|  |
| --- |
| Module CI869 |
| Constructed Wetlands Website |
| Software Design and Development |

Contents

[1 Introduction 4](#_Toc416947118)

[1.1 Purpose 4](#_Toc416947119)

[1.2 Participants 4](#_Toc416947120)

[1.3 Timeline 5](#_Toc416947121)

[2. Project Scope Statement 5](#_Toc416947122)

[2.1 Justification 6](#_Toc416947123)

[2.2 Project Scope Description 6](#_Toc416947124)

[2.3 Deliverables 6](#_Toc416947125)

[2.4 Project Exclusions 6](#_Toc416947126)

[2.5 Project Constraints 7](#_Toc416947127)

[3. Project Management 7](#_Toc416947128)

[3.1 Project Team contract 7](#_Toc416947129)

[3.2 Project Requirements 8](#_Toc416947130)

[3.3 Work Breakdown Structure 8](#_Toc416947131)

[3.4 Risk Analysis 10](#_Toc416947132)

[3.5 Glossary 12](#_Toc416947133)

[3.6 Problem Domain 13](#_Toc416947134)

[3.7 Research into Wetland Data 15](#_Toc416947135)

[4. System Architecture and Tools 17](#_Toc416947136)

[4.1 Selecting Technologies and Tools 17](#_Toc416947137)

[5. System Analysis and Design 18](#_Toc416947138)

[5.1 Requirements 18](#_Toc416947139)

[5.2 Use cases 19](#_Toc416947140)

[5.3 Database Structure 21](#_Toc416947141)

[5.4 Interaction Modelling 25](#_Toc416947142)

[Prototyping 25](#_Toc416947143)

[6. Implementation 28](#_Toc416947144)

[6.1 Code Structure 28](#_Toc416947145)

[6.2 Components 30](#_Toc416947146)

[Bootstrap 30](#_Toc416947147)

[JQuery Grid 32](#_Toc416947148)

[6.3 Data Persistence 33](#_Toc416947149)

[6.4 Security 34](#_Toc416947150)

[Cross-Site Request Forgery (CSRF) 34](#_Toc416947151)

[Password Protection 35](#_Toc416947152)

[SQL Injection 36](#_Toc416947153)

[Registering 36](#_Toc416947154)

[Validation 37](#_Toc416947155)

[Uploading files: 41](#_Toc416947156)

[Querying the Wetland’s Data 44](#_Toc416947157)

[6.5 Website Structure 45](#_Toc416947158)

[Sample page: 45](#_Toc416947159)

[Browse Literature: 49](#_Toc416947160)

[Forgotten Password: 49](#_Toc416947161)

[7. Testing 49](#_Toc416947162)

[7.1 System Testing 49](#_Toc416947163)

[7.2 Interface User Experience Testing 50](#_Toc416947164)

[References 51](#_Toc416947165)

[APPENDIX A Project Progress Reports 52](#_Toc416947166)

[APPENDIX B Source Change Control 54](#_Toc416947167)

[APPENDIX B Preparing for Interviews 55](#_Toc416947168)

# 1 **Introduction**

This document contains the requirements analysis and structural modelling for the Wetlands Database Management System being proposed for the storing site sample data and providing a website so that sample data can be made available online.

## Purpose

Create a constructed wetland performance database which would allow users to query the database to find and download data that matches particular search criteria.

Provide a user-friendly interface to allow interested parties to 'interrogate' the data (e.g. the end user may wish to examine the performance of wetlands, in a particular size category, in treating a specific water quality parameter, for example).

Increase awareness of the work done by The Geo-Environmental Engineering (GENE) research group, based at NUI Galway.

* Examine the material that needs to be modelled for the system. Gain an understanding of the problem domain and evolve a clear perspective on the essential requirements.
* Work with our client to develop a solution that is effective and achievable with in our limits of time and resources.
* Develop and practice group working skills that support good software development practices and encourage productivity from each team member.
* Maintain regular communications with team members, supervisor and client. Providing updates on progress.
* Produce a working demo model for the presentation.

## Participants

The development team throughout this project, 3 student from the HDip Software Design and development Course:

Ciaran Severn

Catherine Gaughan-Smith

Sean Lydon

Senior Technical advisor : Joe O’Connell

Project supervisor : Josephine Griffith.  
  
Module co-ordinator : Finlay Smith.  
  
Stakeholders:

|  |  |
| --- | --- |
| Owner | Dr. **Mark G. Healy** (room ENG-1038).  BE, MEngSc, PhD, Eur Ing, CEng, FIEI, Chartered Engineer,  Senior Lecturer in Civil Engineering |
| Administrator /  Wetlands Researcher | **Collette J Mulkeen** College of Engineering and Informatics  Conducting a PhD project - Biodiversity richness, water quality management and flood attenuation in natural and constructed wetlands |
| Expert User | Dr. Brian Donlon, EPA Research Manager ,  Environmental Protection Agency |

## Timeline

The key dates for the project are as follows:

Project Allocation – October 2014

First Presentation – November 2014

Phase 1 Report – November 2014

Project Demo – March 2015

Final Project Report – March 2015

# **Project Scope Statement**

The Geo-Environmental Engineering (GENE) research group, based at NUI Galway, are investigating the fate of phosphorus and nitrogen in soil-plant-water systems and the effects of agricultural management on soil and water quality. GENE has helped develop decision-making tools to target management alternatives and remedial measures that have reduced the risk of nutrient loss from farms. Currently, GENE’s main research interests are:

(1)   resource recovery and resource efficacy – use of materials for wastewater and water treatment; waste materials and chemicals as phosphorus and nitrogen adsorbents and soil amendments; waste as energy sources and slow-release fertilisers

(2)   forestry – protection of water courses from forestry activities, greenhouse gas emissions

(3)   soil physics – modelling of water movement through soil and management impacts for farmers



## Justification

The Geo-Environmental Engineering Constructed wetlands, sometime referred to as ‘reed beds’, are a common means of treatment of wastewater from single houses and small communities in Ireland.

They have a level of public acceptance, as they are considered to be a ‘green’ technology and blend into the landscape. Over the years, their performance has been monitored by County Council staff, who would frequently have poor methods of data storage (data were often not digitised). This meant that there was no centralised database for Ireland that allowed engineers, scientists, or the general public to determine how effective they were under Irish climatic conditions.

A recent EPA-funded study is seeking to address this knowledge gap.

As part of the study, the Geo-Environmental Engineering (GENE) research group are collecting performance data from wetlands in Ireland. However, they need a user‐friendly interface to allow interested parties to ‘interrogate’ the data (e.g. the end user may wish to examine the performance of wetlands, in a particular size category, in treating a specific water quality parameter, for example).

No such database exists in Europe, and this project constitutes a unique opportunity to advance the state-of-the-art in wetland analysis.

## Project Scope Description

• A constructed wetland performance database similar to an existing one in the USA (http://firehole.humboldt.edu/wetland/twdb.html), which would allow users to query the database to find and download data that matches particular search criteria.

• The data would be returned to the end user in tabular format.

• The database would need to be designed such that it may be continuously updated by the research group. Ideally, it may also be possible for County Council technicians to upload data onto the portal, using a login password.

• Ideally this database would be housed on our research group’s webpage

(http://www.nuigalway.ie/gene/)

## Deliverables

* Database
* Website
* Project Report

## Project Exclusions

The project will not produce any graphs or charts. This is because some of the data does not represent absolute values. Also there are some gaps in available sample data.

Finally, our client is concerned that a graph of the data may give a misleading view of the wetland performance.

The upload of sample data needs intervention from the administrator of the website. This is to ensure that the data is valid. The workflow agreed with the client is as follows: A registered user uploads a file to the site. The administrator reviews the uploaded file and uses this data source to enter sample data to the database.

## Project Constraints

The fixed deadline on in April for producing a working demo may mean that some of the less essential features of the design may need to be cut from the scope, if we come up against technical or scheduling issues. These scope changes will be agreed upon by the team and the client.

# **Project Management**

## Project Team contract

The purpose of the team contract is to establish a set of guidelines for how we will work together as a team. We discussed what we expect of one another in regard to participation, frequency of communication, attendance at meetings, and the quality of work.

Goals

To design and build a website that achieves the core functionality that the client requires which performs efficiently and securely, delivered to a fixed deadline, using the resources that we have access to; i.e. software licenced to NUIG, and working within the project guidelines of the CT869\_Software Design module.

Expectations

Keep other team members informed. Be proactive about anticipating problems and working to prevent them. Be realistic about estimates in team progress meeting.  
Focus on what is best for the project. Attend weekly progress meetings. If you have to miss a meeting, that get up-to-speed on progress as soon as possible.  
  
Policies & Procedures  
Protect confidential information. Use a common approach to development with tools and guidelines set up for the project. Different types of development tasks will be rotated among team members so that everyone gets experience with all parts of the project. Give everyone an equal opportunity to contribute. Be open to different approaches and listen to new ideas. Encourage a diversity of opinions. Get input from the entire team before a decision is made. Seek to find common ground for agreement.

Consequences

If a problem arises with a team member we will first discuss the issue with that person and give them an opportunity the work through the issue with the team’s support. If it still cannot be resolved the issue will be discussed with the supervisor.

## Project Requirements

#### Negotiate Scope

Although it would have been nice to show a visual representations of the distribution of wetlands across Ireland, the team agreed that it would probably not be achievable with in our schedule.

#### Identify Tasks

## Work Breakdown Structure

Our approach to managing the project development was to break down project deliverables into components. Each component is a task that one person can take ownership of and deliver in a weekly work cycle.

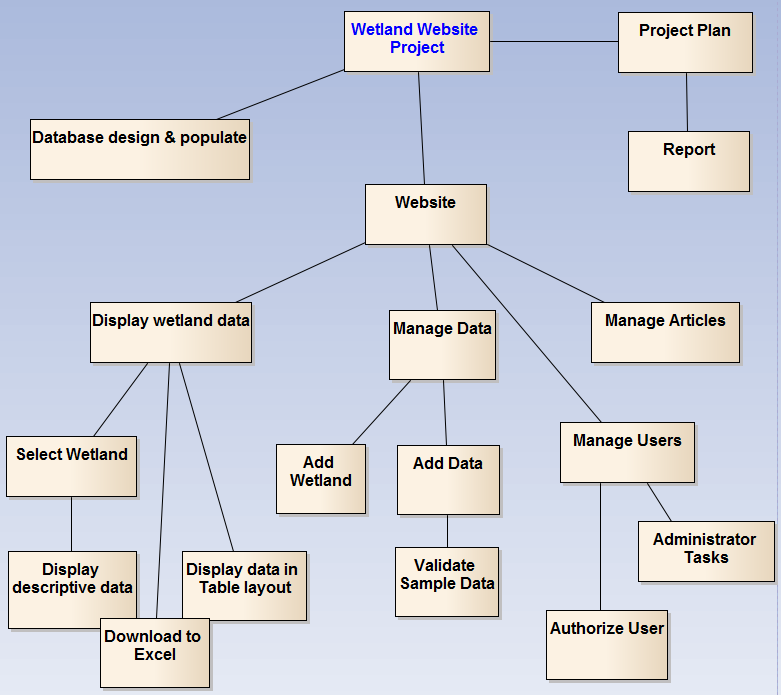
The Product Breakdown Structure diagram shows two major deliverables at the top: the wetland website and the project plan. The hierarchical structure below shows at each level a breaking down of these parts into smaller component parts. These parts are decomposed further until we get a group of parts that can be assigned to a team member as a task within a weekly work cycle based on the importance assigned to the task at each weekly meting.  
  
