {
                return BadRequest(ModelState);
            }

            if (id != standarNilai.SN_Id)
            {
                return BadRequest();
            }


            db.Entry(standarNilai).State = EntityState.Modified;

            try
            {
                db.SaveChanges();
            }
            catch (DbUpdateConcurrencyException)
            {
                if (!StandarNilaiExists(id))
                {
                    return NotFound();
                }
                else
                {
                    throw;
                }
            }

            return StatusCode(HttpStatusCode.NoContent);
        }

        // POST: api/API_StandarNilais
        [EnableQuery]
        [Route("api/api_standarnilais/{id}")]
        [ResponseType(typeof(StandarNilai))]
        public IHttpActionResult PostStandarNilai(StandarNilai standarNilai)
        {
```

```
        if (!ModelState.IsValid)
            {
                return BadRequest(ModelState);
            }

            db.StandarNilais.Add(standarNilai);
            db.SaveChanges();

            return CreatedAtRoute("DefaultApi", new { id = standarNilai.SN_Id }, standarNilai);
        }

        // DELETE: api/API_StandarNilais/5
        [ResponseType(typeof(StandarNilai))]
        public IHttpActionResult DeleteStandarNilai(int id)
        {
            StandarNilai standarNilai = db.StandarNilais.Find(id);
            if (standarNilai == null)
            {
                return NotFound();
            }

            db.StandarNilais.Remove(standarNilai);
            db.SaveChanges();

            return Ok(standarNilai);
        }

        protected override void Dispose(bool disposing)
        {
            if (disposing)
            {
                db.Dispose();
            }
            base.Dispose(disposing);
        }

        private bool StandarNilaiExists(int id)
        {
            return db.StandarNilais.Count(e => e.SN_Id == id) > 0;
        }
    }
}
```

Source Code API ODATA StandarNilaisController.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Linq;
using System.Net;
using System.Net.Http;
```

```
using System.Web.Http;
using System.Web.Http.ModelBinding;
using System.Web.Http.OData;
using System.Web.Http.OData.Routing;
using FinalSkripsi.Models;

namespace FinalSkripsi.Controllers

{
   public class StandarNilaisController : ODataController
   {
      private examdbEntities db = new examdbEntities();

      // GET: odata/StandarNilais
      [EnableQuery]
      public IQueryable<StandarNilai> GetStandarNilais()
      {
         return db.StandarNilais;
      }

      // GET: odata/StandarNilais(5)
      [EnableQuery]
      public SingleResult<StandarNilai> GetStandarNilai([FromODataUri] int key)
      {
         return SingleResult.Create(db.StandarNilais.Where(standarNilai => standarNilai.SN_Id == key));
      }

      // PUT: odata/StandarNilais(5)
      public IHttpActionResult Put([FromODataUri] int key, Delta<StandarNilai> patch)
      {

         Validate(patch.GetEntity());

         if (!ModelState.IsValid)
         {
            return BadRequest(ModelState);
         }

         StandarNilai standarNilai = db.StandarNilais.Find(key);
         if (standarNilai == null)
         {


    return NotFound();
         }

         patch.Put(standarNilai);

         try
         {
            db.SaveChanges();
         }
         catch (DbUpdateConcurrencyException)
         {
            if (!StandarNilaiExists(key))
            {
```

```
            return NotFound();
        }
        else
        {
            throw;
        }
    }

    return Updated(standarNilai);
}

// POST: odata/StandarNilais
public IHttpActionResult Post(StandarNilai standarNilai)
{
    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }

    db.StandarNilais.Add(standarNilai);
    db.SaveChanges();

    return Created(standarNilai);
}

// PATCH: odata/StandarNilais(5)
[AcceptVerbs("PATCH", "MERGE")]
public IHttpActionResult Patch([FromODataUri] int key, Delta<StandarNilai> patch)
{
    Validate(patch.GetEntity());

    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }

    StandarNilai standarNilai = db.StandarNilais.Find(key);
    if (standarNilai == null)
    {
        return NotFound();
    }

    patch.Patch(standarNilai);

    try
    {
        db.SaveChanges();
    }
    catch (DbUpdateConcurrencyException)
    {
        if (!StandarNilaiExists(key))
        {
            return NotFound();
        }
        else
        {
```

```
                throw;
            }
        }



        return Updated(standarNilai);
    }

    // DELETE: odata/StandarNilais(5)
    public IHttpActionResult Delete([FromODataUri] int key)
    {
        StandarNilai standarNilai = db.StandarNilais.Find(key);
        if (standarNilai == null)
        {
            return NotFound();
        }

        db.StandarNilais.Remove(standarNilai);
        db.SaveChanges();

        return StatusCode(HttpStatusCode.NoContent);
    }

    // GET: odata/StandarNilais(5)/DaTaguru
    [EnableQuery]
    public SingleResult<DaTaguru> GetDaTaguru([FromODataUri] int key)
    {
        return SingleResult.Create(db.StandarNilais.Where(m => m.SN_Id == key).Select(m =>
m.DaTaguru));
    }

    protected override void Dispose(bool disposing)
    {
        if (disposing)
        {
            db.Dispose();
        }
        base.Dispose(disposing);
    }



    private bool StandarNilaiExists(int key)
    {
        return db.StandarNilais.Count(e => e.SN_Id == key) > 0;
    }
  }
}
```

