Secure Web Application: Secure Enterprise Blog

NCI Post Graduate Diploma Cyber Security

Author: Jonathon Taaffe

Title: Secure Web Application: Secure Enterprise Blog
Author: Jonathon Taaffe

Warning and Disclaimer
The information is provided on an “as is” basis. The author shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this publication. The opinions expressed in this publication belong to the author.

Trademark Acknowledgments
All terms mentioned in this publication that are known to be trademarks or service marks have been appropriately capitalised. The author cannot attest to the accuracy of this information. Use of a term in this publication should not be regarded as affecting the validity of any trademark or service mark.

Contents

Introduction

Executive Summary

Email and Instant Messaging are the client's existing digital communication applications which allow for one-to-one or one-to-many communications. With a Secure Enterprise Blog Application, employees will be able communicate one-to-all employees. A Secure Enterprise Blog Application will enable easy sharing of information and will allow enterprise users to blog using an intuitive User Interface.

Blog Users will first have to securely authenticate to access the application before they can add, update or delete their blog content. Authorisation and Access to the Secure Enterprise Blog Web Application will be managed through Role Based Access Control (RBAC)[1]. The following RBAC Roles have been configured:

- Bloggers: Users in this role will be able to add, update or delete their own blog content.
- Administrators: Administer all blog user accounts, all blog content and access log information.
- Super Users: Administer all Administrator accounts, manage application updates, manage password policy and access log information.

The existing Internet Usage Policy Terms and Conditions have been updated to include the blog application. Users will have to agree to the Usage Policy Terms and Conditions prior to using the blog application.

Background

This Secure Enterprise Blog Application will enable employees to communicate work related information to all employees.

Users will first have to login to the application with a unique username and use a complex password. The existing corporate Password Policy has been applied to this application. Once logged in, users will be able to add, update or delete their own blog content.

Administrators will create, update and delete all Bloggers accounts. Administrators will also have access to mediate (add, update or delete) all blog content and will have access to Application Event and Access Logs.

Super Users will create, update and delete all Administrator accounts. Super Users will also manage system and/or software updates, manage the application password policy and will also have access to the Application Event and Access Logs.

Goals

The goal is to enable employees to securely communicate work related information to all employees. All users of the application must first securely log in to the application. Once logged in, the User Interface displayed will be determined by the RBAC Role in which the user account resides. For the Bloggers RBAC Role, the following actions are available through an intuitive User Interface:

- Read all Blog posts
- Create a new Blog post
- Publish a new Blog post
- Update one of their own Blog posts
- Delete one of their own Blog posts

[1] Wikipedia.org [2019] Role-Based Access Control https://en.wikipedia.org/wiki/Role-based_access_control [Accessed 1st August 2019]

Architecture

Development Environment

The following platform was used to develop and test the Blog application:

Operating System	Microsoft Windows 10 Build 18362
Application Development Stack	XAMPP for Windows v7.3.7[2] including: Apache 2.4.39 MariaDB 10.3.16 PHP 7.3.7 OpenSSL 1.1.1
Windows Defender Firewall	Apache HTTP Server Private Network Communication Only MySQL Private Network Communication Only
Framework	Laravel 5.7[3] PHP Framework

Table 1. Development Environment Configuration[4]

Development Database Configuration

The MariaDB development database was configured as follows:

Database Name	Seb
Database Type	MariaDB 10.3.16
Hostname	Localhost
MySQL Port	3306
Encoding	UFT8_General_CI
Authentication	Native MySQL
Password Policy	16 Characters including: 2 Numbers, 2 Lower Case, 2 Upper Case
User account	'seb_dba'@'localhost'
MySQL Login	seb_dba
Encryption Key	32 Character Random Key

Table 2. Development Database Configuration[5]

[2] ApacheFriends.org [2019] *XAMPP Download* https://www.apachefriends.org/download.html [Accessed 1st August 2019]

[3] Laravel.com [2019] *Laravel.com* https://laravel.com [Accessed 1st August 2019]

[4] Taaffe, Jonathon [2019] Table 1. Development Environment Configuration [Created 1st August 2019]