Through out the development we re-evaluated which components were more essential and then assigned the development of components to each team member. When the work was completed and tested, we went through a process if merging changes into the repository and then did some integration testing the confirm that the components still worked individually and also as part of the system.

Figure . Product Breakdown Structure

The **Product Breakdown Structure** (PBS) is a hierarchical **structure** of things that the project will deliver.

#### Estimate Task Durations

Project Name: Constructed Wetlands Status Date: 15/4/2015

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **PROJECT STATUS** | **Planned Start Date** | **Actual Start Date** | **Planned Complete Date** | **Actual Complete Date** | **Percent complete** |
| **Business Analysis** |  |  |  |  |  |
| Review problem domain | 20/10/14 | 20/10/14 | 10/1/15 | 20/1/15 | 80% |
| Develop requirements | 20/10/14 | 25/10/14 | 10/1/15 | 28/2/15 |  |
| Refine requirements |  | 10/1/14 |  | 28/2/15 | 100% |
| **Source Data Analysis** |  |  |  |  |  |
| Develop Data Model | 2/11/14 | 27/11/14 |  | 1/2/15 |  |
| Refine Data Model |  |  |  | 1/4/15 | 95% |
| **Prototype Review** |  |  |  |  |  |
| Develop Prototype Model | 10/1/15 | 10/1/14 | 28/2/15 | 28/2/15 | 85% |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
| **Technology Analysis** |  |  |  |  |  |
|  |  |  |  |  |  |
| Develop infrastructure | 10/1/15 |  | 21/1/15 | 21/1/15 |  |
| Transformation test |  |  |  |  |  |
| Develop website |  | 25/1/15 | 10/4/15 | 16/4/2015 | 70% |
|  |  |  |  |  |  |

#### Specify Inter-task Dependencies

The Database structure needs to be set up before starting to develop the website.

#### Direct the Team Effort

Each week we discussed what we had achieved in the previous week, what problems we had encountered that affected the completion of the task and then came to an agreement on what task we would focus on in the next week.

#### Assess Project Results and Experiences

We found that our lack of experience meant that we were over confident about what we could deliver on time. In retrospect, the learning curve had a bigger impact on our workflow. We didn’t factor into the schedule the time it would take to learn PHP and MYSQL and that cut into the time which should have been used to complete tasks.

Coding in PHP, JavaScript and HTML where complicated by the characteristics of the language and the development environment. If a tag is out of place or a script is in the wrong order then the page can fail to load properly and there is very limited feedback to debug the problem. It takes some experience as a developer of website applications to know how to recover from such bugs, and avoid them.

In retrospect we found ourselves cutting corners on testing and good interaction design because we were struggling with the technical issues.

On the other hand we completed about 70% of the website which works with the sample data. The database supports all the functionality that the fully complete website would use.

## Risk Analysis

#### Issues with workflow

As we progressed through the academic year, other modules contributed more demands on time then we had anticipated. Other assignments took up more time then expected and in the short term where more urgent. This meant that we needed to re-evaluate the scope of what we could achieve on a regular basis.

|  |
| --- |
|  |

The Cost / Quality trade-off Triangle shows the trade-offs inherent in any project.

* **Product:** Scope and performance outlined in Requirements
* **Schedule: Fixed –** Week 12 semester 2
* **Cost / Resources : Fixed team size**  **-** 3 developers. **Fixed Software budget** – zero. We are using the development platform provided by NUIG and any additional components must be licenced for academic use.

Our strategy from the beginning for coping with time issues:

* 1. Ask for help early on. Let other team members know if you are struggling to get something done. Ask for help from DISC drop in support centre, or tutors from our course modules.
  2. At our weekly meeting, decide which approach to take when a piece of work has not been achieved on schedule:
     1. change design / function for what your doing.
     2. change how you are doing it, for example consider a different approach.
     3. put in extra time to learn how to do it, maybe do an online tutorial.
     4. assign it to someone else.
     5. renegotiate the requirement.

#### Issues with expertise

There was steep learning curve for all of us as no member of the team had written a client server application.  
  
Our strategy:   
Once we understood what we needed our application to do, we identified the most challenging technical issues and focused our initial effort on a proof of concept for these challenges. A proof of concept is a very basic sample that just tests that we can achieve a working example within the scope of our technical environment, skill set and resources.  
  
For example:   
1. Creating a test page on the web server that communicated with the database.

2. Get a simple logon page functioning before any front end design is applied.

3. Adding a simple grid to a web page to view data from a single table on the database.

Once a proof of concept was completed, we had more confidence in estimating and time and effort required to complete the task.

#### Issues with communication

Given that there were so decisions around planning, requirements analysis, database design, website design, technical approaches, tools to use, and source code control we set up a folder on Microsoft One Drive to keep all our notes and project resources in the cloud.   
We setup ONE-NOTE as a filing system online with sections for each part : Project Planning, Database, Website, Technical (tools and tutorials), Design (requirement and modelling) , Project Progress.  
In this way, we all had easy access the project resources as well as continuous documentation of design decisions.

Communication with the client was not an issue. We were able to set up frequent meetings and they were very generous with the time and effort they gave to supporting our project. They also had a clear idea of what they wanted in terms of functionality of the website. We used wireframe mock-ups the explore good design and user interaction.

#### Issues with technology skill

As a team we shared the development work evenly. That meant that we developed the skills required to work interchangeable on tasks. There was not one expert holding onto knowledge to the exclusion of anyone else.  
  
Knowledge transfer was a key strategy on our project. We shared links to online tutorials so we each progressed to the same level of skill by completing the same training.  
We set up a knowledge transfer on the integration of third party components and maintained references to useful tools and techniques in our ONE-NOTE filing system in the cloud.

#### Issues with protecting source code

Given that our product is entirely code based a very significant risk to guard against is loss or corruption of our code base. We needed to be able to work concurrently on the code base and then be able to integrate our code changes in a way that didn’t wipe out someone else’s work or introduce bugs to a working version that had already been tested.  
At the same time we needed to protect against loosing some or all of the code if it is just held in one location (on one server).

Our solution to this is source change control using GIT repositories and an online repository on the GIT HUB website. We did a full integration of code changes made by each developer and committed these changes into the repository in the cloud, at least once a week. The benefit was that we always had access to the latest version of the code, no matter if we were working at UNI or separately at home.   
Secondly, we had clones of the repository on different servers which gave us protection against loosing one server. An added benefit of version source control is that we can look back at previous versions of the code and compare the changes between versions.

## Glossary

|  |  |
| --- | --- |
| **EPA** | **Environmental Protection Agency** |
| Flow rate | Total effluent volume discharged over the 24-hour period in which the composite sample is collected shall be recorded. |
| BOD | BOD 5 day **Biochemical Oxygen Demand** (without nitrification suppression). The BOD test is a bioassay in which the rate (and extent) of the aerobic degradation of organic matter is assessed in terms of the amount of oxygen consumed in its degradation.  Unit is mgO2/L, or in short mg/L |
| cBOD | 5 day Carbonaceous Biochemical Oxygen Demand (with nitrification  suppression). |
| COD | **Chemical Oxygen Demand** Expressed as amount of oxygen required for chemical oxidation of organic matter by a strong oxidant (permanganate or dichromate) in acid solution.  Unit is mgO2/L, or in short mg/L |
| DO | Dissolved Oxygen |
| NH 4 -N | **Ammonia nitrogen** |
| RBC | **Rotating Biological** **Contactor,** for example in Hollymount, Fenagh and Newtowngore. |
| TSS | **Total suspended Solids** |

## Problem Domain

Constructed wetlands are primarily used to treat domestic and municipal wastewaters but their use for other types of wastewater such as agricultural and industrial wastewaters, various runoff waters and landfill leachate.

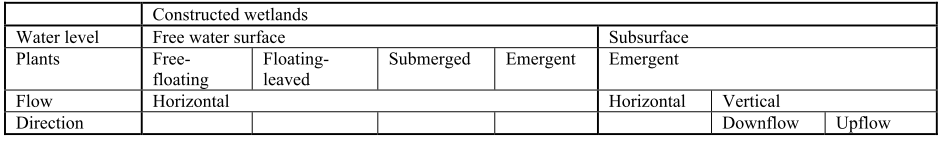


Figure . Types of Constructed Wetlands

|  |
| --- |
|  |

Figure . Pre-treatment

Raw sewage moves from the grit chamber to primary treatment, where sludge is removed and the clarified water then proceeds to secondary treatment (here shown as activated sludge treatment).[[1]](#footnote-1)

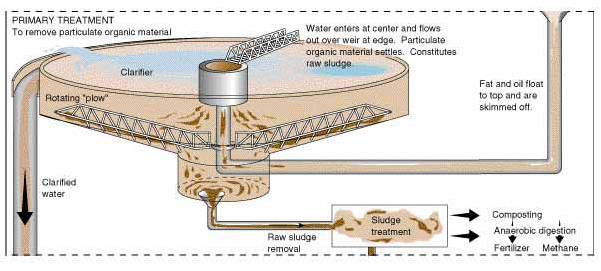


Figure . Primary Treatment

In primary treatment sludge is removed and the clarified water then proceeds to secondary treatment. Raw sewage moves from the grit chamber to primary treatment, where sludge is removed and the clarified water then proceeds to secondary treatment. [[2]](#footnote-2)

|  |
| --- |
| Raw sewage moves from the grit chamber to primary treatment, where sludge is removed and the clarified water then proceeds to secondary treatment (here shown as activated sludge treatment). |
| Figure . Secondary Treatment  The secondary treatment, activated sludge process may be modified to remove nitrogen and phosphate while at the same time breaking down organic matter.[[3]](#footnote-3) |

##### Wetland Performance

The ecological factors that define wetlands (hydrology, soils, and biota) are the basis for indicators (or assessment questions) with broad applicability under a wide range of circumstances and are expected to be components of any method. We define these as the core elements of a method. Common indicators reflecting the core elements are shown in Table 3 and include those on hydrology, soils, vegetation, and landscape setting.

