



Vlaanderen
is landbouw & visserij

TECHNICAL DOCUMENTATION

Web API

smartdots

ILVO

Institute for Agricultural
and Fisheries Research

www.ilvo.vlaanderen.be

<i>Date</i>	<i>Version</i>
2017/07	Version 1.0
2017/10	Version 2.0
2017/12	Version 3.0
2018/06	Version 4.0
2018/10	Version 5.0
2019/05	Version 6.0
2020/07	Version 7.0
2020/09	Version 8.0
2021/02	Version 9.0
2023/02	Version 10.0
2024/11	Version 11.0

Contact:
Wim Allegaert, wim.allegaert@ilvo.vlaanderen.be
Kevin De Coster, kevin.decoester@ilvo.vlaanderen.be

Institute for Agricultural and Fisheries Research
Animal Sciences Unit - Fisheries and Aquatic Production
Jakobsenstraat 1, B-8400 Oostende, Belgium
Tel + 32 59 34 22 50

<http://www.ilvo.vlaanderen.be>



Table of contents

1. Introduction.....	4
2. Basic concepts	5
2.1. Requests	5
2.2. Responses	5
2.3. Example	5
3. Data Transfer Objects	6
3.1. DTO classes	6
3.2. Dynamic objects	11
4. Methods.....	12
4.1. Screen 1: Authentication.....	12
4.2. Screen 2: Analyses	14
4.3. Screen 3: Age reading.....	18

1. Introduction

SmartDots is a software application for performing age determination analysis of fish.

To determine the age of the fish end-users make annotations on otolith images. Each annotation contains a line and one or multiple dots. The number of dots is equal to the age. In addition an annotation can contain a parameter, a quality indication and some comment.

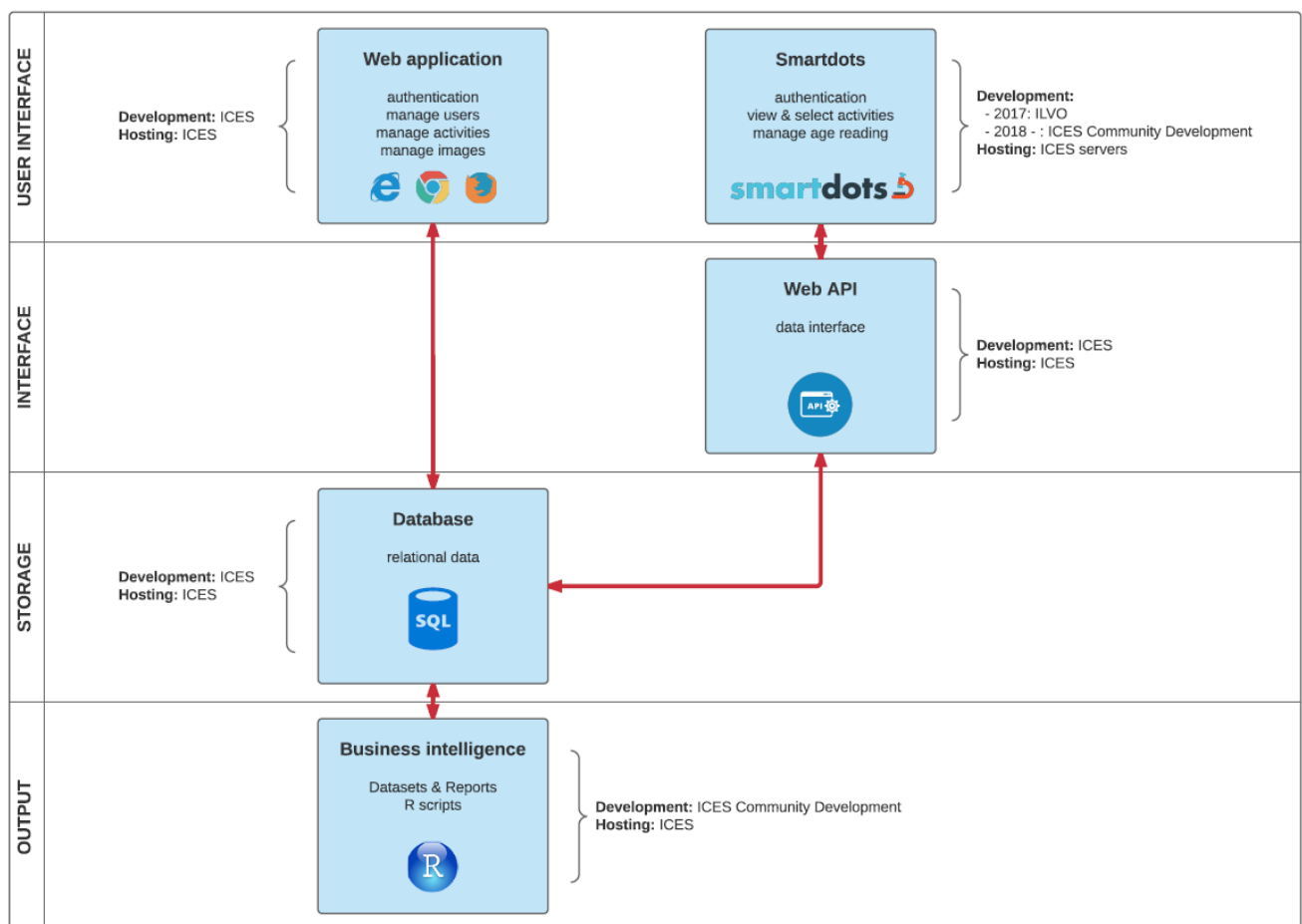
The SmartDots user interface exists out of 3 main screens:

- Screen 1: Authentication
- Screen 2: Analyses
- Screen 3: Age reading

SmartDots is developed as a client-server application. SmartDots (the client) is a Windows-based application (Windows Presentation Foundation). SmartDots exchange messages with a Web API (the server) in a request-response messaging pattern. The Web API accesses the database.