[5] Taaffe, Jonathon [2019] Table 2. Development Database Configuration [Created 1st August 2019]

Production Network

To access the application users will require an internet connection and a secure web browser.
The following network services will be required for the production deployment of the application

Certificate

- Symantec Verisign SSL SHA256RSA (4096-Bit)[6]

Network Firewall

- Internet-facing Outer Firewall
  - Cisco ASA 5500-X NGFW[7]
  - Palo Alto PA-5220 NGFW [8]
- Inner Firewall
  - Cisco ASA 5500-X NGFW[6]
  - Palo Alto PA-5220 NGFW[7]

  Note: This application will utilise the existing production network firewall infrastructure. 2 different network hardware and services vendors are used to ensure defence diversity.

Web Application Firewall

- Cloudflare WAF[9]
- Cloudflare DDoS Prevention[10]

  Note: Cloudflare have a globally distributed cloud network platform and utilise a layered security approach combining multiple mitigation capabilities into one service.

All Network Hardware and Services will be managed by the IT Network Operations team.

[6] Symantec Verisign [2019] SSL/TLS Certificates https://www.websecurity.symantec.com/ssl-certificate[Accessed 1st August 2019]
[7] Cisco [2019] Cisco ASA Next Generation Firewalls https://www.cisco.com/c/en/us/products/security/asa-firepower-services/index.html [Accessed 1st August 2019]
[8] Palo Alto Networks [2019] Palo Alto Networks Next-Gen Firewalls https://www.paloaltonetworks.com/products/secure-the-network/next-generation-firewall [Accessed 1st August 2019]
[9] Cloudflare [2019] Cloud Web Application Firewall https://www.cloudflare.com/waf/ [Accessed 1st August 2019]
[10] Cloudflare [2019] Advanced DDoS Protection https://www.cloudflare.com/ddos/ [Accessed 1st August 2019]

Production Servers

The following server hardware, operating system and applications will be required for the production deployment of the application

Server Function	Front-end Web Server
Operating System	Ubuntu Server 18.04.03 LTS[11]
Application	Apache Tomcat 2.4.3[12]
OS Drive	2 X 120GB with RAID 1 Mirrored
Data Drive	Not applicable
Memory	32GB RAM

Table 3. Front-end Web Server Configuration[13]

Server Function	Back-end Database Server
Operating System	Ubuntu Server 18.04.03 LTS[11]
Application	MySQL Server 8.0[14]
OS Drive	2 X 120GB with RAID 1 Mirrored
Data Drive	5 X 120GB with RAID 5
Memory	32GB RAM

Table 4. Back-end Database Server Configuration[15]

Server Function	Back-end Application Server
Operating System	Ubuntu Server 18.04.03 LTS[11]
Application	PHP 7.3.7[16]
Framework	Laravel 5.7 PHP Framework[17]
OS Drive	2 X 120GB with RAID 1 Mirrored
Data Drive	Not applicable
Memory	32GB RAM

Table 5. Back-end Application Server Configuration[18]

All Server Hardware and Services will be managed by the IT Server Operations team. All Servers will be enrolled in the IT Server Operations Patching process. This will ensure all certified and required Operating System and Application updates are applied in a timely manner.

[11] Ubuntu.com [2019] Download Ubuntu Server https://ubuntu.com/download/server [Accessed 1st August 2019]
[12] Apache.org [2019] Apache Tomcat https://tomcat.apache.org [Accessed 1st August 2019]
[13] Taaffe, Jonathon [2019] Table 3. Front-end Web Server Configuration [[Created 1st August 2019]
[14] MySQL.com [2019] MySQL.com https://www.mysql.com [Accessed 1st August 2019]
[15] Taaffe, Jonathon [2019] Table 4. Back-end Database Server Configuration [Created 1st August 2019]
[16] PHP.net [2019] PHP.net https://www.php.net [Accessed 1st August 2019]
[17] Laravel.com [2019] Laravel.com https://laravel.com [Accessed 1st August 2019]
[18] Taaffe, Jonathon [2019] Table 5. Back-end Application Server Configuration [Created 1st August 2019]

Application Development Framework

The secure enterprise blog application is built on the Laravel PHP Framework. The Laravel PHP Framework was developed on the Symfony PHP Web Application Framework.