Source Code API_TagsController.cs

```
using System;
using System.Collections.Generic;
using System.Data;
```

```
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using System.Web.Http.Description;
using FinalSkripsi.Models;
using System.Web.Http.OData;

namespace FinalSkripsi.Controllers
{
    public class API_TagsController : ApiController
    {
        private examdbEntities db = new examdbEntities();

        // GET: api/API_Tags
        [EnableQuery]
        [Route("api/api_Tags")]
        public IQueryable<Tag> GetTags()
        {
            return db.Tags;
        }

        // GET: api/API_Tags/5
        [ResponseType(typeof(Tag))]
        public IHttpActionResult GetTag(int id)
        {
            Tag Tag = db.Tags.Find(id);
            if (Tag == null)
            {
                return NotFound();
            }

            return Ok(Tag);
        }

        // PUT: api/API_Tags/5
        [ResponseType(typeof(void))]
        public IHttpActionResult PutTag(int id, Tag Tag)
        {

            if (!ModelState.IsValid)
            {
                return BadRequest(ModelState);
            }

            if (id != Tag.Tag_Id)
            {
                return BadRequest();
            }

            db.Entry(Tag).State = EntityState.Modified;

            try
            {
```

```
            db.SaveChanges();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!TagExists(id))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }

        return StatusCode(HttpStatusCode.NoContent);
    }

    // POST: api/API_Tags
    [EnableQuery]
    [Route("api/api_Tags/{id}")]
    [ResponseType(typeof(Tag))]
    public IHttpActionResult PostTag(Tag Tag)
    {
        if (!ModelState.IsValid)
        {
            return BadRequest(ModelState);
        }

        db.Tags.Add(Tag);
        db.SaveChanges();

        return CreatedAtRoute("DefaultApi", new { id = Tag.Tag_Id }, Tag);
    }

    // DELETE: api/API_Tags/5
    [ResponseType(typeof(Tag))]
    public IHttpActionResult DeleteTag(int id)
    {
        Tag Tag = db.Tags.Find(id);
        if (Tag == null)
        {
            return NotFound();
        }



        db.Tags.Remove(Tag);
        db.SaveChanges();

        return Ok(Tag);
    }

    protected override void Dispose(bool disposing)
    {
        if (disposing)
        {
```

```
                db.Dispose();
            }

        base.Dispose(disposing);
        }

        private bool TagExists(int id)
        {
            return db.Tags.Count(e => e.Tag_Id == id) > 0;
        }
    }
}
```

Source Code API ODATA TagsController.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using System.Web.Http.ModelBinding;
using System.Web.Http.OData;
using System.Web.Http.OData.Routing;
using FinalSkripsi.Models;
using System.Threading.Tasks;

namespace FinalSkripsi.Controllers
{
    public class TagsController : ODataController
    {

        private examdbEntities db = new examdbEntities();