The Web API can be written in any technology. Communication between the Web API and the software happens through JSON-objects. The implementation in this manual shows how it is done in ASP.NET Core. There's also a swagger page for the latest version: <https://webapi.smartfisheries.be/swagger/>

Notice the different naming conventions for property-names between .NET and json (PascalCase and camelCase respectively). For SmartDots it doesn't matter.



2. Basic concepts

2.1. Requests

Web API methods used in SmartDots are either POST or GET (PUT and DELETE are not used). POST-requests are used when non-primitive-type objects are sent to the Web API. Examples of these objects are Lists and Complex types.

2.2. Responses

Every successful Web API response returns a custom 'WebApiResponse' object. The response is in JSON-format. This is the way the SmartDots application expects it.

The WebApiResponse Class looks like this:

```
public class WebApiResponse
{
    public dynamic Result { get; set; }
    public string ErrorMessage { get; set; }
    public string WarningMessage { get; set; }
}
```

Where the Result property contains the desired output. This is a dynamic property which means it can contain anything. If an error occurred, for example user permissions, or a code exception, the error can be put in the ErrorMessage property. If SmartDots receives a WebApiResponse, it knows how to handle it.

2.3. Example

```
[HttpGet]
[Route("getreadabilityqualities")]
//Returns a List<Quality> of AQ quality codes
public string GetReadabilityQualities(string token)
{
    var webApiResponse = new WebApiResponse();
    if (!HasPermission(token))
    {
        webApiResponse.ErrorMessage = "User has no permission to get Qualities!";
        return JsonConvert.SerializeObject(webApiResponse);
    }
    try
    {
        using (var ctx = new D1_SmartLabContext())
        {
            var qualities = ctx.Quality.ToList();

            webApiResponse.Result = sdQualities;
            return JsonConvert.SerializeObject(webApiResponse);
        }
    }
    catch (Exception e)
    {
        webApiResponse.ErrorMessage = e.Message;
        return JsonConvert.SerializeObject(webApiResponse);
    }
}
```

3. Data Transfer Objects

The data between SmartDots and the Web API is exchanged via Data Transfer Objects (DTO). DTO objects are used to aggregate the data and reduce the number of requests. The DTO objects are transferred via the WebApiResponse's Result property.

3.1. DTO classes

3.1.1. DTO UserAuthentication

```
// Class contains login data
public class DtoUserAuthentication
{
    public string Username { get; set; }
    public string Password { get; set; }
    public DtoAuthenticationMethod AuthenticationMethod { get; set; }
}
```

3.1.2. DTO AuthenticationMethod

```
// Enum contains authentication method data
public enum DtoAuthenticationMethod { Windows, Basic, Token }
```

3.1.3. DTO User

```
// Class contains user and security data
public class DtoUser
{
    public Guid Id { get; set; }
    public string AccountName { get; set; }
    public string Token { get; set; }
}
```

3.1.4. DTO SmartDotsSettings

```
// Class contains SmartDots configuration settings
public class DtoSmartDotsSettings
{
    public bool CanAttachDetachSample { get; set; }
    public bool CanBrowseFolder { get; set; }
    public bool UseSampleStatus { get; set; }
    public bool CanApproveAnnotation { get; set; }
    public bool RequireAQ1ForApproval { get; set; }
    public bool RequireAQForApproval { get; set; }
    public bool RequireParamForApproval { get; set; }
    public bool AutoMeasureScale { get; set; }
    public bool ScanFolder { get; set; }
    public bool AnnotateWithoutSample { get; set; }
    public bool OpenSocket { get; set; }
    public string EventAlias { get; set; }
    public string SampleAlias { get; set; }
    public bool CanMarkEventAsCompleted { get; set; }
    public bool AllowMultipleApprovements { get; set; }
    public bool IgnoreMultiUserColor { get; set; }
    public float MinRequiredVersion { get; set; }
}
```

Note:

