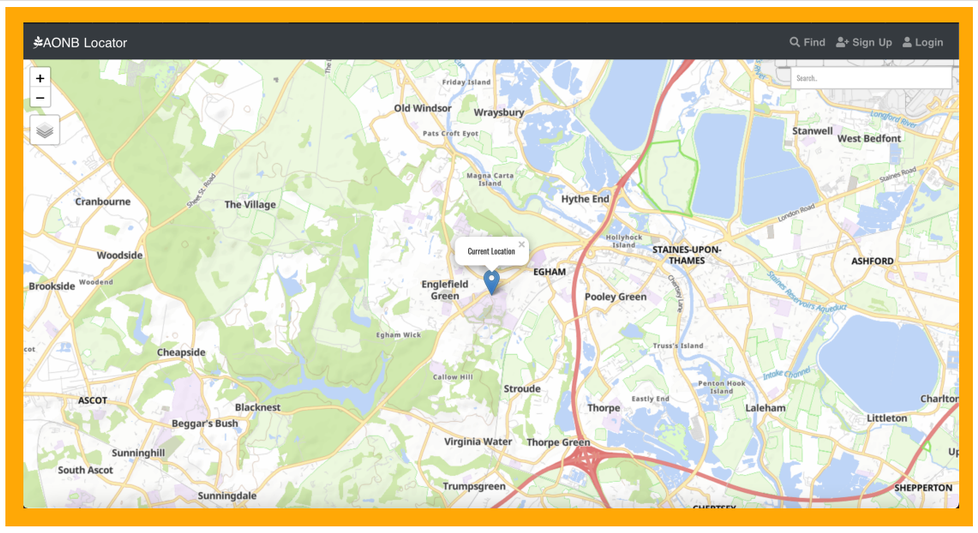
Offline HTML5 Map Applications

Jack Wearn

Offline HTML5 Map

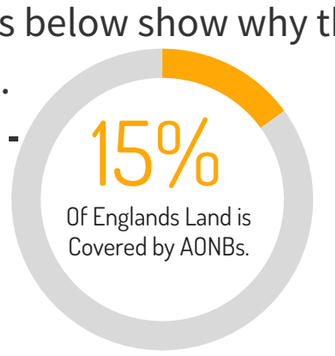
The map application works both online and offline. This means that a user can continue to use the map even whilst not being connected to the Internet, such as when travelling within an AONB location.



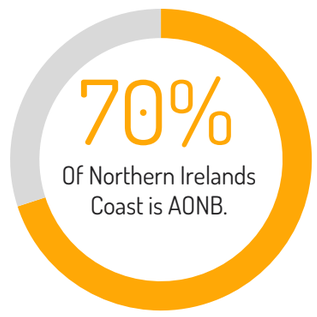
Created using OpenStreetMap and Mapbox API

Focusing on AONB Locations

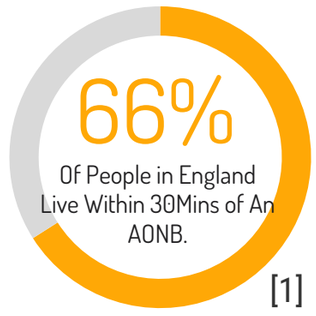
The map that has been created focuses on AONB Locations across the United Kingdom. The figures below show why the decision to focus on AONB Locations was made.



Of Englands Land is Covered by AONBs.



Of Northern Irelands Coast is AONB.

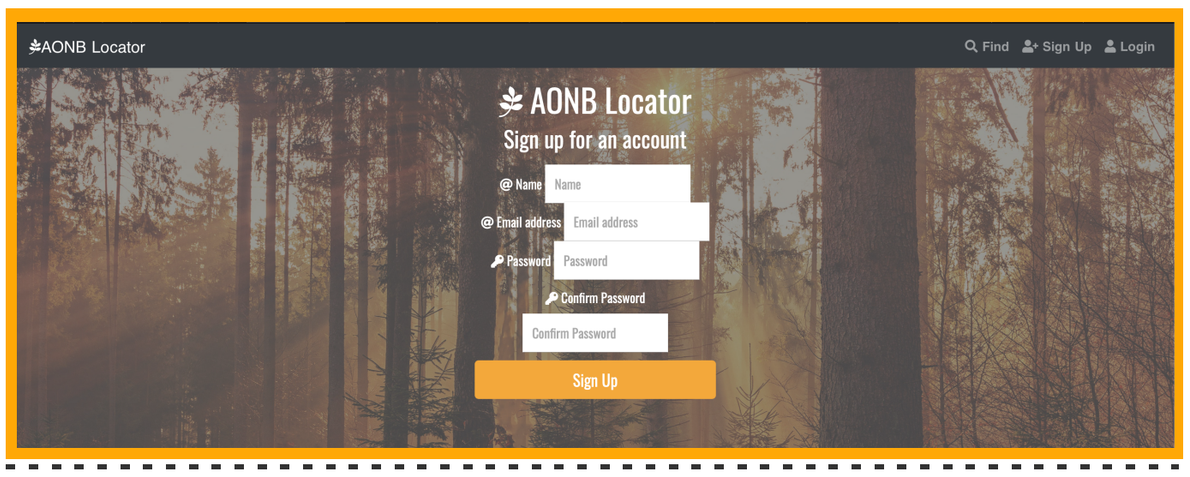


Of People in England Live Within 30Mins of An AONB.