Laravel PHP Framework was chosen for the following functionality[19]:

- Reusable PHP components and libraries which decreases web application development and maintenance time and minimises repetitive coding.
- Low performance overhead with bytecode cache.
- Used for building robust enterprise applications.

Laravel PHP Framework features utilised for this secure enterprise application include:

- **SQL Query Builder:** Direct database access using classes and methods to build queries programmatically[15]
- **Eloquent ORM:** ActiveRecord database implementation where each table has a corresponding "Model" which is used to interact with that table[20]
- **Class Auto Loading:** Automated on-demand loading of required PHP classes[15]
- **Unit Testing:** Used to detect and prevent framework reversions[15]
- **Middleware:** Used to filter incoming HTTP requests, e.g. all requests with user authentication verification[21]
- **Authentication:** Laravel's authentication utilises "guards" and "providers"[22]
    - Guards: define how users are authenticated for each request, e.g. session guard uses session storage and cookies to maintains state.
    - Providers: define how users are retrieved from storage.
- **Authentication Controllers:** Pre-built controllers that manage new user registration, authentication and password resets[23]
- **Authentication Event Logging:** Authentication process events generated and logged[24]
- **Authorisation:** All user actions are authorised through middleware[25]
- **Password Storage:** Password column in database auto-configured for 255 characters to allow for encryption, hashing and salting[26]
- **Encryption:** Data is encrypted using OpenSSL AES-256 with a 32 random character key. Encrypted data is signed with Message Authentication Code (MAC) to detect any modifications to the encrypted data[27]
- **Hashing:** All user passwords stored using secure Bcrypt hashing[28]
- **Cross Site Request Forgery (CSRF) Protection:** Prevents unauthorized commands being performed on behalf of an authenticated user by utilising a secret secure CSRF token mechanism[29]

---

[19] Wikipedia.org [2019] Laravel https://en.wikipedia.org/wiki/Laravel [Accessed 1st August 2019]
[20] Laravel.com [2019] Eloquent https://laravel.com/docs/5.8/eloquent [Accessed 1st August 2019]
[21] Laravel.com [2019] Middleware https://laravel.com/docs/5.8/middleware [Accessed 1st August 2019]
[22] Laravel.com [2019] Authentication https://laravel.com/docs/5.8/authentication [Accessed 1st August 2019]
[23] Laravel.com [2019] Authentication Quickstart https://laravel.com/docs/5.8/authentication#authentication-quickstart [Accessed 1st August 2019]
[24] Laravel.com [2019] Events https://laravel.com/docs/5.8/authentication#events [Accessed 1st August 2019]
[25] Laravel.com [2019] Authorisation https://laravel.com/docs/5.8/authorization [Accessed 1st August 2019]
[26] Laravel.com [2019] Database Considerations https://laravel.com/docs/5.8/authentication#introduction-database-considerations [Accessed 1st August 2019]
[27] Laravel.com [2019] Encryption https://laravel.com/docs/5.8/encryption [Accessed 1st August 2019]
[28] Laravel.com [2019] Hashing https://laravel.com/docs/5.8/hashing [Accessed 1st August 2019]
[29] Laravel.com [2019] CSRF Protection https://laravel.com/docs/5.8/csrf [Accessed 1st August 2019]

Software Architecture

Laravel PHP Framework provides the structure and solutions required for the application to function. Laravel utilises the Model-View-Controller (MVC) architecture[30]

For the development of this secure enterprise application the following technologies were used:

- **Cascading Style Sheets (CSS):** Cascading Style Sheets are used to style the User Interface including fonts, colours, spacing, etc. of the Application.[31]
- **HTML:** Hypertext Markup Language for content to be displayed in a web browser[32]
- **PHP:** Scripting language for web application development which can be embedded into HTML[33]
- **JavaScript:** Scripting language used to display dynamic content on web pages[34]
- **JSON:** JavaScript Object Notation is a syntax for storing and exchanging data converting JavaScript objects into JSON reducing complicated parsing and translations[35]
- **Twig:** Flexible, fast and secure template engine for PHP[36]
- **AJAX Framework:** Dynamically send and receive server-side data asynchronously[37]