- **CanAttachDetachSample:** Enable sample attach & detach button in file window. This allows the user to remove the SampleId from the DtoFile or add a new one (Used at ILVO)
- **CanBrowseFolder:** Enable browse button in file window. Only set this 'true' when you allow the user to select pictures from the file system.
- **UseSampleStatus:** Show linked sample status in file window. (First column, colored box)
- **CanApproveAnnotation:** Show property approve in annotation grid & edit window
- **RequireAQ1ForApproval:** Enable quality control: approval only possible for annotations with 'AQ1' quality code. Please make sure there is an 'AQ1' ReadabilityQuality quality code as the software searches for this string.
- **RequireAQForApproval:** Enable quality control: approval only possible for annotation with a AQ code.
- **RequireParamForApproval:** Enable quality control: approval only possible for annotations with a parameter.
- **AutoMeasureScale:** Automatically try to measure the scale as soon as a file with no scale is loaded. If set to false, the user will have to click the Measure scale button manually.
- **ScanFolder:** Searches all filenames in a folder to pass to the Web API. Set to false if you only want to retrieve filenames from the database.
- **AnnotateWithoutSample:** Lets the user make annotations on files with no linked sample.
- **OpenSocket:** Opens a socket that listens on 127.0.0.1 on port 11000 to incoming commands. Used at ILVO to communicate with other software. Set to false if SmartDots does not rely on other software.
- **EventAlias:** The name of the objects loaded in the 2nd screen with GetAnalysisDynamic Web API call. It will be displayed in the title bar.
- **SampleAlias:** An alternative name for 'Sample' will be show throughout the application. Leave empty to keep the default 'Sample'.
- **CanMarkEventAsCompleted:** When set to true, this will show the Finish/Reopen button in the 2nd screen.
- **AllowMultipleApprovements:** When set to true, this allows the user to approve multiple annotations in the Annotation panel.
- **IgnoreMultiUserColor:** When set to true, all dots will keep their original color when selecting multiple annotations.
- **MinRequiredVersion:** Used to check if the version is still compatible. Incompatible versions will prompt the user to update. The latest version is 4.1.

7

3.1.5. DTO ReadabilityQuality

```
// Class contains vocabulary with readability quality data
public class DtoReadabilityQuality
{
    public Guid Id { get; set; }
    public string Code { get; set; }
    public string Description { get; set; }
    public string Color { get; set; } // A hex color eg: #CC0000
}
```

Example data used at ILVO:

Id	Code	Description	Color
0BA0FC71-3EC4-424F-A6BA-A158FE8E3157	AQ1	Easy to age with high precision.	#00b300
371089A1-A646-4E80-ACC8-B3C6FB716900	AQ2	Difficult to age with age with acceptable precision.	#ff8000
3BC001ED-2994-457D-92C6-F35163A6A654	AQ3	Unreadable or very difficult to age with acceptable precision.	#cc0000

3.1.6. DTO Analysis

```
// Class contains analysis data (which is used on SmartDots screen "agereading")
public class DtoAnalysis
{
    public Guid Id { get; set; }
}
```

```

    public int Number { get; set; }
    public DtoFolder Folder { get; set; }
    public List<DtoAnalysisParameter> AnalysisParameters { get; set; }
    public string HeaderInfo { get; set; } // Used to display in the title-bar
    public bool UserCanPin { get; set; } // Allows the user to pin annotations
    public bool ShowNucleusColumn { get; set; } // Displays the Nucleus column for annotations
    public bool ShowEdgeColumn { get; set; } // Displays the Edge column for annotations
    public bool UseGrowthGraphAllSamples { get; set; } // Use the new features of the growth graph*
}

```

* If set to true, more modes will be available in the growth graph to display data. The other modes allow growth data to be compared over multiple samples. These however require that annotation data for all files/samples is loaded in the *getfiles* api call

3.1.7. DTO AnalysisParameter

```

// Class contains vocabulary with analysis parameter data
public class DtoAnalysisParameter
{
    public Guid Id { get; set; }
    public string Code { get; set; }
    public string Description { get; set; }
}

```

Example data used at ILVO:

Id	Code	Description
1376FE08-7C9C-4270-8665-B217818D4351	OWR	Age in years, method: Otolith Winter Rings
6901B4BF-F556-4DEA-A79C-BBB5FDA9603C	OSR	Age in years, method: Otolith Summer Rings
25EFF9A6-D671-4869-AF19-DD313BDFDB59	ODR	Age in days, method: Otolith Day Rings

3.1.8. DTO Folder

```

// Class contains folder path
public class DtoFolder
{
    public Guid Id { get; set; }
    public string Path { get; set; }
}

```

3.1.9. DTO File

```

// Class contains file data, including related annotation and sample data
public class DtoFile
{
    public Guid Id { get; set; }
    public string Filename { get; set; } // Used to retrieve the file
    public string DisplayName { get; set; } // Used to display an alternate filename, if left empty,
the Filename will be displayed
    public string Path { get; set; } // Optional, the full path of the file, when provided this
takes priority over Folder.Path + "/" + Filename
    public string SampleNumber { get; set; }
    public Guid? SampleId { get; set; }
    public int AnnotationCount { get; set; }
    public bool IsReadOnly { get; set; }
    public bool CanApprove { get; set; } // Requested by WMR
    public decimal? Scale { get; set; }
    public List<DtoAnnotation> Annotations { get; set; }
    public DtoSample Sample { get; set; }
}

```



```

    public Dictionary<string, object> SampleProperties { get; set; }
}

```

3.1.10. DTO Sample

```

// Class contains dynamic sample data
public class DtoSample
{
    public Guid Id { get; set; }
    public string StatusCode { get; set; }
    public string StatusColor { get; set; }
    public int StatusRank { get; set; }
    public Dictionary<string, string> DisplayedProperties { get; set; }
}

```

3.1.11. DTO Annotation

```

// Class contains annotation data, including related line and dot data
public class DtoAnnotation
{
    public Guid Id { get; set; }
    public Guid? ParameterId { get; set; }
    public Guid FileId { get; set; }
    public Guid? SampleId { get; set; }
    public Guid? AnalysisId { get; set; }
    public Guid? QualityId { get; set; }
    public DateTime DateCreation { get; set; }
    public Guid? LabTechnicianId { get; set; }
    public string LabTechnician { get; set; }
    public int Result { get; set; }
    public bool IsApproved { get; set; }
    public bool IsReadOnly { get; set; }
    public bool IsFixed { get; set; } // A pinned aka fixed annotation. When users start a new
    // annotation, the Line of this annotation will be used. Make sure only 1 annotation with IsFixed =
    // true is fetched from the database for the selected image.
    public string Nucleus { get; set; } // "Opaque"/"Translucent"/null
    public string Edge { get; set; } // "Opaque"/"Translucent"/null
    public string Comment { get; set; }
    public List<DtoDot> Dots { get; set; }
    public List<DtoLine> Lines { get; set; }
}