Machine generated alternative text:
Core Element 
Soils/substrate: 
Indicators developed for, or 
based on: 
H drol ic alterations stressors 
iod 
T of outlet restriction 
Water li 
Surface water connectivi 
Flood stora 
tential 
Groundwater recharge and/or 
disch 
Water source s 
Dc of water level fluctuation 
Maximum water th 
soil 
Substrate disturbance 
Presence of mottles 
Dc th of A horizon 
Munscll color matrix/mottles 
Micro 
Sediment com ition 
settin : 
Number of v etation classes 
Degree of interspersion 
communl 
s or 
Extent of invasive s ecies 
Vc tation alterations 
Habitat value to wildlife 
Endangered/threatencd species, 
their habitat or communities 
Coarse wood debris 
Dominant Vc tation 
Plant s ecies divcrsl 
Area Of en water 
Surroundin land use cover 
Connectivity to Other wetlands or 
corridors 
Extent Of and/or vegetation type 
in buffer zone 
Extent of human land use in 
buffer 
Wetland size 
Ratio of wetland to watershed 
sin or watershed size 
Land use in watershed 
Wetland 
hoi 
Position of wetland in watershed 

Figure . Wetland Condition Assessment Common Indicators

## Research into Wetland Data

|  |  |  |  |
| --- | --- | --- | --- |
| Wetlands | Identification: Wetland name (for example - Moycullen, Galway) Location: GPS coordinates,  Type: Bog, fens, cutover and cutaway bog areas.   |  |  | | --- | --- | | Wetland location   * latitude and longitude * city * state * EPA region * EPA facility ID * USGS watershed ID | Wetland owner, operator, designer, and regulator contact information   * name * addresses * phone numbers * email * web page entries |  * Design characteristics descriptors   + include population of the service community   + scale of the operation (bench, pilot, or full-scale)   + source of wastewater   + period and type of hydraulic loading   + type of pre-treatment   + type of pre disinfection   + objectives and beneficial uses of the wetland system   + dates of start-up and full operation   + capital costs   + O/M costs   + hydraulic type and loading rate   + total wetland footprint area |
| Site Source Type | Municipal,  Agricultural,  Industrial |
| Types of Pre-treatment | Primary,  Secondary |
| sample points | The sample points data table contains a description of all sample point locations where data is collected for the wetland systems included in the database. Each entry in this table contains   * identification number of the associated wetland * name and description of the point * cell in the wetland (if any) where the sample point is located * indication whether this sample point is a treatment system influent or effluent point |
| constituent measured | The samples data table contains the actual measured wetland descriptive or performance data. Each entry in this table contains   * identification number of the associated wetland * date the data was collected * identification number of the constituent measured * observed value of the constituent or sample variable * text comment associated with the sample |
| Meteorological characteristics | * monthly average precipitation * monthly minimum temperature * monthly maximum temperature * monthly evapotranspiration * annual average temperature, precipitation, and evapotranspiration * average first day of ice cover * average number of ice cover days |

# **System Architecture and Tools**

##### LAMP Stack – Linux, Apache, MySQL and PHP

We chose this as a best fit for developing our website project.   
The advantages are that it is open-source, and free to use. There is a large community of developers using this architecture meaning that it is easy to get answers to question when problems arise.

We also considered that this website would be hosted on the university servers, so the LAMP Stack matched what was already the standard architecture at NUI Galway.

## Selecting Technologies and Tools

* Twitter bootstrap
* My SQL
* PHP
* Java script

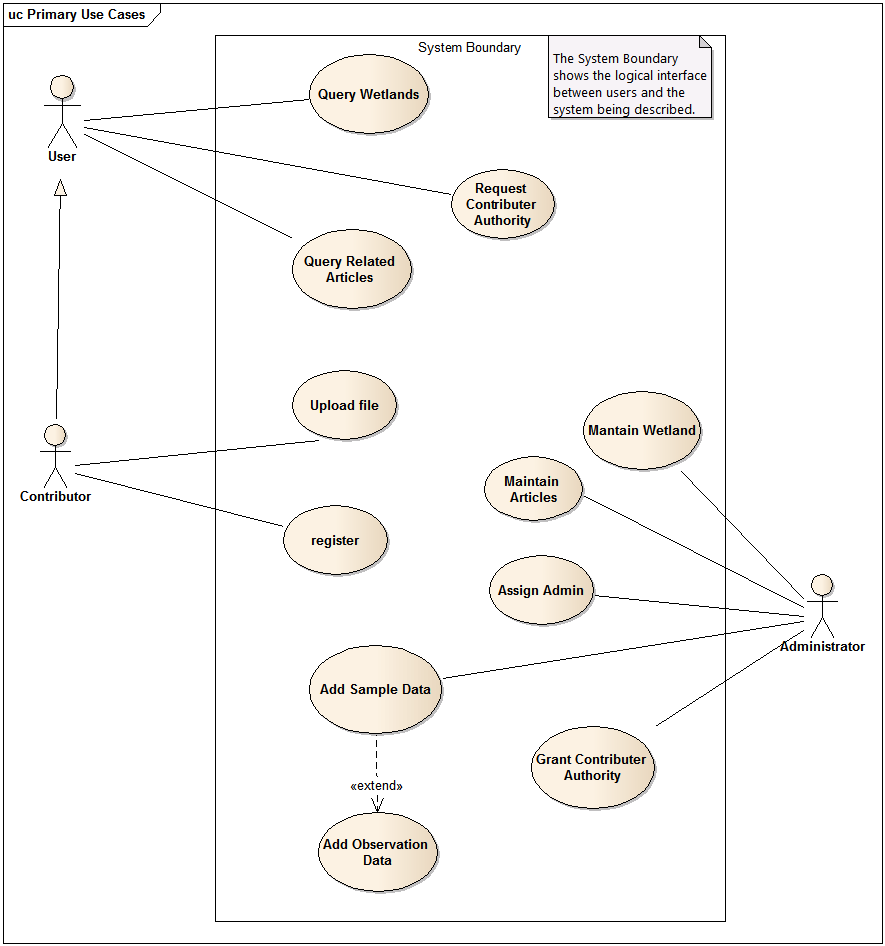
# **System Analysis and Design**

This document

## Requirements

1. **Scope of Functions**
   * 1.1. The system will allow user to query a list of wetlands in Ireland by county, site source and pre-treatment type.
   * 1.2 The system will allow the user to view the sample data is available for a give wetland
   * 1.3 The system will allow the user to download a set of sample data in a given date range.
   * 1.4 The system will present the user with a list of publications about wetlands, showing NUIG publications at the top of the list.
   * 1.5 The system will show information and images about the wetland when they are available.
   * 1.6 The system will allow registered users to upload files to the server for the purpose of contributing data to the site.
   * 1.7 The system will show a list of uploaded files to the logged in administrator.
   * 1.8 The Administrator can download the contributed files and use them to enter the sample data through an add/update/ delete form for Sample Data.
   * 1.9 The Administrator can add/update/ delete wetland data.
   * 2.0 The Administrator can maintain the publication list and change the order it is displayed on the website.
2. **Non-Functional requirements**
   * Design the interface to target people with scientific background.
   * Database can be updated without having specialized skills.
   * Promote NUIG publications and the GENE Research website.
3. **Operational**
   * 2.1 The system will run on any Web browser and on the intranet.
4. **Performance**
   * 3.1 Download speeds.
5. **Security**
   * 4.1 Only registered users can contribute to the website and upload files.
   * 4.2 File uploaded to the system are kept securely and cannot be access through a URL request.

## Use cases



USE CASE - View Sample Data

|  |
| --- |
|  |
| 1. Select County, Site Source and Pre-treatment on a wetlands Query   * The system lists the wetlands found for the search criteria.   2. Select a wetland from the returned list.   * The system lists the sample data for the selected wetland.   3. The user selects a date range to filter the sample data.   * The system lists the sample data in the date range.   4. The user selects the export to excel option.   * The system send the excel file to the browser. |

USE CASE - Submit For Review

|  |
| --- |
|  |
| 1. Log in to the website   * The system confirms you registered and logs you into a new session.   2. Navigate to webpage to upload data.   * The system presents a option to browse the local directory for the target file.   3. The user selects a target file to upload, and a description of the  target file, then submits the request   * The system copies the target file on the server and sends a response to the user confirming the upload was successful.   Exception:  1 (a). The system does not find the user in the database. The response to the user ask the user to try again or register if you are not a user. |
|  |

USE CASE – Request Contributor authority

|  |
| --- |
|  |
| 1. Go to contact page of the website.  2. Submit a request to contribute data as a registered user. |
|  |

USE CASE – Grant Contributor authority

|  |
| --- |
|  |
| 1. Sign in to the website as Administrator.  2. Navigate to the Administrators Tasks Page and select Maintain Users  3. Select a user from the list of registered users  4. Change the Access Level for the user to Authorized   * The system updates the user with the new access level and emails the user to confirm that the request was processed.   Exception:  3 (a). The system does not find the user in the database.  The Use case ends. |
|  |

USE CASE – Enter Sample Data

|  |
| --- |
|  |
| 1. Sign in to the website as Administrator.  2. Navigate to the Administrators Tasks Page and select Maintain Sample Data  3. Select a wetland and enter a date for the sample  4. Enter the inlet and outlet constituent values from the sample data.   * The system validates the values entered are the numeric * The system save the data to the database. |
|  |

## Database Structure

After the initial analysis of the required database scheme required, the list of tables needed to support the website functionality was extended to include tables for managing user credentials and managing the upload of files.

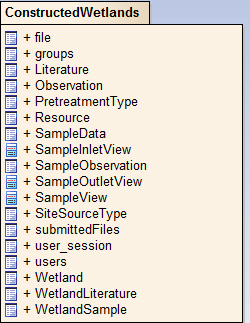


Figure . Final list of tables and views implemented

The schema for the data was formed through the following processes:

* We examined a example spreadsheet and discussed with the client how the information was collected, and the characteristics of the data : how often, how is it measured
* Normalized the collection of data to remove repeating groups and prevent redundancy. For example a wetland would have multiple sample taken on different dates. On the same site inspection date, there may be one or two samples taken at different sample point. This set of data is spread across three tables, Wetlands, WetlandsSample and SampleData.
* As we modelled the interface and use-case descriptions, we reviewed the information that the system needs to capture and retain.