        // GET: odata/Tags
        [EnableQuery]
        public IQueryable<Tag> GetTags()
        {
            return db.Tags;

        }


        // GET: odata/Tags(5)
        [EnableQuery]
        public SingleResult<Tag> GetTag([FromODataUri] int key)
        {
            return SingleResult.Create(db.Tags.Where(Tag => Tag.Tag_Id == key));
        }

        // PUT: odata/Tags(5)
        public IHttpActionResult Put([FromODataUri] int key, Delta<Tag> patch)
        {
```

```
        Validate(patch.GetEntity());

        if (!ModelState.IsValid)
        {
            return BadRequest(ModelState);
        }

        Tag tag = db.Tags.Find(key);
        if (tag == null)
        {
            return NotFound();
        }

        patch.Put(tag);

        try
        {
            db.SaveChanges();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!TagExists(key))
            {
                return NotFound();

            }
            else
            {
                throw;
            }
        }

        return Updated(tag);
    }

    // POST: odata/Tags
    public IHttpActionResult Post(Tag tag)
    {
        if (!ModelState.IsValid)
        {
            return BadRequest(ModelState);
        }

        db.Tags.Add(tag);
        db.SaveChanges();

        return Created(tag);

    }

    // PATCH: odata/Tags(5)
    [AcceptVerbs("PATCH", "MERGE")]
    public IHttpActionResult Patch([FromODataUri] int key, Delta<Tag> patch)
    {
        Validate(patch.GetEntity());
```

```
            if (!ModelState.IsValid)
            {
                return BadRequest(ModelState);
            }

            Tag Tag = db.Tags.Find(key);
            if (Tag == null)
            {
                return NotFound();
            }

            patch.Patch(Tag);

            try
            {
                db.SaveChanges();
            }
            catch (DbUpdateConcurrencyException)
            {
                if (!TagExists(key))
                {
                    return NotFound();
                }
                else
                {
                    throw;
                }
            }

            return Updated(Tag);
        }

        // DELETE: odata/Tags(5)
        public IHttpActionResult Delete([FromODataUri] int key)
        {
            Tag Tag = db.Tags.Find(key);
            if (Tag == null)
            {
                return NotFound();
            }

            db.Tags.Remove(Tag);
            db.SaveChanges();

            return StatusCode(HttpStatusCode.NoContent);
        }

        // GET: odata/Tags(5)/MataPelajaran


        [EnableQuery]
        public SingleResult<MataPelajaran> GetMataPelajaran([FromODataUri] int key)
        {
            return SingleResult.Create(db.Tags.Where(m => m.Tag_Id == key).Select(m => m.MataPelajaran));
        }
```

```
        [EnableQuery]
        public SingleResult<Nilai> GetNilaiSiswa([FromODataUri] string key)
        {
            return SingleResult.Create(db.Nilais.Where(nilai => nilai.Siswa_Id == key));
        }

        //public async Task<DataSiswa> GetDataSiswa(string siswa_id)
        //{
        //    return await Task.Run(() => db.DataSiswas.Where(a => a.Siswa_Id == siswa_id
).SingleOrDefault());
        //}

        // GET: odata/DataSiswas(5)
        [EnableQuery]
        public SingleResult<DataSiswa> GetDataSiswa([FromODataUri] string key)
        {
            return SingleResult.Create(db.DataSiswas.Where((dataSiswa => dataSiswa.Siswa_Id == key)));
        }

        // GET: odata/Tags(5)/Soals
        [EnableQuery]
        public IQueryable<Soal> GetSoals([FromODataUri] int key)
        {
            return db.Tags.Where(m => m.Tag_Id == key).SelectMany(m => m.Soals);
        }

        protected override void Dispose(bool disposing)
        {
            if (disposing)
            {
                db.Dispose();
            }
            base.Dispose(disposing);
        }