```

9

3.1.12. DTO Line

```

// Class contains line data
public class DtoLine
{
    public Guid Id { get; set; }
    public Guid AnnotationId { get; set; }
    public int X1 { get; set; }
    public int Y1 { get; set; }
    public int X2 { get; set; }
    public int Y2 { get; set; }
    public int Width { get; set; }
    public string Color { get; set; } // Hexcolor
    public int LineIndex { get; set; }
}

```

3.1.13. DTO Dot

```

// Class contains dot data
public class DtoDot
{
    public Guid Id { get; set; }
}

```

```
public Guid AnnotationId { get; set; }
public int X { get; set; }
public int Y { get; set; }
public int Width { get; set; }
public int DotIndex { get; set; }
public string Color { get; set; } // Also Hexcolor
public string DotShape { get; set; }
public string DotType { get; set; }
}
```

3.2. Dynamic objects

3.2.1. Analysis

In the Analysis Overview screen, a List<dynamic> is displayed. The institute can decide how these objects look. It should contain all relevant information about an analysis. The only requirement is that these objects have an ID and a Folder property. Please make sure no property is a complex type as it's ToString() will not show any convenient information. The ID-property will not be displayed in the Grid.

3.2.1.1. Example

```
var webApiResponse = new WebApiResponse();

var dtoAnalyses = new List<object>();

dynamic dtoAnalysis = new ExpandoObject();
dtoAnalysis.ID = Guid.NewGuid();
dtoAnalysis.Number = 2;
dtoAnalysis.Service = "SRV01";
dtoAnalysis.SampleSet = "SampleSet02";
dtoAnalysis.Folder = "\\someserverpath\\somefolder\\"; // Or "http://somefolderlocation/"

dtoAnalyses.Add(dtoAnalysis);

webApiResponse.Result = dtoAnalyses;

return JsonConvert.SerializeObject(webApiResponse);
```

4. Methods

4.1. Screen 1: Authentication

In order to call Web API methods, the Web API url has to be set in the first screen. An example on a local machine looks like this: <http://localhost:63216/api/SmartDots/>.



This screen is used for Authentication only. A `DtoUserAuthentication` object is created with the user input and is then used to send to the Web API.

4.1.1. GetGuestToken

4.1.1.1. Definition

Returns token (string) for the user to login with

URL	/getguesttoken
Method	GET
URL Params	
Post Params	
Response	(JSON) WebApiResponse with a token (string) as the Result property

4.1.1.2. ASP.NET implementation example

```
[HttpGet]
[Route("getguesttoken")]
public ActionResult GetGuestToken()
{
    //Implementation logic
}
```

4.1.1.3. Example Response

```
{"Result": "ojjkjchc15464", "ErrorMessage": null, "WarningMessage": null}
```

4.1.2. Authenticate

4.1.2.1. Definition

Returns a DtoUser object if authentication is succeeded.

URL	/authenticate
Method	POST
URL Params	
Post Params	DtoUserAuthentication userauthentication
Response	(JSON) WebApiResponse with a DtoUser-object as the Result property

4.1.2.2. ASP.NET implementation example

```
[HttpPost]
[Route("authenticate")]
public ActionResult Authenticate([FromBody] DtoUserAuthentication userauthentication)
{
    //Implementation logic
}
```

4.1.2.3. Example Response

```
{"Result": {"Id": "6fd67e04-eadf-43bf-b7ee-e397da490972", "AccountName": "kdecoster", "Token": "ojjkjchc15464"}, "ErrorMessage": null, "WarningMessage": null }
```

13

4.1.3. GetSettings

4.1.3.1. Definition

Returns a DtoSmartDotsSettings object which contains the institute depending settings. The DtoSmartDotsSettings objects is used in SmartDots to enable/disable certain features.

URL	/getsettings
Method	GET
URL Params	string token
Post Params	
Response	(JSON) WebApiResponse with a DtoSmartDotsSettings-object as the Result property

4.1.3.2. ASP.NET implementation example

```
[HttpGet]
[Route("getsettings")]
public ActionResult GetSettings(string token)
{
    //Implementation logic
}
```

4.1.3.3. Example Response

```
{"Result":{"CanAttachDetachSample":true,"CanBrowseFolder":true,"UseSampleStatus":true,"CanApproveAnnotation":true,"RequireAQ1ForApproval":true},"ErrorMessage":null,"WarningMessage":null}
```

4.1.4. GetReadabilityQualities

4.1.4.1. Definition

Returns a List<DtoReadabilityQuality> which contains the institute depending settings.