[30] Wikipedia.org [2019] Model-View-Controller https://en.wikipedia.org/wiki/Model–view–controller [Accessed 1st August 2019]
[31] W3.org [2019] Cascading Style Sheets https://www.w3.org/Style/CSS/Overview.en.html [Accessed 1st August 2019]
[32] Wikipedia.org [2019] HTML https://en.wikipedia.org/wiki/HTML [Accessed 1st August 2019]
[33] PHP.net [2019] What is PHP? https://www.php.net/manual/en/intro-whatis.php [Accessed 1st August 2019]
[34] Wikipedia.org [2019] JavaScript https://en.wikipedia.org/wiki/JavaScript [Accessed 1st August 2019]
[35] W3Schools.com [2019] JSON – Introduction https://www.w3schools.com/js/js_json_intro.asp [Accessed 1st August 2019]
[36] Symfony.com [2019] Twig https://twig.symfony.com [Accessed 1st August 2019]
[37] Wikipedia.org [2019] AJAX Programming https://en.wikipedia.org/wiki/Ajax_(programming) [Accessed 1st August 2019]

System Requirements

Documented below are all the function requirements for the Secure Enterprise Blog Application. The functional requirements detail the behaviour of the application when a user is interacting with it.

Browser Security

To access the application, clients must be using a secure up-to-date browser. As per policy, the default browser for all enterprise users is Google Chrome[38]. The following security options have been configured which will apply to all browsers when connecting to the application:

- End-2-End TLS/SSL Communication
- Same Origin Policy (SOP)
- Cookie Domain: www.SEB.com (Production URL)
- Cookie Secure: Transmit over HTTPS Connection Only
- HttpOnly: HTTP Only Access to Cookie Data

Core Functional Requirements

When accessing the URL of the application, all users will be presented with a login page. All users must first successfully authenticate before they can access the application. Authorisation is provided through a Role Based Access Control (RBAC) system which determines the UI presented when the user logs in dependant on which role the user account has been assigned to.

All passwords are securely stored in a MySQL database using a 32-bit random character key, encrypted with Advanced Encryption Standard (AES) 256 Bit Cipher Block Chaining (CBC)[39] and hashed using secure Bcrypt[40].

Core Functional requirements for all users of the application are:

- View Blog Posts
- Manage Own Blog Posts
- Publish Own Blog Posts

Security Functional Requirements

Role Based Access Control is used to manage access to the application. The RBAC roles configured to manage access to the application include:

- Bloggers: Users in this role will be able to add, update or delete their own blog content.
- Administrators: Administer all blog user accounts, all blog content and access log information
- Super Users: Administer all Administrator accounts, manage application updates, manage password policy and access log information.

Security Functional requirements of the application are:

- Manage Access through Authentication and Authorisation
- Guarantee Privacy and Confidentiality
- Guarantee Integrity of all sensitive data
- Guarantee Availability of the application
- Guarantee non-repudiation of all data

[38] Google.com [2019] Chrome https://www.google.com/chrome/ [Accessed 1st August 2019]
[39] Wikipedia.org [2019] Advance Encryption Standard https://en.wikipedia.org/wiki/Advanced_Encryption_Standard [Accessed 1st August 2019]
[40] Wikipedia.org [2019] Bcrypt https://en.wikipedia.org/wiki/Bcrypt [Accessed 1st August 2019]

Actions Permitted per Use Case

Role: Bloggers		
Area	**Use Case**	**Description**
Security	Login Authentication	Securely login to the application
Security	View Account Details	View the account details of the logged in user
Security	Modify Account Details	Modify the account details of the logged in user
Security	Securely Logout	Securely logout of the application
Blog Content	View All Blog Posts	View all posted blog posts
Blog Content	View Own Blog Posts	View blog posts the user has posted
Blog Content	Modify Own Blog Posts	Modify all blog posts the user has posted
Blog Content	Create New Blog Post	Create new blog post under logged in account
Blog Content	Publish New Blog Post	Publish new blog post for all employees to view