        private bool TagExists(int key)
        {
            return db.Tags.Count(e => e.Tag_Id == key) > 0;
        }
    }
}
```

Source Code API_UjiansController.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Linq;
using System.Net;

using System.Net.Http;
using System.Web.Http;
using System.Web.Http.Description;
using FinalSkripsi.Models;
```

```
using System.Web.Http.OData;

namespace FinalSkripsi.Controllers
{
    public class API_UjiansController : ApiController
    {
        private examdbEntities db = new examdbEntities();

        // GET: api/API_Ujians
        [EnableQuery]
        [Route("api/api_ujians")]
        public IQueryable<Ujian> GetUjians()
        {
            return db.Ujians;
        }

        // GET: api/API_Ujians/5
        [ResponseType(typeof(Ujian))]
        public IHttpActionResult GetUjian(int id)
        {
            Ujian ujian = db.Ujians.Find(id);
            if (ujian == null)
            {
                return NotFound();
            }

            return Ok(ujian);
        }

        // PUT: api/API_Ujians/5
        [ResponseType(typeof(void))]
        public IHttpActionResult PutUjian(int id, Ujian ujian)
        {
            if (!ModelState.IsValid)
            {
                return BadRequest(ModelState);
            }

            if (id != ujian.Ujian_Id)
            {
                return BadRequest();
            }

            db.Entry(ujian).State = EntityState.Modified;

            try
            {
                db.SaveChanges();
            }
            catch (DbUpdateConcurrencyException)
            {
                if (!UjianExists(id))
                {
                    return NotFound();
                }
```

```
            else
            {
                throw;
            }
        }

        return StatusCode(HttpStatusCode.NoContent);
    }

    // POST: api/API_Ujians
    [EnableQuery]
    [Route("api/api_ujians")]
    [ResponseType(typeof(Ujian))]
    public IHttpActionResult PostUjian(Ujian ujian)
    {
        if (!ModelState.IsValid)
        {
            return BadRequest(ModelState);
        }

        db.Ujians.Add(ujian);
        db.SaveChanges();

        return CreatedAtRoute("DefaultApi", new { id = ujian.Ujian_Id }, ujian);
    }

    // DELETE: api/API_Ujians/5
    [ResponseType(typeof(Ujian))]
    public IHttpActionResult DeleteUjian(int id)
    {
        Ujian ujian = db.Ujians.Find(id);
        if (ujian == null)
        {
            return NotFound();
        }

        db.Ujians.Remove(ujian);
        db.SaveChanges();

        return Ok(ujian);
    }

    protected override void Dispose(bool disposing)
    {
        if (disposing)
        {
            db.Dispose();
        }
        base.Dispose(disposing);
    }

    private bool UjianExists(int id)
    {
        return db.Ujians.Count(e => e.Ujian_Id == id) > 0;
    }
}
```

```
}
```

Source Code API ODATA UjiansController.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.Entity;
using System.Data.Entity.Infrastructure;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using System.Web.Http.ModelBinding;
using System.Web.Http.OData;
using System.Web.Http.OData.Routing;
using FinalSkripsi.Models;

namespace FinalSkripsi.Controllers
{
    public class UjiansController : ODataController
    {
        private examdbEntities db = new examdbEntities();

        // GET: odata/Ujians
        [EnableQuery]
        public IQueryable<Ujian> GetUjians()
        {
            return db.Ujians;
        }

        // GET: odata/Ujians(5)
        [EnableQuery]
        public SingleResult<Ujian> GetUjian([FromODataUri] int key)
        {
            return SingleResult.Create(db.Ujians.Where(ujian => ujian.Ujian_Id == key));
        }

        // PUT: odata/Ujians(5)
        public IHttpActionResult Put([FromODataUri] int key, Delta<Ujian> patch)
        {
            Validate(patch.GetEntity());

            if (!ModelState.IsValid)
            {
                return BadRequest(ModelState);
            }