URL	/getreadabilityqualities
Method	GET
URL Params	string token
Post Params	
Response	(JSON) WebApiResult with a List<DtoReadabilityQuality> as the Result property

4.1.4.2. ASP.NET implementation example

```
[HttpGet]
[Route("getreadabilityqualities")]
public ActionResult GetReadabilityQualities(string token)
{
    //Implementation logic
}
```

4.1.4.3. Example Response

```
{"Result":[{"Id":"0ba0fc71-3ec4-424f-a6ba-a158fe8e3157","Code":"AQ1","Description":"Easy to age with high precision.", "Color":"#00b300"}, {"Id":"371089a1-a646-4e80-acc8-b3c6fb716900", "Code":"AQ2", "Description":"Difficult to age with age with acceptable precision.", "Color":"#ff8000"}, {"Id":"3bc001ed-2994-457d-92c6-f35163a6a654", "Code":"AQ3", "Description":"Unreadable or very difficult to age with acceptable precision.", "Color":"#cc0000"}], "ErrorMessage":null, "WarningMessage":null}
```

4.2. Screen 2: Analyses

This screen displays a list of dynamic objects. It should contain all relevant information about an analysis. The institute can decide how these objects look. The only requirement is that these objects have an ID and a Folder property.

Please make sure no property is a complex type as its ToString() will not show any convenient information. The ID-property will not be displayed in the Grid.

smartdots Event overview							
Number	Service	Sample Set	Method	Sample State	Date Due	Folder	Status
20170001	Workshop - Scopthal...	TUR - 27.4.C	S&S	PHO	2/12/2017 12:00:00	http://193.190.112.136/...	TODO
20170002	Workshop - Pleuronec...	PLE - 27.7.F	WHL	PHO	2/12/2017 12:00:00	http://193.190.112.136/...	TODO
20170003	Training - Sardina pilc...	PIL - 37.1.1	WHL	PHO	2/12/2017 12:00:00	http://193.190.112.136/...	TODO
20170004	Training - Microstornu...	LEM - 27.4.C	WHL	PHO	2/12/2017 12:00:00	http://193.190.112.136/...	TODO
20170013	Workshop - Pagellus e...	PAC	WHL	PHO	29/12/2017 8:07:54	http://193.190.112.136/...	TODO
20170005	Workshop - Scopthal...	BLL - 27.4.C	S&S	PHO	2/12/2017 12:00:00	http://193.190.112.136/...	TODO
20170006	Workshop - Gadus mo...	COD - 27.7.G	S	PHO	11/12/2017 12:00:00	http://193.190.112.136/...	TODO
20170007	Exchange - Merlangius...	WHG - 27.4	WHL	PHO	11/12/2017 12:00:00	http://193.190.112.136/...	TODO
20170008	Training - Lepidorhom...	MEG - 27.7.G	WHL	PHO	11/12/2017 12:00:00	http://193.190.112.136/...	TODO
20170009	Workshop - Clupea ha...	HER - 27.3.A - 27.4.B	WHL	PHO	11/12/2017 12:00:00	http://193.190.112.136/...	TODO

4.2.1. GetAnalysesDynamic

4.2.1.1. Definition

Returns a List<dynamic>

15

URL	/getanalysesdynamic
Method	GET
URL Params	string token, int number (optional)
Post Params	
Response	(JSON) WebApiResult with a List<dynamic> as the Result property

4.2.1.2. ASP.NET implementation example:

```
[HttpGet]
[Route("getanalysesdynamic")]
public ActionResult GetAnalysesDynamic(string token, int? number = null)
{
    //Implementation logic
}
```

Note:

If a number is provided in the request, a single analysis will be returned to the user.

Example Response:

```
{
  "Result": [
    {
      "ID": "97cf2113-8b31-414a-be96-50db0b729616",
      "Number": "20170073",
      "Service": "TEST",
      "SampleSet": "TEST",
      "Method": "WHL",
      "SampleState": "PHO",
      "DateDue": "2017-06-30T16:01:20",
      "Folder": null,
      "Status": "TODO",
      "CountSample": 0
    },
    {
      "ID": "f938b6ec-c722-4b45-a568-dc8ce2a38909",
      "Number": "20170067",
      "Service": "IBTS deel II 26-55",
      "SampleSet": "IBTS st 32",
      "Method": "WHL",
      "SampleState": "PHO",
      "DateDue": "2017-06-28T14:48:50",
      "Folder": "\\c\\clo.be\\dfs\\Data\\data_d1\\pictures\\0t1\\2017\\NDGP Bilat\\ILVO\\WHG"
    }
  ]
}
```

IBTS Q1 2017\\LF 028
3400032", "Status": "TODO", "CountSample": 11}], "ErrorMessage": null, "WarningMessage": null}

4.2.2. GetAnalysis

4.2.2.1. Definition

Returns a Single DtoAnalysis. The properties of this DtoAnalysis are used in the AgeReading screen.

URL	/getanalysis
Method	GET
URL Params	string token, Guid id
Post Params	
Response	(JSON) WebApiResult with a DtoAnalysis as the Result property

4.2.2.2. ASP.NET implementation example

```
[HttpGet]
[Route("getanalysis")]
public ActionResult GetAnalysis(string token, Guid id)
{
    //Implementation logic
}
```

Note: Please make sure all properties are present when transferring a single analysis. All properties are used in the age reading screen.

