|  |
| --- |
| Group 2  Shawn Sabraw  Rob Bishop  Asma Javaid  Praveena Pinnika  Samira Saki |

|  |
| --- |
| COMP-6905 |
| Nature’s Palette |
| Assignment 1 |

**Table of Contents**

1. **Vision Document**
2. **Use Cases**
   1. **System Use Case**
   2. **Brief Descriptions**
   3. **Detailed Descriptions**
3. **Entity, Boundary, and Control Objects**
4. **Sequence Diagrams**
5. **Design Goals**
6. **System Decomposition**
7. **Logical Architecture**

**Vision Document**

*Introduction*

For this project, the aim is design and implement a web application that will allow users to upload, search, and download spectral data that can be used in research and additional projects through useful search tools and data itself.

*Problem Statement*

Research projects that require quantification of colours usually require thousands of spectral measurements. The issue in this; for researchers, is obtaining all that data. For example, comparative studies in birds have characterized the plumage colours for over 40% of described species of birds. However, less than 5% of that data is publicly available to other researchers. The applications that this data is available through is not designed in a way that makes them easy to use and/or provide the metadata that would allow that information to be used in other studies. Consequently, the possibilities for larger studies are very limited and lots of resources are being wasted in order to obtain (usually duplicating) the data for a different research project.

*Stakeholders & Key Interests*

|  |  |
| --- | --- |
| **Stakeholders** | **Key Interests** |
| Researchers  (Professors/Students) | Using the data provided in their projects |
| Hobbyists/Users | Adding their data to be used in research projects |
| Establishments | Access to a large database of spectral information for R&D |

*Users & User-Level Goals*

|  |  |
| --- | --- |
| **User** | **Goals** |
| General User  (Researchers/Hobbyists) | Login, Upload/edit/search/download spectral data, Refine search through meta data/georeferencing/colour range, Upload many files, … |
| Administrator | Login, Upload/edit/search/download/remove spectral data, Edit user’s access rights, Edit parts of the application, Adding meta data tags, … |

*Summary of System Features*

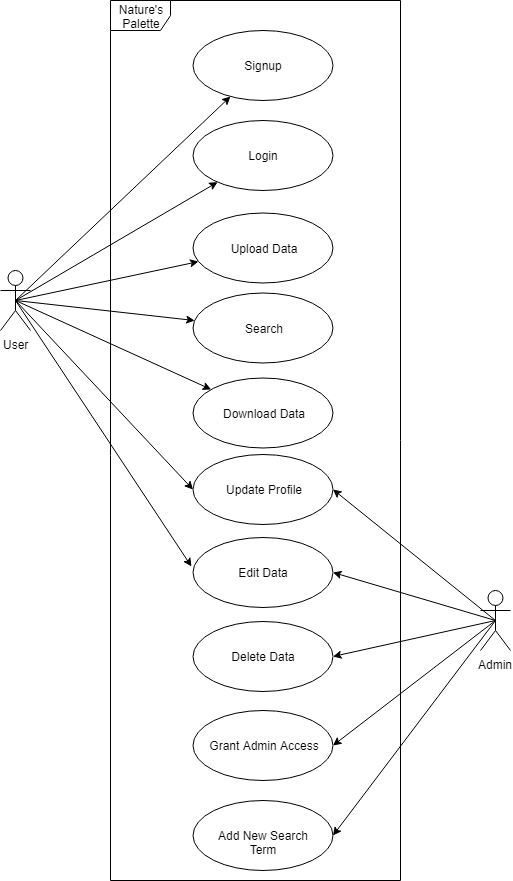
* The application will allow users to upload spectral data with it’s meta data tags.
* The application will let users download the data as well as it’s meta data.
* The application will let users search through the data using georeferencing, meta data tags, and specifying a colour range.
* The application will allow users to edit their uploads in case of mistakes.