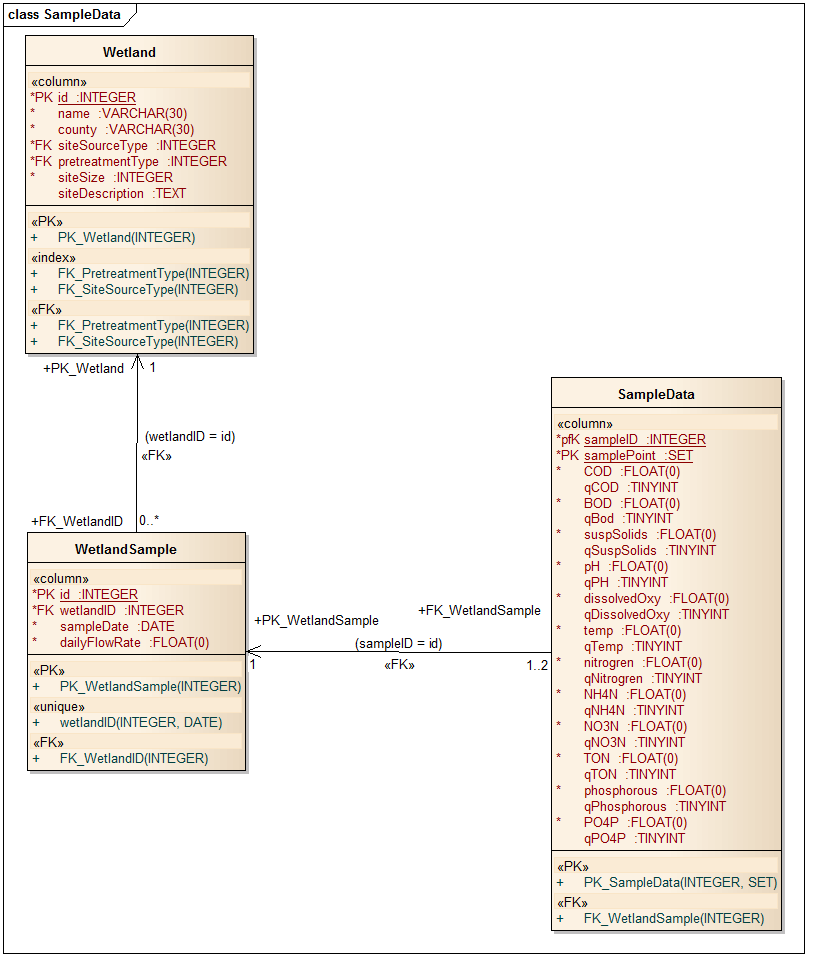
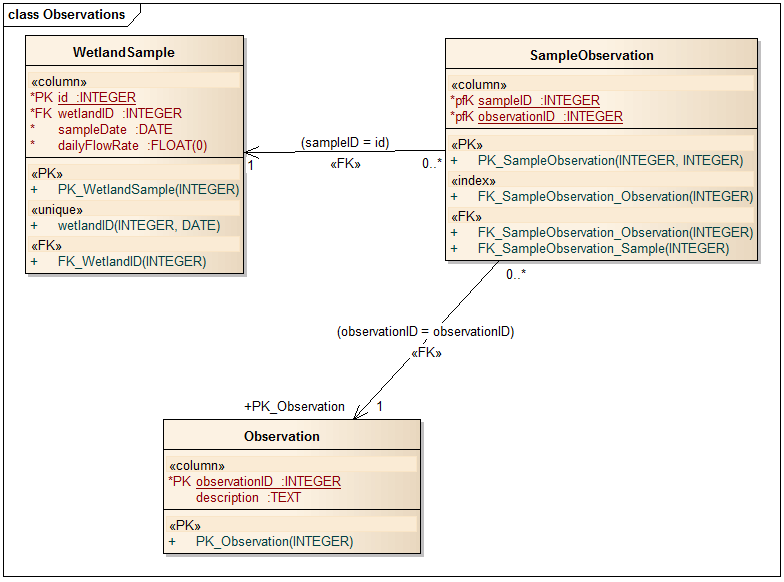
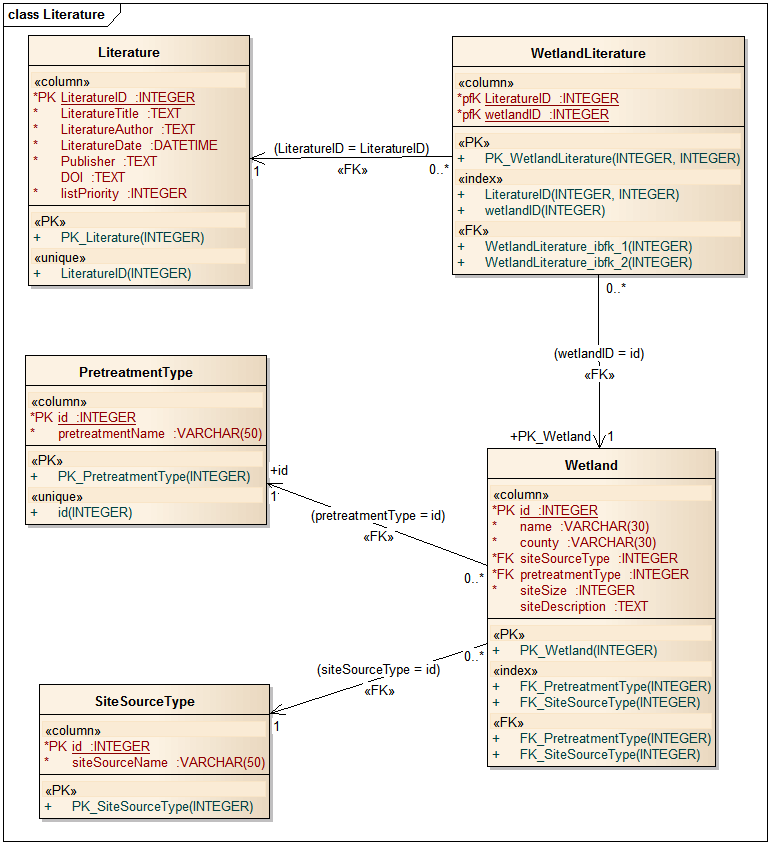


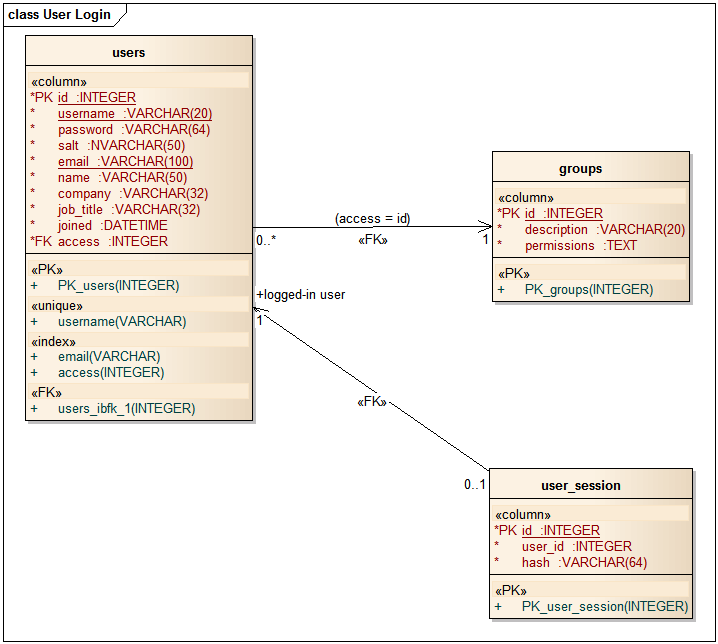
Figure . Table schema for Wetland Samples

Wetland sample data  
When a site engineer makes a visit to inspect the wetlands site he/she may measure the flowrate on that day and will take a sample for either the inlet or outlet points, or even a samples from both points. The same type of sample information is collected at both points.  
In the database schema the WetlandSample table represents the data collected on a single site visit, uniquely identified as WetlandSample.id column. The SampleData table represents one or two samples taken on a visit uniquely identified with the key { WetlandSample.id, SampleData.samplePoint }.

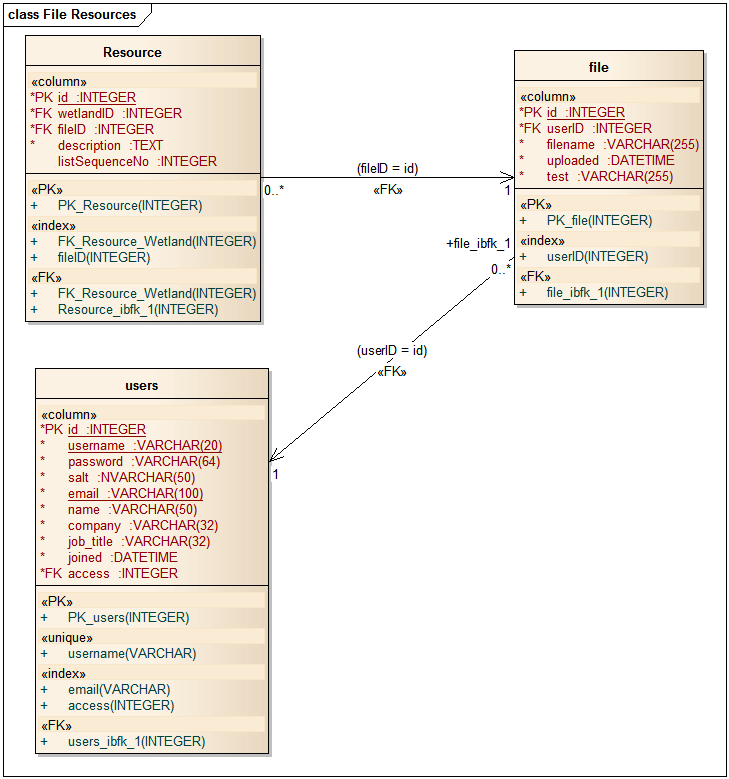


Wetland Site Visit Observations  
If the inspection of the wetlands yields additional information that has a bearing on the quality of the sample data, then a description of the observation from the visit may be recorded. This provides background information which may relate to sample data variations, performance issues or contaminations.  


Literature  
The clients requested a list of articles / publications. Since that is a many to many relationship between Literature and Wetlands, as in there may be a research paper written about a group of wetlands we would want to list this paper on a lookup of literature for any one of these wetlands. Therefore we have a linking table called WetlandLiterature which replaces the many to many relationship with a one to many relationship from table WetlandLiterature to both Literature and Wetland.



Registered User  
We use the Group table to hold the permissions for different type of user.  
This distinguishes between users, registered user and administrators. These permissions control what views and functions are available to the logged in user.



Files can only be uploaded to the website by logged in users. The actual file is stored at a location on the server and this location is what is stored in the file Table. A resource record links a file on the system to or more Wetlands and has a description for the file, for instance if the file in an image then the resource description would contain a description about the image.

## Interaction Modelling

One of the most important elements of human-centered design is making sure that all the different design choices you make —

what functionality is available and what interface elements, information architecture, and visual design present that functionality harmoniously combine to fluidly support the activities that matter to people that are using your system.

### Prototyping

Prototyping is a valuable strategy for effective design. By trying things out and learning — from that exploration — we are able to improve our design and were able to explore the problem domain with our client to gain insight into features required and core concerns that need to be resolved.

Using **Balsamiq** wireframes provided a very effective way to mock up proposed interfaces in a low fidelity format. This provided a common ground for discussing the interaction required to achieve user goals. It provided visual language to help people understand really concretely what everybody is talking about and stimulate ideas for alternative approaches to solving the user interaction design.

The wireframe allowed us to focus on the goals from the point of view of the user of our website.

We evolved the design through discussion with our client. Then we used the wireframe design as a template for developing the pages of our website.