4.2.2.3. Example Response for single analysis

```
{"Result":{"Id":"2a79894d-8755-444d-94d8-b82f5fab20dd","Number":20160028,"Folder":{"Id":"21e65ccb-6cc9-4b34-a292-0a22c035af97","Path":"\\\\c\\lo.be\\dfs\\Data\\data_d1\\pictures\\0t1\\2016\\Market\\LF028_PLE_7D_D_M_EK1_BYDR01"},"AnalysisParameters":[{"Id":"1376fe08-7c9c-4270-8665-b217818d4351","Code":"OWR","Description":"Age, method: Otolith Winter Rings"}],"HeaderInfo":"ILV0:20160028 PLE_7D_D_M"},"ErrorMessage":null,"WarningMessage":null}
```

4.2.3. UpdateAnalysisFolder

4.2.3.1. Definition

Updates an Analysis' folder, returns true if succeeded

URL	/updateanalysisfolder
Method	POST
URL Params	string token, string folderpath
Post Params	Guid analysisid
Response	(JSON) WebApiResult with a Boolean if succeeded as the Result property

4.2.3.2. ASP.NET implementation example

```
[HttpPost]
[Route("updateanalysisfolder")]
//Updates a file, returns true if succeeded
public ActionResult UpdateAnalysisFolder(string token, string folderpath, [FromBody] Guid analysisid)
```



```
{
    //Implementation logic
}
```

4.2.3.3. Example Response

```
{"Result":true,"ErrorMessage":null,"WarningMessage":null}
```

4.2.4. ToggleAnalysisUserProgress

4.2.4.1. Definition

Toggles the user's progress for the event/analysis

URL	/toggleanalysisuserprogress
Method	POST
URL Params	string token
Post Params	Guid analysisid
Response	(JSON) WebApiResponse with a Boolean if succeeded as the Result property

4.2.4.2. ASP.NET implementation example

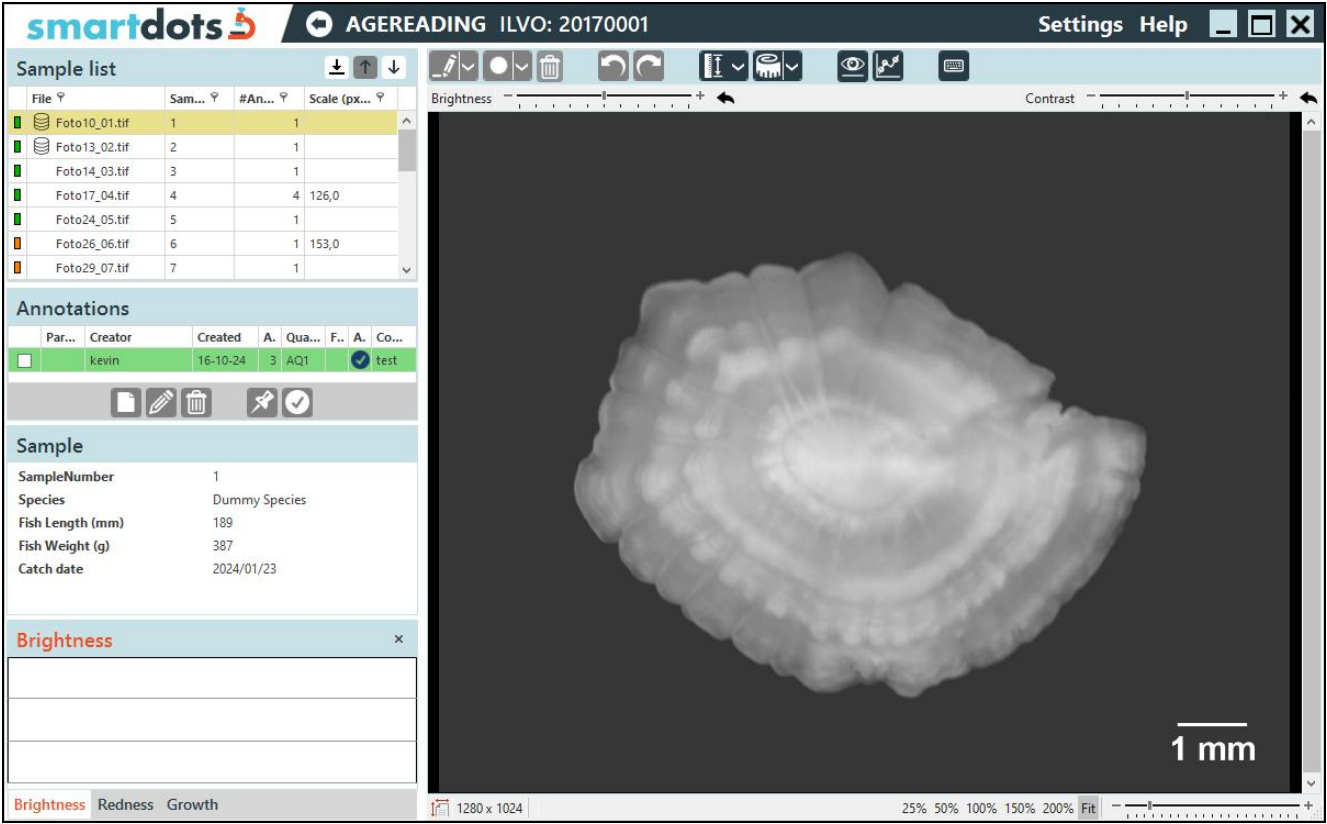
```
[HttpPost]
[Route("toggleanalysisuserprogress")]
public ActionResult ToggleAnalysisUserProgress(string token, [FromBody] Guid analysisid)
{
    //Implementation logic
}
```

17

4.2.4.3. Example Response

```
{"Result":true,"ErrorMessage":null,"WarningMessage":null}
```

4.3. Screen 3: Age reading



4.3.1. GetFiles

4.3.1.1. Definition

Returns a List<DtoFile> for every imagename passed through to the web api. Imagenames come from all images provided in the analysis folder path on filesystem level.