[1]

Create An Account & Leave a Comment

When using the map application, a user is able to create an account and login once they have made one. When interacting with the site, they will know if they are logged in. A user can also leave a comment on any of the AONB information pages.




```
(function(){
    if ('serviceWorker' in navigator) {
        console.log('CLIENT: service worker registration in progress.');
        navigator.serviceWorker.register('service-worker.js').then(function() {
          console.log('CLIENT: service worker registration complete.');
        }, function() {
          console.log('CLIENT: service worker registration failure.');
        });
    } else {
        console.log('CLIENT: service worker is not supported.');
    }
})();
```

Offline Capabilities Via Service Workers

The use of service workers has allowed for the map web application to work both online and offline.

When the user loads into the website, the service worker begins running and starts to cache the required files and media, including the map and a number of extra map tiles.

This means that when the user loses Internet connection, they can still use the whole web application and even move around the map further than what they was just looking at.

Backend Database Handled by MySQL

A user is able to create an account, login and also leave a comment on any AONB information webpage. All this data is stored within a MySQL database which interacts with the PHP frontend to allow users to connect and query the database. The PHP code is able to use SQL queries to perform user login and user creation. Which has a number of security features to keep a users data secure.

```
mysql> SELECT * FROM comments;
+----+---------+-----------+------+-----------------+---------------------+
| id | page_id | parent_id | name | content         | submit_date         |
+----+---------+-----------+------+-----------------+---------------------+
|  4 |       1 |        -1 | Test | I liked it here! | 2021-03-23 23:04:38 |
+----+---------+-----------+------+-----------------+---------------------+
1 row in set (0.00 sec)

mysql> SELECT * FROM users;
+---------+-----------+--------------------------------------------------------------+-----------------------+
| user_id | user_name | user_password_hash                                           | user_email            |
+---------+-----------+--------------------------------------------------------------+-----------------------+
|       1 | Test      | $2y$10$FI4Kd7uk4uw4DlB4S/Af5eGQRdW5g6pNMEVVdwx5PeHco1b.icCb. | testuser@testing.com |
+---------+-----------+--------------------------------------------------------------+-----------------------+
1 row in set (0.00 sec)
```

```
if (empty($_POST['user_name'])) {
    $this->errors[] = "Empty Username";
} elseif (empty($_POST['user_password_new']) || empty($_POST['user_password_repeat'])) {
    $this->errors[] = "Empty Password";
} elseif ($_POST['user_password_new'] !== $_POST['user_password_repeat']) {
    $this->errors[] = "Password and password repeat are not the same";
} elseif (strlen($_POST['user_password_new']) < 6) {
    $this->errors[] = "Password has a minimum length of 6 characters";
} elseif (strlen($_POST['user_name']) > 64 || strlen($_POST['user_name']) < 2) {
    $this->errors[] = "Username cannot be shorter than 2 or longer than 64 characters";
} elseif (!preg_match('/^[a-z\d]{2,64}$/i', $_POST['user_name'])) {
    $this->errors[] = "Username does not fit the name scheme: only a-Z and numbers are allowed, 2 to 64 characters";
} elseif (empty($_POST['user_email'])) {
    $this->errors[] = "Email cannot be empty";
} elseif (strlen($_POST['user_email']) > 64) {
    $this->errors[] = "Email cannot be longer than 64 characters";
} elseif (!filter_var($_POST['user_email'], FILTER_VALIDATE_EMAIL)) {
    $this->errors[] = "Your email address is not in a valid email format";
} elseif (!empty($_POST['user_name'])
    && strlen($_POST['user_name']) <= 64
    && strlen($_POST['user_name']) >= 2
    && preg_match('/^[a-z\d]{2,64}$/i', $_POST['user_name'])
    && !empty($_POST['user_email'])
    && strlen($_POST['user_email']) <= 64
    && filter_var($_POST['user_email'], FILTER_VALIDATE_EMAIL)
    && !empty($_POST['user_password_new'])
    && !empty($_POST['user_password_repeat'])
    && ($_POST['user_password_new'] === $_POST['user_password_repeat'])
) {
    $sql = "SELECT * FROM users WHERE user_name = '" . $user_name . "' OR user_email = '" . $user_email . "';";
    $query_check_user_name = $this->db_connection->query($sql);
}
```

Secure User Account Creation

When a user attempts to login and create an account, their details entered are securely added to the MySQL Database. There are a number of security features in place that have been created to ensure that an attempt to hack into the database is not being made and that anything they enter will not 'break' the database such as duplicate records.

The image on the left shows some of the checks that are made before a user is registered onto the database, such as checking all of the required fields and then hashing the users password.

Managed With GitHub, Deployed through Heroku

Check Out The Web Application! ->

[1] https://landscapesforlife.org.uk/application/files/3415/5586/2985/NAAONB-Response-to-Glover-Review-of-Designated-Landscapes.pdf