            Ujian ujian = db.Ujians.Find(key);
            if (ujian == null)
            {
                return NotFound();
            }

            patch.Put(ujian);
```

```
        try
        {
            db.SaveChanges();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!UjianExists(key))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }

        return Updated(ujian);
    }

    // POST: odata/Ujians
    public IHttpActionResult Post(Ujian ujian)
    {
        if (!ModelState.IsValid)
        {
            return BadRequest(ModelState);
        }

        db.Ujians.Add(ujian);
        db.SaveChanges();

        return Created(ujian);
    }

    // PATCH: odata/Ujians(5)
    [AcceptVerbs("PATCH", "MERGE")]
    public IHttpActionResult Patch([FromODataUri] int key, Delta<Ujian> patch)
    {
        Validate(patch.GetEntity());

        if (!ModelState.IsValid)
        {
            return BadRequest(ModelState);
        }

        Ujian ujian = db.Ujians.Find(key);
        if (ujian == null)
        {
            return NotFound();
        }

        patch.Patch(ujian);

        try
```

```
            {
                db.SaveChanges();
            }
            catch (DbUpdateConcurrencyException)
            {
                if (!UjianExists(key))
                {
                    return NotFound();
                }
                else
                {
                    throw;
                }
            }

            return Updated(ujian);
        }

        // DELETE: odata/Ujians(5)
        public IHttpActionResult Delete([FromODataUri] int key)
        {
            Ujian ujian = db.Ujians.Find(key);
            if (ujian == null)
            {
                return NotFound();
            }

            db.Ujians.Remove(ujian);
            db.SaveChanges();

            return StatusCode(HttpStatusCode.NoContent);
        }

        // GET: odata/Ujians(5)/DaTaguru
        [EnableQuery]
        public SingleResult<DaTaguru> GetDaTaguru([FromODataUri] int key)
        {
            return SingleResult.Create(db.Ujians.Where(m => m.Ujian_Id == key).Select(m => m.DaTaguru));
        }

        // GET: odata/Ujians(5)/MataPelajaran
        [EnableQuery]
        public SingleResult<MataPelajaran> GetMataPelajaran([FromODataUri] int key)
        {
            return SingleResult.Create(db.Ujians.Where(m => m.Ujian_Id == key).Select(m =>
m.MataPelajaran));
        }

        protected override void Dispose(bool disposing)
        {

            if (disposing)
            {
                db.Dispose();
            }
            base.Dispose(disposing);
```

```
        }

        private bool UjianExists(int key)