URL	/getfiles
Method	POST
URL Params	string token, Guid analysisid
Post Params	List<string> imagenames
Response	(JSON) WebApiResult with a List<DtoFile> as the Result property

4.3.1.2. ASP.NET implementation example

```
[HttpPost]
[Route("getfiles")]
public ActionResult GetFiles(string token, Guid analysisid, [FromBody]List<string> imagenames)
{
    //Implementation logic
}
```

Note:

- This Api Method is used to fill the list of files. Not all File properties are required.
- It is advised to leave the Annotations null, they are not required here.
 - Unless DtoAnalysis.UseGrowthGraphAllSamples is true, then pass all Annotation data.
- If the institute decides not to work with Sample statuses, the File's Sample can be left null

4.3.1.3. Example Response

```
{ "result": [ { "id": "13357fcf-a535-4d65-9425-788099d08878", "filename": "NS-BTS
01_BSS_4C_scale_1_2019082620577.png", "displayName": null, "sampleNumber": "1", "sampleId":
"48ab8e63-6681-4bc4-ace7-bd12c1ec25ad", "annotationCount": 2, "isReadOnly": false, "canApprove":
false, "scale": 268.000, "annotations": null, "sample": { "id": "48ab8e63-6681-4bc4-ace7-
bd12c1ec25ad", "statusCode": "Completed", "statusColor": "#00b300", "statusRank": 100,
"displayedProperties": null }, "sampleProperties": { "CatchDate": 2019, "IcesDivision": "IVc",
"RandomString": "Hello" } } ], "errorMessage": null,"warningMessage":null}
```

4.3.2. GetFileWithSampleAndAnnotations

4.3.2.1. Definition

Returns a List<DtoFile>

URL	/getfilewithsampleandannotations
Method	GET
URL Params	string token, Guid id, bool withAnnotations, bool withSample
Post Params	
Response	(JSON) WebApiResponse with a DtoFile as the Result property

4.3.2.2. ASP.NET implementation example

```
[HttpGet]
[Route("getfilewithsampleandannotations")]
public ActionResult GetFile(string token, Guid id, bool withAnnotations, bool withSample)
{
    //Implementation logic
}
```

19

Note:

- Load all annotation data for the file if withAnnotations = true.
- Load the file's sample if withSample = true.

4.3.2.3. Example Response

```
{"Result":{"Id":"6dd1f01e-d703-4ebb-afd8-
c26d6a8c43a5","Filename":"BYDR01_PLE_7D_D_M_1.00_59574","DisplayName":"File1","SampleNumber":"59574"
,"SampleId":"48e2aa84-fd52-4e08-a0ad-
203fea6141c8","AnnotationCount":2,"IsReadOnly":true,"Scale":0.0, "Annotations":[{"Id":"b13e2ddf-
e993-4bce-811f-c5a2857c8011","ParameterId":"1376fe08-7c9c-4270-8665-
b217818d4351","FileId":"6dd1f01e-d703-4ebb-afd8-c26d6a8c43a5","QualityId":"371089a1-a646-4e80-acc8-
b3c6fb716900","DateCreation":"2016-04-
07T09:13:34.68","LabTechnicianId":null,"LabTechnician":"imaertens","Result":4,"IsApproved":false,"Is
ReadOnly":false,"Comment":null,"Dots":[{"Id":"8c13910e-bdba-4a0e-9a81-
04abaea61035","AnnotationId":"b13e2ddf-e993-4bce-811f-
c5a2857c8011","X":706,"Y":942,"Width":13,"DotIndex":0,"Color":"#0070C0","DotShape":"FilledCircle","D
otType":null},{ "Id":"1c732a6c-faef-48a9-b277-2141bdef425","AnnotationId":"b13e2ddf-e993-4bce-811f-
c5a2857c8011","X":843,"Y":1026,"Width":13,"DotIndex":0,"Color":"#0070C0","DotShape":"FilledCircle","
DotType":null},{ "Id":"44ae4e30-c750-4c81-a53a-5a882edde8a3","AnnotationId":"b13e2ddf-e993-4bce-811f-
c5a2857c8011","X":635,"Y":898,"Width":13,"DotIndex":0,"Color":"#0070C0","DotShape":"FilledCircle","D
otType":null},{ "Id":"6154640a-5738-40c6-af66-e7904dcb9b05","AnnotationId":"b13e2ddf-e993-4bce-811f-
c5a2857c8011","X":786,"Y":991,"Width":13,"DotIndex":0,"Color":"#0070C0","DotShape":"FilledCircle","D
otType":null}], "Lines":[{"Id":"ee7d268b-5d26-4cd4-8d53-6588e8b86909","AnnotationId":"b13e2ddf-e993-
4bce-811f-
c5a2857c8011","X1":953,"Y1":1093,"X2":635,"Y2":898,"Width":2,"Color":"#18EFEC","LineIndex":0}]],{"Id
":"35542245-bf9f-43ba-810a-dac3c59069fb","ParameterId":"1376fe08-7c9c-4270-8665-
b217818d4351","FileId":"6dd1f01e-d703-4ebb-afd8-c26d6a8c43a5","QualityId":"0ba0fc71-3ec4-424f-a6ba-
```

```

a158fe8e3157", "DateCreation": "2016-02-
23T09:36:54.583", "LabTechnicianId": null, "LabTechnician": "mmoerman", "Result": 4, "IsApproved": true, "IsR
eadOnly": false, "Comment": null, "Dots": [{ "Id": "2b7ddb72-15c9-4e44-8e9e-
eded764ff879", "AnnotationId": "35542245-bf9f-43ba-810a-
dac3c59069fb", "X": 1323, "Y": 1183, "Width": 10, "DotIndex": 0, "Color": "#FFFF00", "DotShape": "FilledCircle",
"DotType": null }, { "Id": "c5b55bf5-5386-4ec4-93a4-9615cdaa3b94", "AnnotationId": "35542245-bf9f-43ba-
810a-
dac3c59069fb", "X": 1216, "Y": 1153, "Width": 10, "DotIndex": 0, "Color": "#FFFF00", "DotShape": "FilledCircle",
"DotType": null }, { "Id": "c2db8dca-7b1b-48e8-81d7-c14890ccecba", "AnnotationId": "35542245-bf9f-43ba-
810a-
dac3c59069fb", "X": 1084, "Y": 1115, "Width": 10, "DotIndex": 0, "Color": "#FFFF0000", "DotShape": "FilledCircle",
"DotType": null }, { "Id": "9c448012-474e-416a-b5bb-2796ad77a56b", "AnnotationId": "35542245-bf9f-43ba-
810a-
dac3c59069fb", "X": 1139, "Y": 1131, "Width": 10, "DotIndex": 0, "Color": "#FFFF00", "DotShape": "FilledCircle",
"DotType": null } ], "Lines": [{ "Id": "263d4e96-ff80-4939-a4ee-21576a71a125", "AnnotationId": "35542245-
bf9f-43ba-810a-
dac3c59069fb", "X1": 937, "Y1": 1074, "X2": 1384, "Y2": 1200, "Width": 2, "Color": "#18EFEC", "LineIndex": 0 } ] }, "
Sample": { "Id": "48e2aa84-fd52-4e08-a0ad-
203fea6141c8", "StatusCode": "Completed", "StatusColor": "#00b300", "StatusRank": 100, "DisplayedProperties
": { "Number": "59574", "Number External": null, "Sample type": null, "Sample state": null, "Catch
date": "12/02/2016", "Ices Division": null, "Length": "235", "Maturity": "2756110C-8242-43AC-A783-
128A625808B8", "Sex": "DF320029-3172-47DC-AB38-D138C1D22B88", "Species": null, "Statistical
Rectangle": null, "Weight": "101.00" } }, "SampleProperties": { "CatchDate": 2019, "IcesDivision": "IVc",
"RandomString": "Hello" } }, "ErrorMessage": null, "WarningMessage": null }