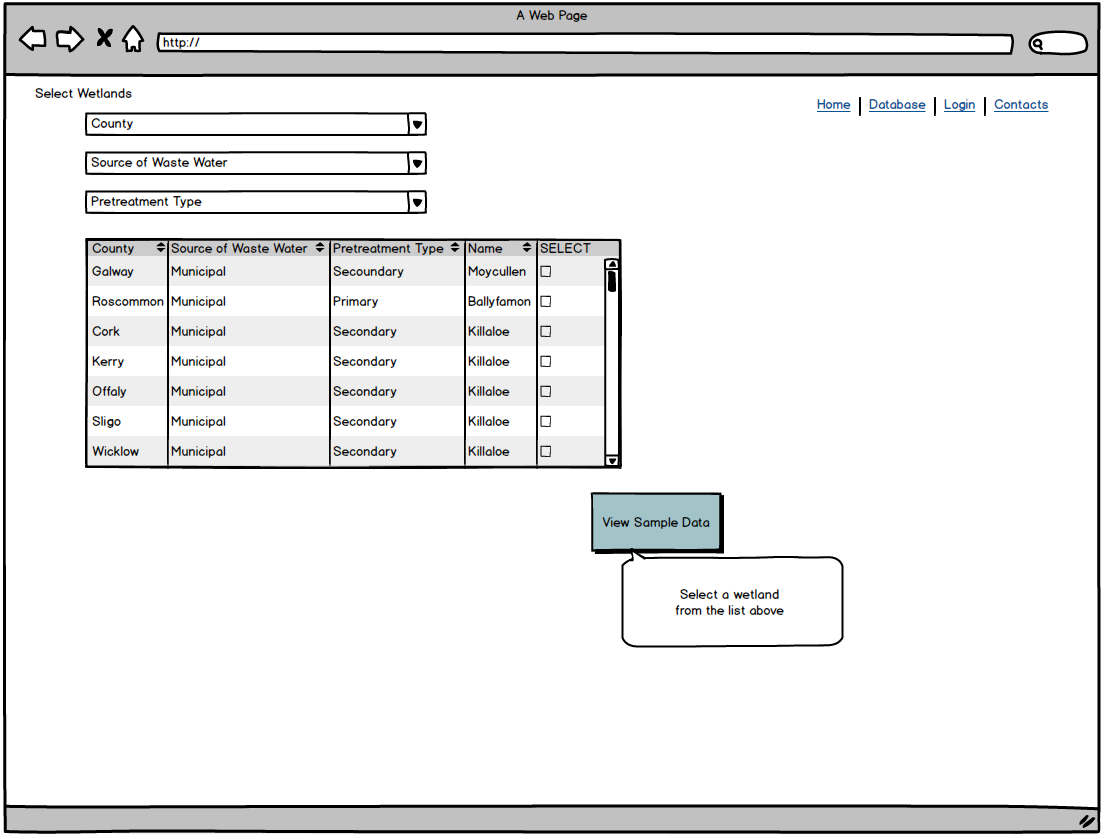


Figure . Design for selecting a Wetland from a list of wetlands

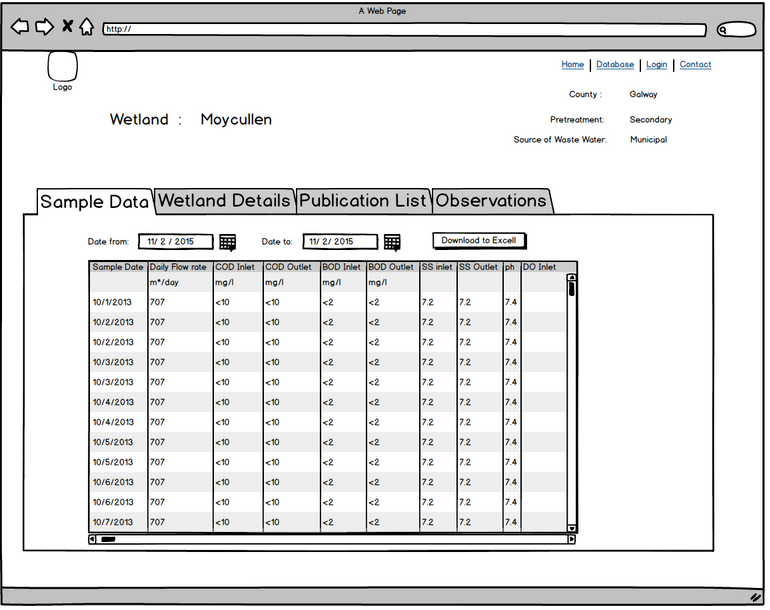


Figure . Design of Sample Data View

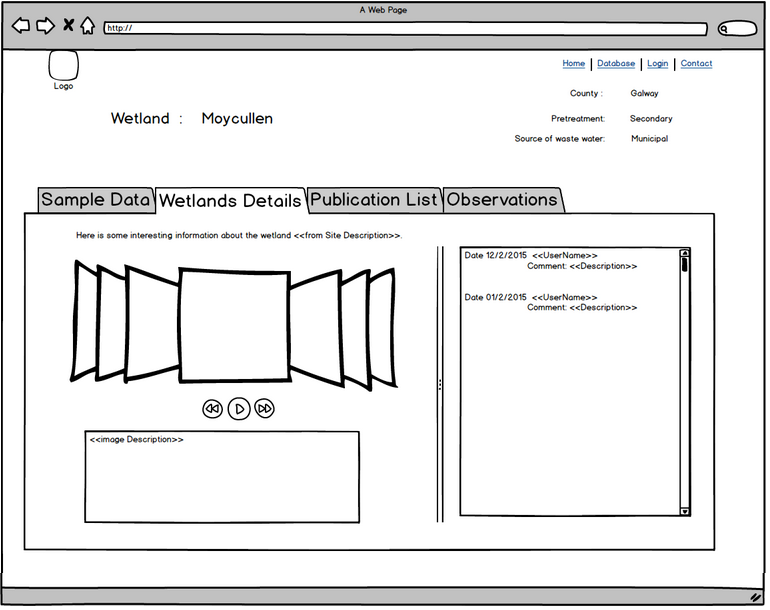


Figure . Design for Wetland Images and descriptions

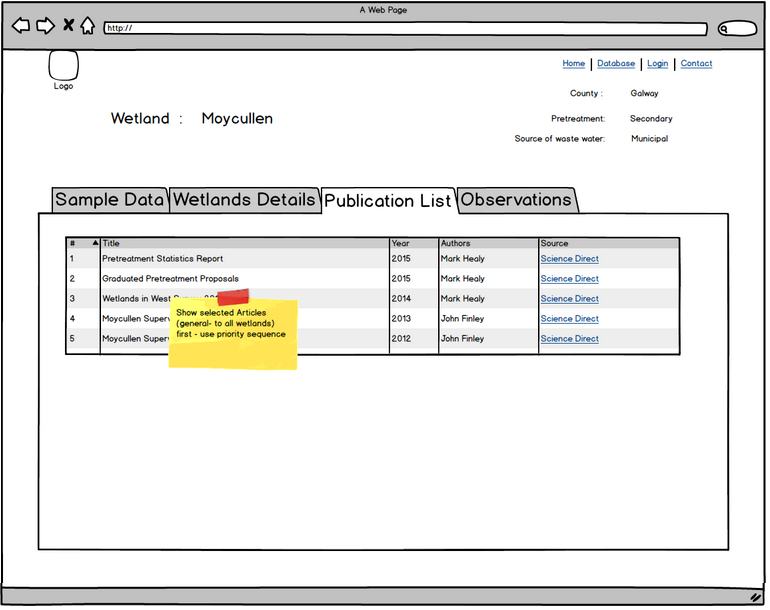


Figure . Design for Literature linked to the wetlands

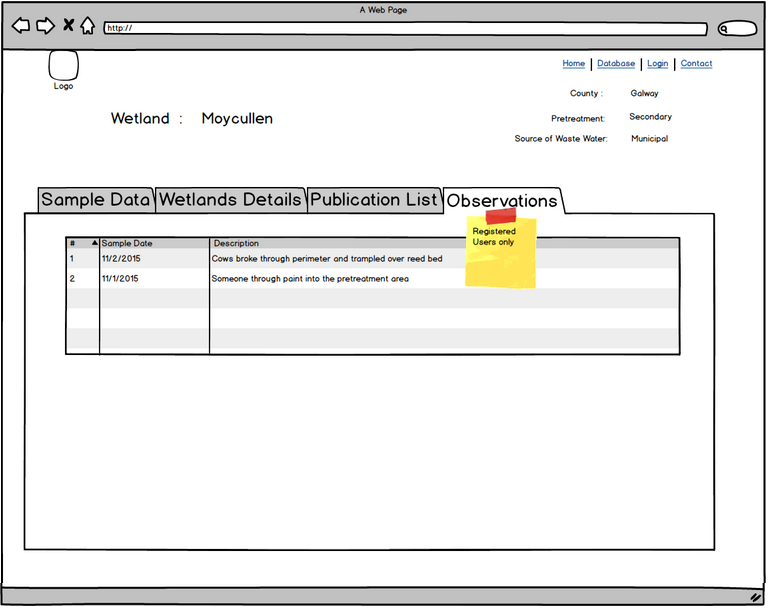


Figure . Design for Sample Inspection Observation, restricted to the Registered Users only

# **Implementation**

## Code Structure

Object oriented PHP Classes  
We decided to use an object oriented approach to developing the website. The main advantages of using the OOP method are the reusability of code and a clearer structure of code. As part of this method several classes were built that have functionality that can be used in different parts of the website.

|  |
| --- |
|  |

Figure . Classes in website directory

The classes were a mix of main classes and helper classes. For example the “DB” class is a main class with a lot of functionality and the “Session” class is a helper class that made session configuration more efficient.

##### classes folder : helper functions

**Config.php:** provides a static getter function for global variables which can traverse a path through nested array elements :  **Cookie.php** uses static functions for getters / setters on cookies.  
**DB.php** is a wrapper *class* for PDO access to database. It provides helper methods to simplify queries and I/O on the data. Uses the *singleton pattern* to return a instance of database that only connects once on a page load ‘round trip’.  
**Hash.php** uses a secure hash algorithm to protect passwords and cookies.

**Input.php** for accessing super global GET / POST values, useful for doing form validation.  
**Redirect.php:** provides a static function which wraps the php header redirect command.  
**Session.php:** uses static functions for getters / setters on sessions.

**Token.php**: provides *cross site request forgery protection*, matches the current uses session token when a form is submitted.

**User.php:** controller class for **User** entity, logging in and checking permissions.

**Validate.php**: provides user input validation, boundary class.

##### core Folder : autoload classes

**init.php**: functionality to include on each page: start session so people can log in, access global configuration settings for the database, cookies and session token, and autoload classes so that classes are loaded efficiently (when they are required) offering *dependency* *injection* with php function spl\_autoload\_register. This will replace the list of all the required\_once functions on each page for a single point of reference.