Role: Administrators		
Area	**Use Case**	**Description**
Security	Login Authentication	Securely login to the application
Security	View Account Details	View the account details of the logged in administrator
Security	Modify Account Details	Modify the account details of the logged in administrator
Security	Create User Accounts	Create new user accounts in the Bloggers role
Security	Activate User Accounts	Activate new user accounts in the Bloggers role
Security	Modify User Accounts	Modify user accounts with the Bloggers role
Security	Delete User Accounts	Delete user accounts with the Bloggers role
Security	View Event Logs	View the application event logs
Security	View Access Logs	View the application access logs
Security	Securely Logout	Securely logout of the application
Blog Content	View All Blog Posts	View all blog posts all users have posted
Blog Content	Update All Blog Posts	Update all blog posts all users have posted
Blog Content	Create New Blog Post	Create new blog post under logged in account
Blog Content	Publish New Blog Post	Publish new blog post for all employees to view
Blog Content	Export Blog Posts	Export all blog posts to a file
Blog Content	Import Blog Posts	Import blog posts from a file
Blog Content	Delete Blog Posts	Delete any blog post any user has posted

Role: Super Users		
Area	**Use Case**	**Description**
Security	Login Authentication	Securely login to the application
Security	View Account Details	View the account details of the logged in super user
Security	Modify Account Details	Modify the account details of the logged in super user
Security	Create Administrator Accounts	Create new Administrator accounts
Security	Activate Administrator Accounts	Activate new Administrator accounts
Security	Modify Administrator Accounts	Modify Administrator accounts
Security	Delete Administrator Accounts	Delete Administrator accounts
Security	View Event Logs	View the application event logs
Security	View Access Logs	View the application access logs
Security	Manage Roles	Manage the role configured in the application
Security	Manage Password Policy	Manage the password policy for the application
Security	Securely Logout	Securely logout of the application
System	Manage Software Updates	Manage software updates to the application
System	Manage Backend Configuration	Manage the administration console of the application

Design and Architecture

Network and Server Infrastructure Architecture

The application utilises 3 servers to function: a front-end web server, a back-end database server and a back-end application server.

As the application will be accessible from the internet, the back-end servers need to be located behind network firewalls and a web application firewall to ensure malicious network packets are dropped before reaching the servers.