```

4.3.3. UpdateFile

4.3.3.1. Definition

Updates a file, (links a Sample to a file) returns true if succeeded

URL	/updatefile
Method	POST
URL Params	string token
Post Params	DtoFile file
Response	(JSON) WebApiResponse with a Boolean if succeeded as the Result property

4.3.3.2. ASP.NET implementation example

```
[HttpPost]
[Route("updatefile")]
public ActionResult UpdateFile(string token, [FromBody] DtoFile file)
{
    //Implementation logic
}
```

4.3.3.3. Example Response

```
{"Result":true,"ErrorMessage":null,"WarningMessage":null}
```

4.3.4. AddAnnotation

21

4.3.4.1. Definition

Adds an annotation, returns true if succeeded

URL	/addannotation
Method	POST
URL Params	string token
Post Params	DtoAnnotation annotation
Response	(JSON) WebApiResponse with a Boolean if succeeded as the Result property

4.3.4.2. ASP.NET implementation example

```
[HttpPost]
[Route("addannotation")]
public ActionResult AddAnnotation(string token, [FromBody] DtoAnnotation annotation)
{
    //Implementation logic
}
```

4.3.4.3. Example Response

```
{"Result":true,"ErrorMessage":null,"WarningMessage":null}
```

4.3.5. UpdateAnnotations

4.3.5.1. Definition

Updates list of annotations, returns true if succeeded

URL	/updateannotations
Method	POST
URL Params	string token
Post Params	List<DtoAnnotation> annotations
Response	(JSON) WebApiResponse with a Boolean if succeeded as the Result property

4.3.5.2. ASP.NET implementation example

```
[HttpPost]
[Route("updateannotations")]
public ActionResult UpdateAnnotations(string token, [FromBody] List<DtoAnnotation> annotations)
{
    //Implementation logic
}
```

Note: SmartDots sends all modified annotations to the web api.

4.3.5.3. Example Response

```
{"Result":true,"ErrorMessage":null,"WarningMessage":null}
```

4.3.6. DeleteAnnotations

4.3.6.1. Definition

Deletes list of annotations, returns true if succeeded

URL	/deleteannotations
Method	POST
URL Params	string token
Post Params	List<Guid> ids
Response	(JSON) WebApiResponse with a Boolean if succeeded as the Result property

4.3.6.2. ASP.NET implementation example

```
[HttpPost]
[Route("deleteannotations")]
public ActionResult DeleteAnnotations(string token, [FromBody] List<Guid> ids)
{
    //Implementation logic
}
```

4.3.6.3. Example Response

```
{"Result":true,"ErrorMessage":null,"WarningMessage":null}
```