##### functions Folder : sanitize functions

**Sanitize.php**: function that uses html encoding to sanitize input / output.

##### includes Folder: Error Handling 404

errors folder : **404**.**php** standard message

##### Static Helper Files

Validation Helpers : Input is automatically sanitized  
config.php : Single location for configuration values

The “init” class is required on every web page. This class includes the “Sql\_autoload\_register()” function. The “Sql\_autoload\_register()” is a quick and efficient way of loading in classes when they are required rather than creating and maintain a list of requires. The function is called, the parameter $class represents DB and the DB class is required in. This eliminates unnecessary requiring.

|  |
| --- |
|  |

Figure . “Init” class being required

|  |
| --- |
|  |

Figure . Auto loading classes

##### Login / Permissions

Functionality: User Validation, User Session (Remember me option), Manage profiles, Update passwords, Ability to Register

User\_Session : Holds session hashes of people who have asked to be remembered when they logged in. Stores a hash that corresponds to a user ID and if this matches a cookie hash then that user will be logged in.

## Components

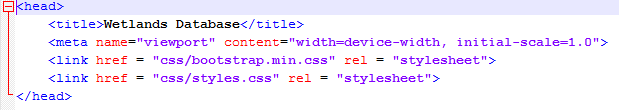
### Bootstrap

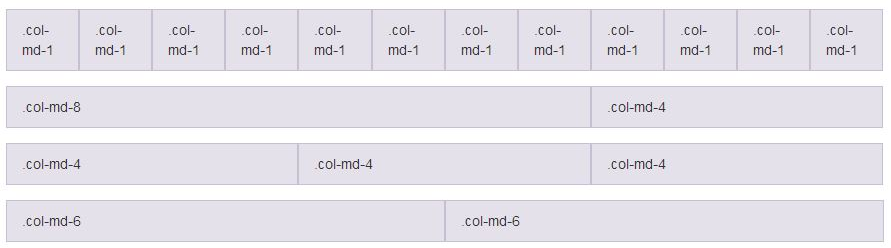
We decided to use Bootstrap on the website as it an efficient and easy to use HTML, CSS, and JavaScript framework for developing web sites.

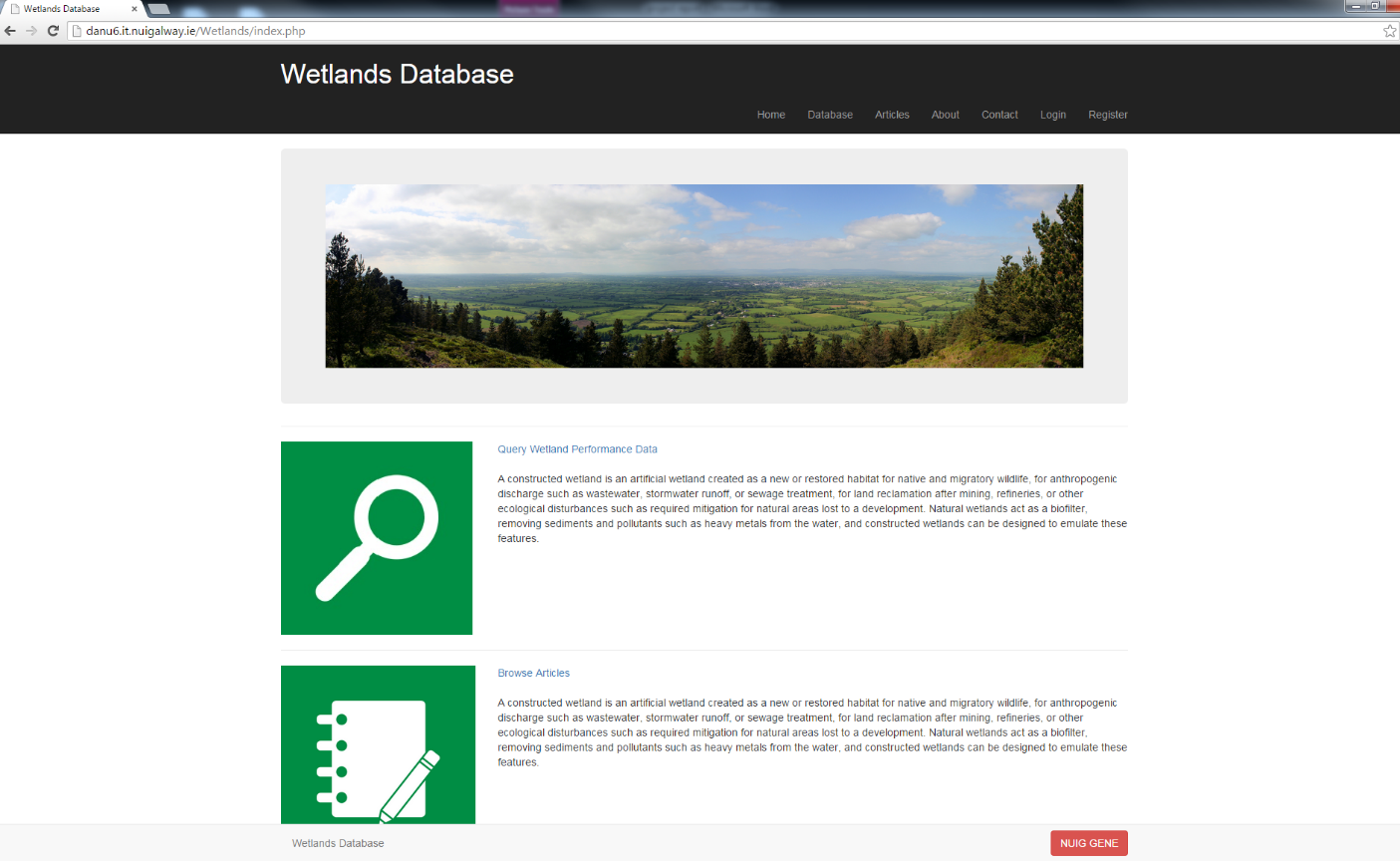
Bootstrap provides a set of style sheets that provide basic style definitions for all key HTML components. These provide a uniform, modern appearance for formatting text, tables and form elements. It also has cross-browser compatibility and CSS-Less functionality. This ensures uniformity across platforms so output remains the same whether you’re using Firefox, Chrome or Internet Explorer.

Bootstrap uses a responsive grid system that re-arrange automatically depending on the screen size. Whether you are using mobile phones, tablets and PCs each variation adjusts the width of the columns.

The “viewport meta tag” ensures proper rendering and touch zooming on mobile devices. The width property controls the width of the device. Setting it to device-width ensure that that the web page is rendered across various devices properly. The “initial-scale=1.0” property ensures that when loaded the web page will be rendered at 1:1 scale, and no zooming will be applied out of the box.

Figure . Bootstrap CSS in the header

Figure . Bootstrap grid system

Figure . Bootstrap web page

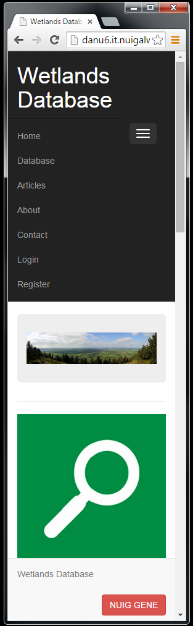
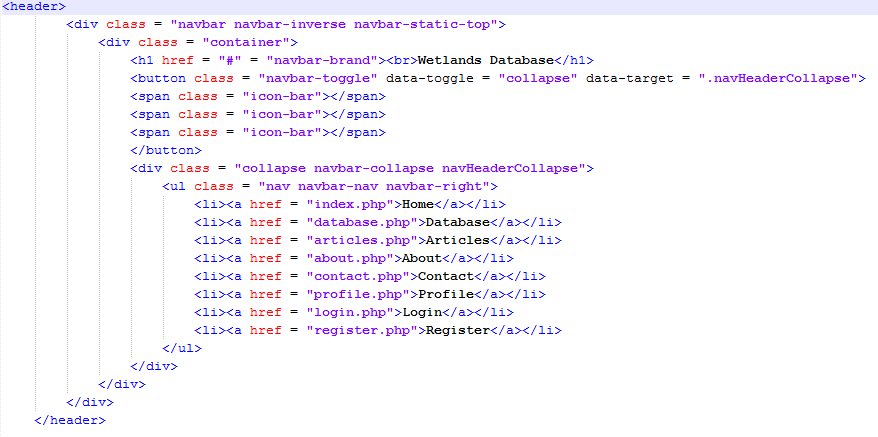


Figure . Bootstrap webpage resizing

Figure . Bootstrap HTML example

JQuery Grid

The “jqGrid” is a grid plugin for the JQuery JavaScript library. It is an efficient way to represent database information and provides solutions for manipulating tabular data on the web. The data is loaded into the table via a “SelectCommand” and the grid provides the option of searching, adding, editing and exporting the data. The grid was used in several areas on the website.

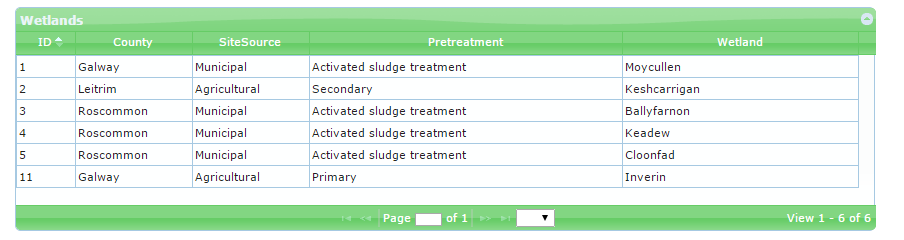


Figure jqGrid



Figure jqGrid code

## Data Persistence

Here is an example of the OO functionality of the classes.

The “DB” class is a database wrapper class. Its purpose is to have an abstracted way to work with the database. It contains various functions that can be used in different parts of the website to perform different functions.

The “\_construct” function connects to the database. A global configuration is set up in the “init” class. This contains the information needed to connect to the database. This allows for more efficient management. For example, the database information only needs to be updated once and not in ever place that the database is used.

|  |
| --- |
|  |

Figure . Database private constructor, singleton pattern

|  |
| --- |
|  |

Figure . getInstance() function of singleton pattern

The “getInstance” function gets an instance of our database if it’s already been instantiated. This means we don’t have to keep connecting to the database again and again on each page. We can use the database on the fly.

|  |
| --- |
|  |

Figure \_construct function in the DB class

## Security

### Cross-Site Request Forgery (CSRF)

Cross-Site Request Forgery (CSRF) is a type attack where an attacker will use the identity and privileges of the victim and impersonate them in order to perform any actions desired by the attacker, such as change form submission details.