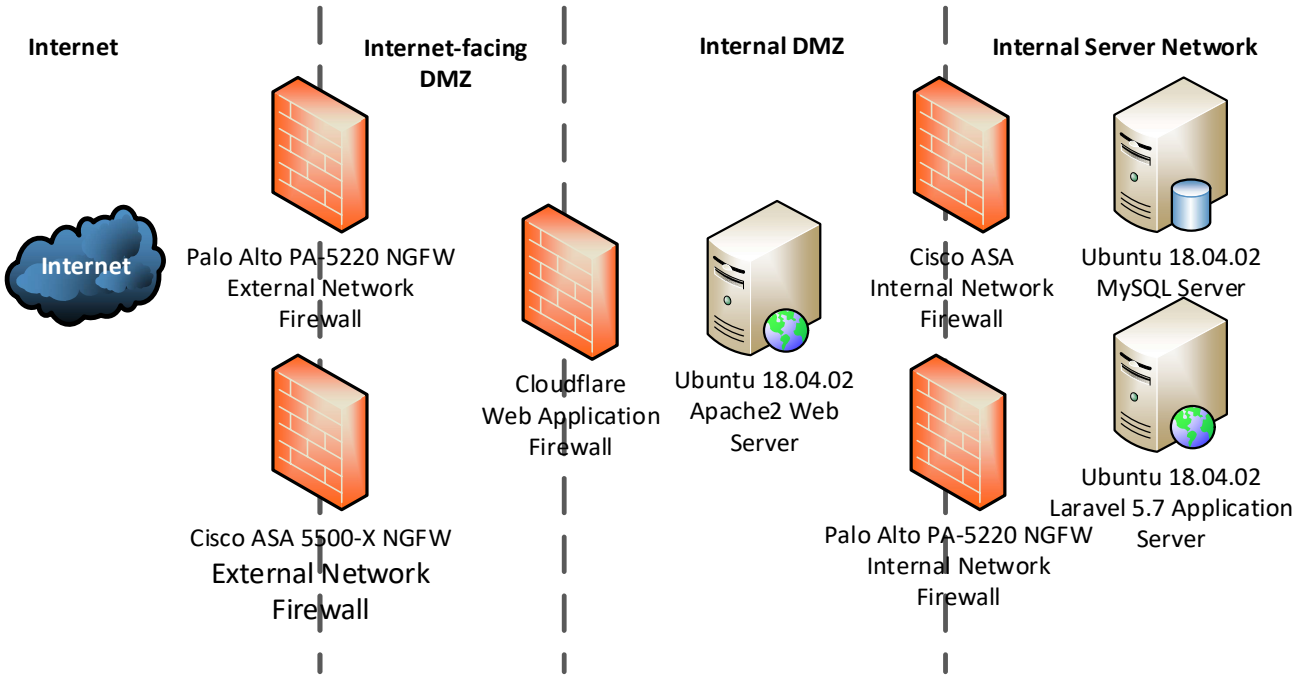


Diagram 1. Network and Server Infrastructure Architecture[41]

[41] Taaffe, Jonathon [2019] Diagram 1. Network and Server Infrastructure Architecture [Created 1st August 2019]

Authentication Process Flow

Following diagram outlines the authentication, authorisation and access process flow for all application users

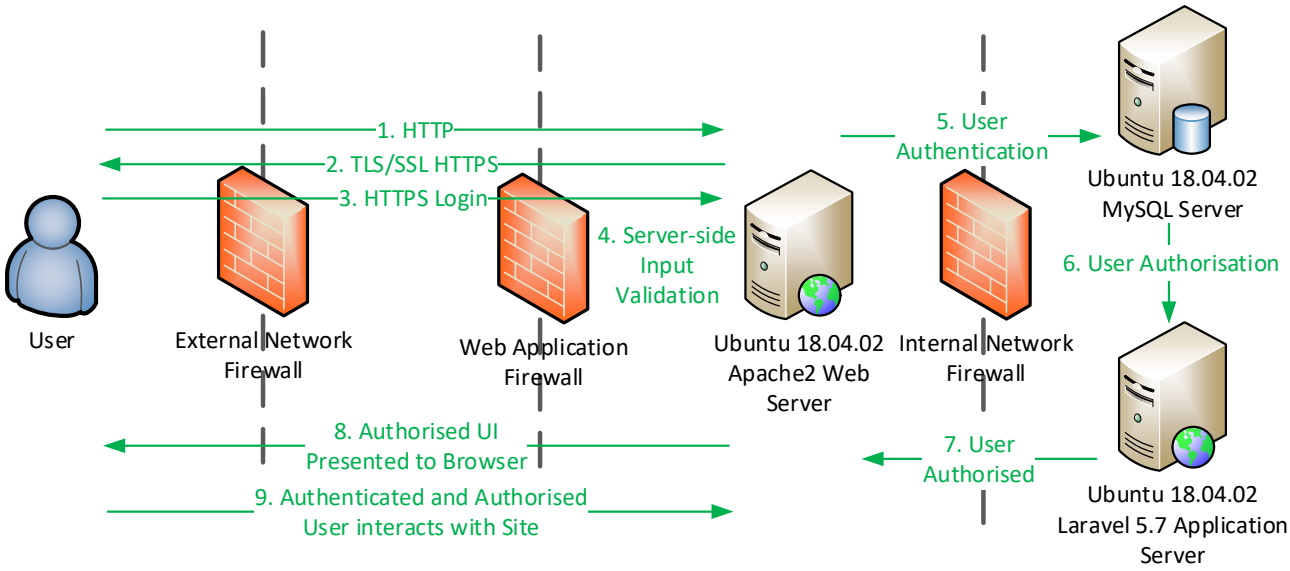


Diagram 2. Authentication Process Flow [42]