*Project Risks*

* Utilizing and learning Node.js, MongoDB and other modules as a client -> server model.
* Mass uploading of thousands of files as well as storing them in a way to allow for scaling.
* Using a Python/R package in order to “clean” spectral data that is being uploaded.
* Using a polygon georeferencing tool to search meta data in a database.

**Use Cases**

*System Use Case*



*Brief Descriptions*

|  |  |
| --- | --- |
| **Use case name** | **Description** |
| *SignUp* | Users enters in the required information (username, password, etc.…) to create an account. |
| *Login* | Users enters in the required information to gain access to their account. |
| *Update Profile* | Users can update their profile details. Admins are also able to edit their own as well as other user’s details. |
| *Upload* | Users can upload their spectral data accompanied with their meta data. |
| *Search* | Users can search for spectral data giving metadata as input terms. |
| *Download* | Users can download the spectral and metadata provided. |
| *AddNewSearch* | Admins can add new search terms for the users to search by. |
| *AdvancedSearch* | User can search for spectral data by providing multiple filters (regions, colour space, etc.…) given in the advanced search. |
| *ChangeAccess* | Admins can change a user’s access to ban, unban, or make them an admin. |
| *Edit Data* | Users (with an account) can edit or delete their own data. Admins can edit and delete any data. |

*Detailed Descriptions*

# Use Case: Upload

|  |  |
| --- | --- |
| **Use Case Name** | Upload |
| **Participating Actors** | User |
| **Flow of Events** | **Main**   1. User is prompted to upload the specified files. 2. Select appropriate files to upload and continues. 3. After file upload, the spectral data will be compared to the metadata. 4. The data is confirmed with the metadata and is saved to the repository before displaying a success status message to the user. (**Alt1**)   **Alt1**   1. The data and metadata do not match. The data is not saved, and a failure status message is displayed to the user. |
| **Entry Condition** | * Clicks upload button |
| **Exit Condition** | * User clicks “x” or navigates away from the upload prompt |
| **Quality Requirements** | * Upload should be fast for the user; the server should do most of the work. |

# Use Case: Search

|  |  |
| --- | --- |
| **Use Case Name** | Search |
| **Participating Actors** | User |
| **Flow of Events** | **Main**   1. The user enters a query. 2. The webpage communicates the query to the server. 3. A series of entries that match the user’s query are displayed, each with some information about the entry. (**Alt1**)   **Alt1**   1. A status message is displayed saying that no data matched the user’s query. |
| **Entry Condition** | * User clicks on the search field |
| **Exit Condition** | * The user navigates away from the page |
| **Quality Requirements** | * Search should take place within a reasonable speed (To be determined by client) |

# Use Case: Download

|  |  |
| --- | --- |
| **Use Case Name** | Download |
| **Participating Actors** | User |
| **Flow of Events** | **Main**   1. The user is prompted for the download. 2. The user selects and downloads the spectral data. (**Alt1, Alt2**)   **Alt1**   1. The user selects and downloads the meta data.   **Alt1**   1. The data is not available yet for download. |
| **Entry Condition** | * Clicks download button |
| **Exit Condition** | * The download finishes or the user navigates away from the page |
| **Quality Requirements** | * Download should be available quickly |

**Entity, Boundary, and Control Objects**

*Entities:*

* User
* Admin
* Server

*Boundaries:*

* Webpage interfaces

*Controls:*

* Upload
* Search
* Download
* SignUp
* Login

**Sequence Diagrams**

**Design Goals**

**System Decomposition**

**Logical Architecture**

The architecture for this application could potentially use the Client/Server architectural style. However, we need to use the repository style as well, so in order to do this we have chosen to use the Three-Layer Architectural Style.

* A user interface layer (Webpage), which will be allow for interaction with the second layer as well as displaying of any information.
* A middleware layer (NodeJS), which is responsible for accepting any requests and of processing of data before it is sent to the first(webpage) or third(MongoDB) layer.
* A database layer (MongoDB), which responds with data queried to/from the second(NodeJS) layer.