A token generation system was implemented to protect against this kind of attack. We generate a token that only allows data from the form to interpret. This makes sure that the requests are coming from the actual users of the site. A token is generated for the forms and must be tied to the user's sessions. It is used to send requests to the server, in which the token validates them.

|  |
| --- |
|  |

Figure Token in the session array

|  |
| --- |
|  |

Figure Token generation in the form

|  |
| --- |
|  |

Figure Token validation

### Password Protection

To protect passwords salted password hashing was employed.

Hashing a password will take a clear text string and perform an algorithm on it to get a completely different value. This prevents you from storing the clear text passwords in the database. The Hash function “SHA256” was used in this website.

Instead of storing a user's password, you can store the hash of the password. When a user logs in again, instead of checking the password they type in against the one you have stored, you calculate the hash of the password they type in and compare that to the stored hash.

However, hashing alone will not provide enough protection. There are many ways to recover passwords from plain hashes very quickly such as dictionary and brute force attacks. An effective way for ensuring that passwords are securely protected is by adding a salt.

A salt improves the security of a password hash because it adds a randomly generated secure string of data onto the end of a data that can then be added to. The hashed password and the salt are stored in the database. In figure 111, the users have the same password but there hashed password is different because of the salt. This ensures secure password protection.

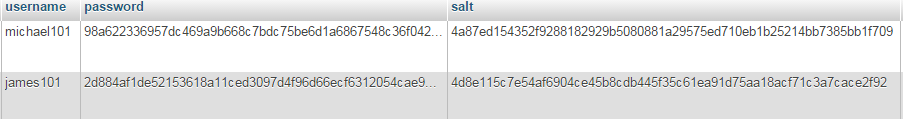


Figure Example of a salted password hash on the same password for two different users

The website required the ability to register a user, log in, query a database, have a permissions system, upload files, download files\*

### SQL Injection

A SQL injection attack is where harmful/malicious SQL code is into injected into the database, and the database is forced to run his SQL. An example of SQL injection create an SQL statement to select a user with a given user id

**Htaccess**

A “Htaccess” file was created and inserted into the “uploaded” folder. With this header the browser opens a download dialog instead of showing the uploaded folder path in the URL while downloading files.

|  |
| --- |
|  |

Figure Htaccess file

### Registering

To register an account a user must fill out the “Register Now” form and click the register button. There details are stored in the database once registered and can then log in.

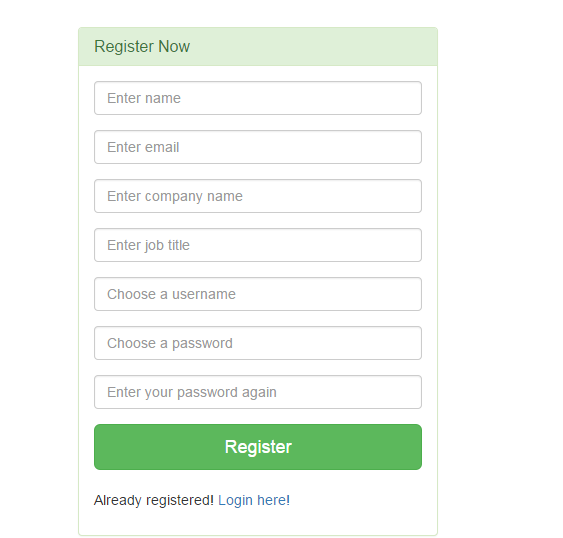


Figure Register form

### Validation

The input from the user must pass validation conditions. For example, the “username” chosen must be unique from any other username in the database and the “Enter your password again” input must match the input on “Choose a password”.

The “check” function in the “validate” class outlines the validation cases. The conditions of these cases are defined when the “validate” class is instantiated. For example, the case “min” calls the function “strlen” on the posted data. The “strlen” checks the length of the string on the data. If the “$rule\_value” is set to “2” the input from the user must contain at least two characters or else an error message is outputted.

|  |
| --- |
|  |

Figure Validation rule example

|  |
| --- |
|  |

Figure Defining validation requirements

#### Logging in

To login a registered user must enter their chosen username and password into the login area. If an incorrect username and/or password is/are entered an error message is outputted.

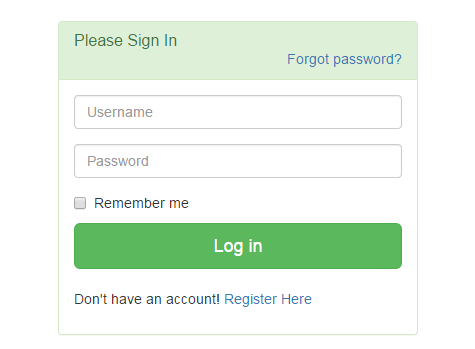


Figure Login form

When a user logs in there “userid” is assign to the “Session” which essentially logs the user in.

|  |
| --- |
|  |

Figure Logging user in

When a user is logged in, different content will be available for them to see. For example, the navigation menu will contain different links for a logged in user and a user that is not logged in. This is achieved by the functionality in the “User” class which checks if the “userid” is assigned to the “Session” and if so the “\_isLogginIn” condition is “true”. An example of this can be seen in figures 12 and 3.

|  |
| --- |
|  |

Figure Checking if user is logged in



Figure Menu bar when user is not logged in



Figure Menu bar when user is logged in

#### Remember me functionality

The user has the option of being remembered as they log in. Cookie functionally is used to carry out this functionality. If a user chooses to be remembered, a cookie with a hash value is stored on the user’s computer and in the database. The cookie is in the database with the “userid”. When the user visits the website again on the same computer “userid” is assigned into the session and the user will be automatically logged in.

|  |
| --- |
|  |

Figure Cookie stored in the session

#### Logging out

Logging out is essentially the act of deleting a particular session. The “logout” function in the “User” class uses the “Session helper” to delete the session and the cookie assigned to the user. The cookie hash is also deleted from the user session. Without deleting the cookie hash the user would stay logged in.

|  |
| --- |
|  |

Figure User session and cookie being deleted

***Permissions system***

The website required a permission system as there would be different users with different access levels on the website. For example, an administrator would have different permissions to a registered user. There are three permissions levels; registered, administrator and authorized. A registered user cannot upload any data, only an authorized or an administrator can. An administrator has the ability to change user’s access level.

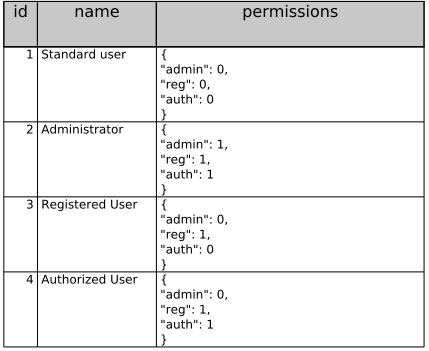


Figure . Table Groups, showing different roles

A user’s access value determines their permissions level. The access value corresponds to the id value in the groups table.

|  |
| --- |
|  |

Figure Users access values

When a user registers there access value is defaulted to “1” which is a registered user. It is done using a “jqgrid”. A user’s permissions level can be changed by a administrator via the “Maintain Registered Users” page. A user can be is selected in the “jqgrid” and their access level can be updated to a different permission level.

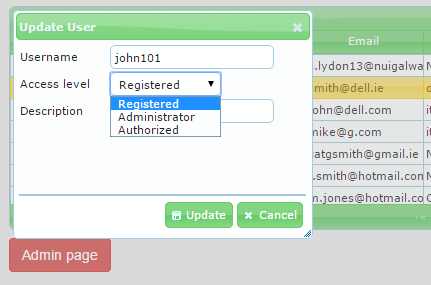


Figure Changing user's permissions level

The permission system is used via the “hasPermission” function which checks the access level of the user. For example, this allows only a user with admin permissions to see the “admin” as there is a check to see if the user has the correct permission otherwise the user is redirected to the homepage.



Figure Checking users permission

|  |
| --- |
|  |

Figure Redirecting user

### Uploading files:

Users that have the suitable permissions have the option to upload files. The “UploadFile” class handles the file uploads. Duplicate files are automatically renamed by inserting a number before the file name extension. Uploads can be restricted to a predefined range of MIME type. The class will automatically append a suffix to the names of files that could cause damage to the website. This includes files such as PHP and JavaScript files. It automatically handles both single, and multiple file uploads. The “filename”, “date” and “userID” are stored in the database.

|  |
| --- |
|  |

Figure Inputting file details into the database

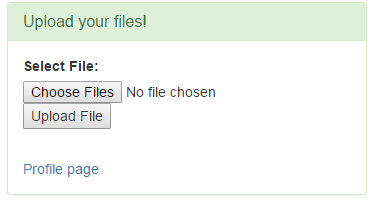


Figure Upload file(s) form

**Downloading Files**

The files can be downloaded on the “Uploaded Files” page. The database is queried to access the filename, date and username. These values are displayed in a table and the file can be downloaded by clicking on the filename. A download dialog is initiated when the filename is clicked.

|  |
| --- |
|  |

Figure Downloading files

|  |
| --- |
|  |

Figure Getting the filename from the database

**Maintaining Wetlands**

CRUD functionality can be carried out on wetlands in the “Maintain Wetlands” page. Wetlands can be added and edited using the “jqGrid”.

|  |
| --- |
|  |

Figure Add/Edit Wetland grid

|  |
| --- |
|  |

Figure Adding a wetland

|  |
| --- |
|  |

Figure Editing a wetland

Querying the Wetland’s Data

|  |
| --- |
|  |

|  |
| --- |
|  |

### Sample page:

The sample page for the constructed wetland database displays the data on the individual wetland. This page contains four tabs: sample data, wetland details, publications and observations.

Each of these four tabs illustrates different pieces of information about the wetland, three of which will be accessible by the unregistered user. The sample data tab contains all of the information gathered from the wetland over time by sample measurements on specific dates. This includes measurements of certain chemicals such as phosphorous as well as results from chemical oxygen demand tests.

The wetland details tab contains images relevant to the wetland as well as a site description giving an overview of the constructed wetland details.

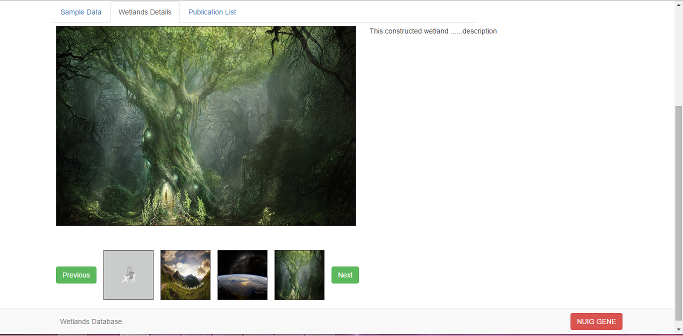


Figure . Wetlands Detail

This tab relies heavily on JavaScript functions to operate traversing through the images, and allows for the user to scroll iterate through the image selection for a wetland, provided there is one. The smaller images below are thumbnails and on mouse over will change the larger picture above them to become the same picture. There is a slight delay effect added here to add a pseudo-animated style to the gallery.

|  |
| --- |
|  |

Figure . JavaScript for scrolling sequence.