Secure Authentication Process

1. Client Browser: Establishes HTTP non-Secure Connection
2. Web Server: Establishes End-2-End TLS/SSL HTTPS Connection
3. Client Browser: User Enters Login Credentials
4. Web Server: Server-side Input Validation
5. SQL Server: Authenticates User
6. Application Server: Authorises User
7. Application Server: Authorisation Forwarded to Web Server
8. Web Server: Authorised UI Presented to Browser
9. Client Browser: User Securely Interacts with Site

[42] Taaffe, Jonathon [2019] Diagram 2. Authentication Process Flow [Created 1st August 2019]

Use Case Architectural Diagrams

The following Use Case Architecture Diagrams outline the Actions Permitted per Use Case.

Blogger Role Use Case

The following diagram shows the actions that are permitted by accounts assigned to the Blogger Role

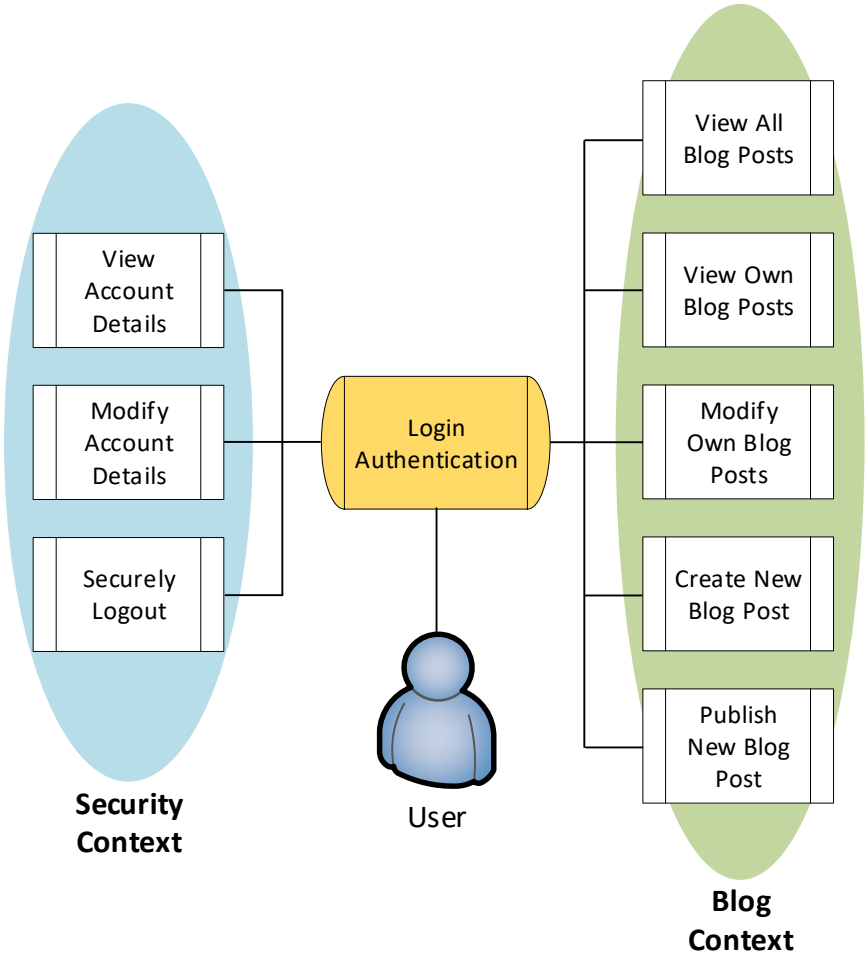


Diagram 3. Blogger Role Use Case[43]

[43] Taaffe, Jonathon [2019] Diagram 3. Blogger Role Use Case [Created 1st August 2019]

Administrator Role Use Case

The following diagram shows the actions that are permitted by accounts assigned to the Administrators Role