    {
            return db.Ujians.Count(e => e.Ujian_Id == key) > 0;
        }
    }
}
```

Source Code WebApiConfig.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net.Http;
using System.Web.Http;
using Microsoft.Owin.Security.OAuth;
using Newtonsoft.Json.Serialization;
using System.Net.Http.Headers;
using System.Web.Http.Routing;
using FinalSkripsi.Models;
using System.Web.Http.OData.Builder;
using System.Web.Http.OData.Extensions;

namespace FinalSkripsi
{
    public static class WebApiConfig
    {
        public static void Register(HttpConfiguration config)
        {
            ODataConventionModelBuilder builder = new ODataConventionModelBuilder();

            //ODATA DaTagurus//
            builder.EntitySet<DaTaguru>("DaTagurus");
            builder.EntitySet<Grup>("Grups");
            builder.EntitySet<Soal>("Soals");
            builder.EntitySet<StandarNilai>("StandarNilais");
            builder.EntitySet<Ujian>("Ujians");
            builder.EntitySet<MataPelajaran>("MataPelajarans");

            //ODATA MataPelajarans//
            builder.EntitySet<MataPelajaran>("MataPelajarans");
            builder.EntitySet<DaTaguru>("DaTagurus");
            builder.EntitySet<Ujian>("Ujians");

            //ODATA Soals//
            builder.EntitySet<Soal>("Soals");
            builder.EntitySet<DaTaguru>("DaTagurus");

            //ODATA StandarNilais
            builder.EntitySet<StandarNilai>("StandarNilais");
            builder.EntitySet<DaTaguru>("DaTagurus");


            //ODATA Ujians//
```

```
            builder.EntitySet<Ujian>("Ujians");
            builder.EntitySet<DaTaguru>("DaTagurus");
            builder.EntitySet<MataPelajaran>("MataPelajarans");

            //ODATA DataSiswas//
            builder.EntitySet<DataSiswa>("DataSiswas");
            builder.EntitySet<Nilai>("Nilais");


            //ODATA Nilais//
            builder.EntitySet<Nilai>("Nilais");
            builder.EntitySet<DataSiswa>("DataSiswas");

            //ODATA Tags//
            builder.EntitySet<Tag>("Tags");
            builder.EntitySet<MataPelajaran>("MataPelajarans");
            builder.EntitySet<Soal>("Soals");


            config.Routes.MapODataRoute("odata", "odata", builder.GetEdmModel());
            config.AddODataQueryFilter();

            //config.EnableQuerySupport();

            config.Formatters.Remove(config.Formatters.XmlFormatter);
            config.Formatters.JsonFormatter.SupportedMediaTypes.Add(new
MediaTypeHeaderValue("application/jason"));
            // Web api configuration and services
            // Configure Web api to use only bearer token authentication.
            config.SuppressDefaultHostAuthentication();
            config.Filters.Add(new HostAuthenticationFilter(OAuthDefaults.AuthenticationType));

            // define route
            IHttpRoute defaultRoute = config.Routes.CreateRoute("api/{Controller}/{id}", new { id =
RouteParameter.Optional }, null);

            // Add route
            config.Routes.Add("DefaultApi", defaultRoute);

            // WebAPI_Routes - AspNetUser
            config.Routes.MapHttpRoute(
            name: "AspNetUsers",
            routeTemplate: "api/aspnetusers/{id}",
            defaults: new { Controller = "aspnetusers", id = RouteParameter.Optional },
            constraints: new { id = "/d+" }
        );

            // WebAPI_Routes - DaTaguru
            config.Routes.MapHttpRoute(
            name: "API_DaTagurus",
            routeTemplate: "api/daTagurus/{id}",
            defaults: new { Controller = "daTagurus", id = RouteParameter.Optional },
            constraints: new { id = "/d+" }


                );
```

```
        //WebAPI_Routes - DataSiswa
        config.Routes.MapHttpRoute(
        name: "API_DataSiswas",
        routeTemplate: "api/api_datasiswas/{id}",
        defaults: new { Controller = "api_datasiswas", id = RouteParameter.Optional },
        constraints: new { id = "/d+" }
    );

        //WebAPI_Routes - MataPelajaran
        config.Routes.MapHttpRoute(
        name: "API_MataPelajarans",
        routeTemplate: "api/matapelajarans/{id}",
        defaults: new { Controller = "matapelajarans", id = RouteParameter.Optional },
        constraints: new { id = "/d+" }
    );

        //WebAPI_Routes - Soal
        config.Routes.MapHttpRoute(
        name: "API_Soals",
        routeTemplate: "api/api_soals/{id}",
        defaults: new { Controller = "api_soals", id = RouteParameter.Optional },
        constraints: new { id = "/d+" }
    );

        //WebAPI_Routes - Standar Nilai
        config.Routes.MapHttpRoute(
        name: "API_StandarNilais",
        routeTemplate: "api/api_standarnilais/{id}",
        defaults: new { Controller = "api_standarnilais", id = RouteParameter.Optional },
        constraints: new { id = "/d+" }
    );

        //WebAPI_Routes - Ujian
        config.Routes.MapHttpRoute(
        name: "API_Ujians",
        routeTemplate: "api/api_ujians/{id}",
        defaults: new { Controller = "api_ujians", id = RouteParameter.Optional },
        constraints: new { id = "/d+" }
    );

        //WebAPI_Routes - Nilai
        config.Routes.MapHttpRoute(
        name: "API_Nilais",
        routeTemplate: "api/api_nilais/{id}",
        defaults: new { Controller = "api_nilais", id = RouteParameter.Optional },
        constraints: new { id = "/d+" }
    );

        //WebAPI_Routes - Tag
        config.Routes.MapHttpRoute(
        name: "API_Tags",

        routeTemplate: "api/api_Tags/{id}",
        defaults: new { Controller = "api_Tags", id = RouteParameter.Optional },
        constraints: new { id = "/d+" }
    );
```

```
        }
    }
}
```