Images are uploaded to the site via a form which validates server side the format of the file through PHP. The user must be logged in in order to upload images. If it passes validation, it is uploaded to the server and a database record is created for the image. The database record then contains a link to the image on the server for use in the html table on the wetland details tab. However, before it is shown in the gallery for a specific wetland, it must be validated by the administrator, who verifies that it is suitable. If the image is appropriate, there is a checkbox on the record which will signal that the image is acceptable for viewing. The SQL query on the tab will only link images which have passed this validation. If the administrator does not think that the image is appropriate, then it will be deleted from the database and the server via the same admin page that would allow it. The images are stored on the server because unlike the sample data images sent by researchers, these images will be available to the public and so storing tem in the database through binary encoding or otherwise would result in a sizeable reduction in the databases speed and ease of use. Many popular photo sharing websites store publicly viewable images online using a file system rather than in a database so this seemed like a good idea to prevent too much delay when trying to view images for a wetland. The site description to the right of the images is an overall description for the wetland. This description is non-editable by anyone but the administrator. The administrator may update this whenever, and as it is unique to each wetland, it is simply a field in the Wetland table in the database.

The next tab on the page is the publications listing. This is a table containing the details of all of the relevant articles for the specified wetland.

|  |
| --- |
|  |

This tab contains the PHP which retrieves the literature from the database and outputs it row by row in a table format, which is outputted in order of priority. This arbitrary level of priority is decided by the administrator, who as the client specified, is more likely to hold articles written by the NUI Galway research group in a higher precedence than articles written about the wetland by other researchers. It is allowed by a field in the database record which indicates its importance. It is to encourage traffic to NUI Galway papers. The table uses some JQuery to allow highlighting of the rows on hovering over a row, and also allows sorting of the articles by the year by which they were written. Note that sorting the publications like this takes away the priority that NUIG papers had on initial loading of the tab.

|  |
| --- |
| Figure . simple JQuery for highlighting the row which is being hovered over. |

The articles are not stored on the database or website itself, but rather a DOI is given for each publication, enabling the user to read a publication as long as the paper remains available to the public.

The observations tab is a tab available to view only by registered members of the website. The data from the wetland on the previous three tabs has been purely objective data, which has been observed and researched. This tab is more concerned with incidents that may have happened to the wetland over time, and possible conclusions drawn from that.

The objective data is open to interpretation, but it will only be those who have an interest in the field who will have an understanding for what each field in the sample grid means. Therefore it is unlikely to be misinterpreted by the public. However, observations and incidents which may be perfectly fine in the case of a specific wetland, may be misconstrued by the user who does not know much about the operation of constructed wetlands.

|  |
| --- |
|  |

As such, it is important that the database is not used against NUIG or anyone involved with the maintenance and/or construction of a wetland, so this data must be hidden from unregistered members. This tab is generated in a very similar format to the publications tab, in that it queries the database and retrieves records specific to the wetland ID gained from the user searching the query page. The main difference is that on the sample page, the tab is only shown if the user has logged in through a simple <?php (if($isLoggedIn) ?> verification.

This verification minimises the risk that a user will misunderstand the data and lead to unnecessary difficulties with any parties involved with the wetland.

### Browse Literature:

This page operates in much the same manner as the publications tab for the wetland page, querying the database through PHP and then echoing out rows of articles, if there are any. In this case though, the query is not tied to any specific wetland, it reads and writes all the literature contained in the database. Like the publications tab, it contains row highlighting and sorting by year, as well as a scroll bar for navigating through the body of the table so the user doesn’t have to scroll down the page in order to view all of the literature in the database.

The styling for the table is done through bootstraps CSS and the colour was chosen by creating an image of a colour gradient and using it as an image for the table’s header. It only takes up the top half of the table header columns height wise and so can be positioned on top of the background colour of the row, so that it stands out. The row’s background colours are decided through a php condition which designates the class based on the number of the row.

### Forgotten Password:

If the user has forgotten their password to the system, they will be able to enter this page and input their e-mail address into a form. This form will be passed to a page which sees whether there is an account with the specified email address, and if there is, creates new password for the user. This is a randomly generated string which uses numbers and letters but does not use letters or numbers which are likely to create confusion, such as the number ‘1’ and lower case ‘l’. The password is encoded and given a salt, and replaces the old password in the database. It is sent to the specified email address, along with a message encouraging the user to change their password, to prevent against security leaks. In addition, the person who fills out the form is not notified as to whether an e-mail has been sent or not, as this facilitates potentially unwanted situations where someone mass-inputs email addresses in order to determine the addresses linked to an account. Although this is unlikely to be an issue with a website for a constructed wetland database, it is nonetheless a good practice to take when designing a system, to be safe.

# **Testing**

This document

## System Testing

Task : Weekly review of code to be merged into the Wetlands repository.   
Check that the code chages works prior to integration of source code.

After the code has been merged into the code base – check that the website still works, by going through each page and testing the functionality of the changed webpage.

## Interface User Experience Testing

**Candidates for user experience test :**

County Council Engineer,   
Scientist,

Ordinary person who might have a interest in Wetlands

Undergrad Student

**Aim of UX Test:**

As an engineer I am interested in querying wetlands for County Galway for a particular type of waste water for instance Municipal waste water, and typically Secondary treatment.

What wetlands are in County Galway ?

What data is available for a selected wetlands?

**Observations**:

Were you able to find what you were looking for?  
 Did you have any difficulties finding your way to the information that you were   
 seeking?

Where you satisfied that you could easily search for sample data and download it to   
 an excel spreadsheet.

=============================================================================

## References

***Environmental Science****.* (2014, 11 30). Retrieved from Dolores Gende's AP Sciences Website: http://apesnature.homestead.com/

***Geo-Environmental Engineering Research Group*** <http://www.nuigalway.ie/gene/>

*Constructed Treatment Wetlands*, United States Environmental Protection Agency Available online at <http://www.epa.gov/owow/wetlands/pdf/ConstructedW.pdf>

*Review of Rapid Assessment Methods for Assessing Wetlands Condition* , United States Environmental Protection Agency Available online at <http://water.epa.gov/type/wetlands/assessment/upload/monitoring-rapid-method-review.pdf>

# APPENDIX A Project Progress Reports

**Project Kickoff Meeting**

* Discussed what approach to take to working together as a team, what prior experience and technical skills do we each bring to the project.
* What do we want to accomplish? What skills do we want to develop or refine?
* Discussed what was technically feasible:   
  Which programming language(s) and operating systems that we could develop the project on. What is our knowledge / skillset gap and what effort / time required to get up to speed on the chosen development environment.
* Discussed the problem brief.  
  Came up with a list of questions to ask our client on our initial meeting.  
  *see Appendix C. questions for interview 1*.

**Meeting Outcome**

Identifying Key stakeholders : Mark Healy (Client), Collette Mulkeen (Expert user/Administrator)

Set up meeting with supervisor: Josephine Griffith

##### Client Meetings:

28/10/2014 : Meeting with Client

Client supplied a example Samples Spreadsheet : <http://1drv.ms/1avc4lG>

**Meeting Outcome**

Discussion about Wetlands Data: What are the characteristics and meaning of the sample data, how it is collected, by whom. What are the important functions that the user would required of the website.  
What is the business purpose for the project – how would it benefits the client. How urgent is it.

28/10/2014 : Meeting with Client

Client supplied a example Samples Spreadsheet : <http://1drv.ms/1avc4lG>

**Meeting Outcome**

Negotiation on the upload functionality. It was decided that the county council engineers would only engage with the upload of data function, if all they were required to do was browse to a file located on their local drive and press upload button on the website to copy it onto our server.

10/11/2014 : Meeting with Collette Mulkeen.   
Discussed how data is collected at the wetlands site and how the samples are analysed. Someone visits the site daily to check the wetland boundaries and site conditions. Depending on the set up for the constructed wetlands they may be able to take daily flow rate reading on site.   
  
Regular sampling of the inlet and outlet points are required under the terms of the EPA licence. These samples collected by site engineers are sent to a lab.   
The report sent back from the lab provides the indicators on wetland performance in terms of measurements of constituents found in the sample.  
Currently, Collete has been collating these lab reports and enters the sample analysis figures into a spreadsheet.

22/11/2014 : Meeting with Client

Discussed the data in terms of how it should be presented on the site.  
  
11/2/2015 : Meeting with Client

Review of Wireframe Prototype. Discussion about how to present the sample view, clarification on labels, scrolling horizontally, filtering by date.  
Discussion on the Wetlands Detail tabs and what information to show with the gallery of photos.  
Discussion of permissions for logged on users to view the Observation related to a site inspection.  
Discussed the interaction and different layouts for presenting the data.

20/2/2015 : Collette provided more data for the website: Sample Data and Publication list

28/2/2015 : Ciaran met with Mark about the functionality around publications list, viewing and maintaining Literature.

# APPENDIX B Source Change Control

Link to repository : <https://github.com/danugc4/WetlandsRepository>

|  |
| --- |
|  |

# APPENDIX B Preparing for Interviews

Questions Interview 1

22 October 2014

11:52

|  |  |
| --- | --- |
| Describe the Users | County Council technicians |
| Describe the key concerns that the system should address for each user | Who will be using the web site ?   What types of user would be accessing and using this system ?  For each type of user, what aspects of the data would they be concerned with? |
| Describe how the data is collected - samples onsite, sent to labs |  |
|  | What information would be most important to the users? |
| Understand the Data involved | Sources/ Sinks of information ?    What external systems need to be interfaced to? |
| Submit Data | What is the process for submitting data?    How is it collected? By whom?  What form does the data come in? |
| Wetland Performance | What are the performance indicators that need to be calculated? |
| Understand the processes involved | What typical searches would be essential to be made available on the website?  What services does the application need to provide |

Questions for interview 2

05 November 2014

11:56

|  |  |
| --- | --- |
| Concern with size of spreadsheet - how it will grow and how that can be uploaded | There will be approx 140 sheets - how will that be uploaded  How much would that grow with samples for each month |
| Identify stakeholders / roles | ACTORs 1 - general public ACTORs 2 - Engineer , Environmental Scientist,   Students, Local Interest Groups |
| How users will use the website | If they saw the tabulated form would that make sense to them?    What sort of questions would they have in mind when they come to the website    Would they be comparing wetlands?  What would they expect to see?   Can you describe the scenario where actor 1 / actor 2 visits the web site. |
| GET TERMINOLOGY - glossary | Clarify the 8 important properties …. Would we be inputting the same statistics - or different element(sample components) |
| Graphical Representation | What graphs do you have on the spreadsheet?  Is that something that would be useful to show on line? |

1. <http://apesnature.homestead.com/chapter17.html> [↑](#footnote-ref-1)
2. <http://apesnature.homestead.com/chapter17.html> [↑](#footnote-ref-2)
3. <http://apesnature.homestead.com/chapter17.html> [↑](#footnote-ref-3)