Source Code Global.asax

```
using Newtonsoft.Json.Serialization;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Http;
using System.Web.Mvc;
using System.Web.Optimization;
using System.Web.Routing;

namespace FinalSkripsi
{
    public class WebApiApplication : System.Web.HttpApplication
    {
        protected void Application_Start()
        {
            AreaRegistration.RegisterAllAreas();
            GlobalConfiguration.Configure(WebApiConfig.Register);
            FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);
            RouteConfig.RegisterRoutes(RouteTable.Routes);
            BundleConfig.RegisterBundles(BundleTable.Bundles);

            GlobalConfiguration.Configuration.IncludeErrorDetailPolicy = IncludeErrorDetailPolicy.Always;

GlobalConfiguration.Configuration.Formatters.JsonFormatter.SerializerSettings.ReferenceLoopHandling =
Newtonsoft.Json.ReferenceLoopHandling.Ignore;

GlobalConfiguration.Configuration.Formatters.Remove(GlobalConfiguration.Configuration.Formatters.XmlF
ormatter);



            HttpConfiguration config = GlobalConfiguration.Configuration;

((DefaultContractResolver)config.Formatters.JsonFormatter.SerializerSettings.ContractResolver).IgnoreSeria
lizableAttribute = true;

            var json = GlobalConfiguration.Configuration.Formatters.JsonFormatter;
            json.SerializerSettings.PreserveReferencesHandling =
Newtonsoft.Json.PreserveReferencesHandling.All; }}}
```

Source Code TagBySiswaController.cs

```
using API.Models;
using FinalSkripsi.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
```

```
using System.Net.Http;



using System.Web.Http;
using System.Web.Http.OData;

namespace FinalSkripsi.Controllers
{
    public class TagBySiswaController : ApiController
    {
        private examdbEntities db = new examdbEntities();

        [Route("api/Tagbysiswa")]
        [HttpGet]
        [EnableQuery]
        public List<TagDataService.TagSiswa> getTagBySiswa()
        {
            var Tag = new List<TagDataService.TagSiswa>();
            Tag = (from ds in db.DataSiswas
                        join n in db.Nilais on ds.Siswa_Id equals n.Siswa_Id
                        join eu in db.EventUjians on n.EventUjian_Id equals eu.EventUjian_Id
                        join s in db.Soals on eu.Soal_Id equals s.Soal_Id
                        join t in db.Tags on s.Tag_Id equals t.Tag_Id
                        select new TagDataService.TagSiswa
                        {
                            Tag_Id = t.Tag_Id,
                            Tag1 = t.Tag1,
                            Nama_Siswa = ds.Nama_Siswa,
                            Siswa_Id = ds.Siswa_Id
                        }).ToList();

            return Tag;
        }
    }
}
```

Source Code NilaiTagBySiswaController.cs

```
using API.Models;
using FinalSkripsi.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;

using System.Web.Http.OData;

namespace FinalSkripsi.Controllers
{
    public class NilaiTagBySiswaController : ApiController
    {
        private examdbEntities db = new examdbEntities();
```

```
        [Route("api/nilaiTagbysiswa")]
        [HttpGet]

    [EnableQuery]

     public List<TagDataService.NilaiTagPerSiswa> getNilaiTagBySiswa()
          {
            var nilaiTagbysiswa = new List<TagDataService.NilaiTagPerSiswa>();
            nilaiTagbysiswa = (from ds in db.DataSiswas
                               join n in db.Nilais on ds.Siswa_Id equals n.Siswa_Id
                               join eu in db.EventUjians on n.EventUjian_Id equals eu.EventUjian_Id
                               join s in db.Soals on eu.Soal_Id equals s.Soal_Id
                               join t in db.Tags on s.Tag_Id equals t.Tag_Id
                               select new TagDataService.NilaiTagPerSiswa
                               {
                                  Tag_Id = t.Tag_Id,
                                  Tag1 = t.Tag1,
                                  Nama_Siswa = ds.Nama_Siswa,
                                  Siswa_Id = ds.Siswa_Id,
                                  Nilai1 = n.Nilai1,
                                  Kelas = ds.Kelas
                               }).ToList();

            return nilaiTagbysiswa; }}}
```