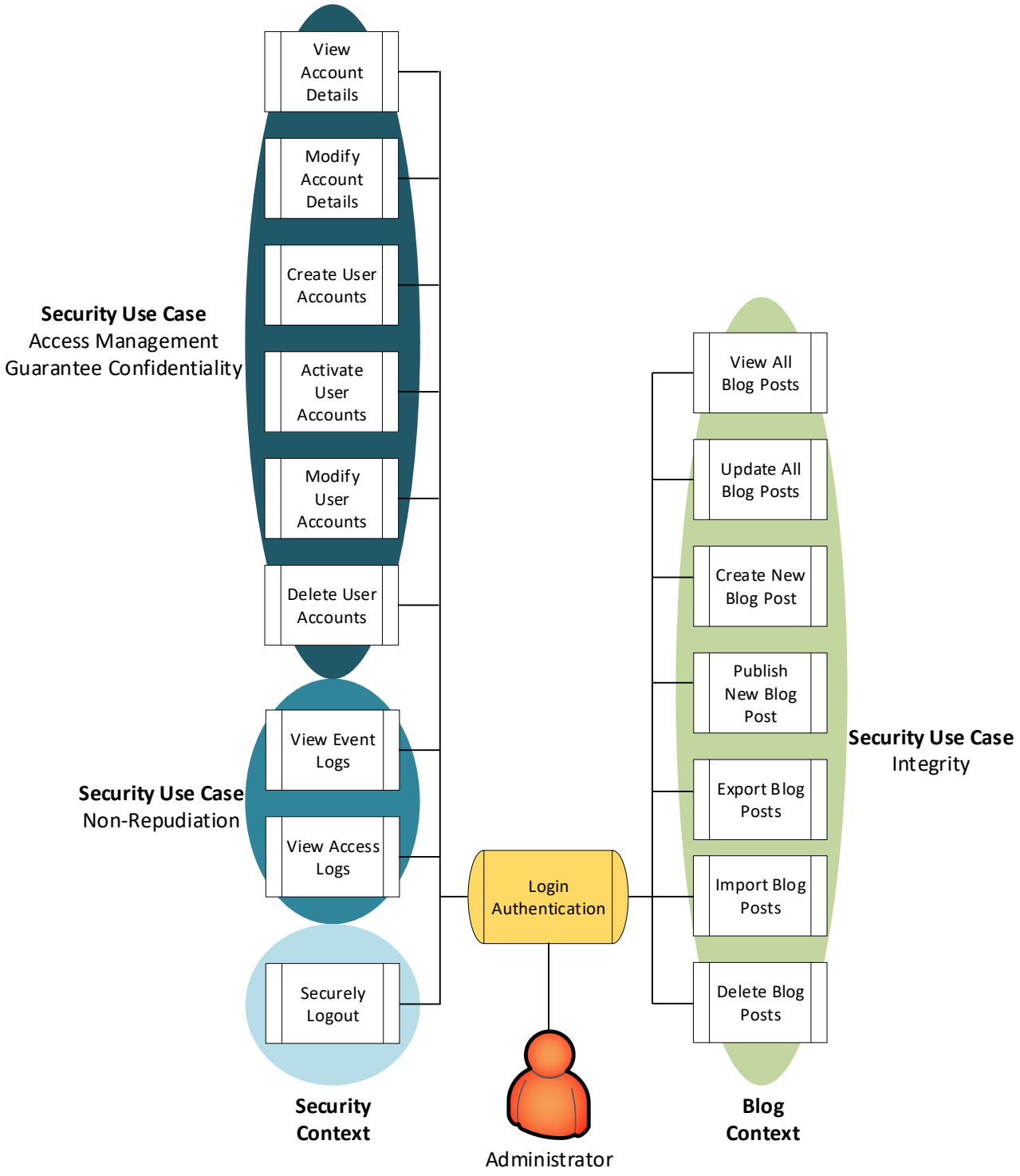


Diagram 4. Administrator Role Use Case[44]

[44] Taaffe, Jonathon [2019] Diagram 4. Administrator Role Use Case [Created 1st August 2019]

Super User Role Use Case

The following diagram shows the actions that are permitted by accounts assigned to the Super Users Role

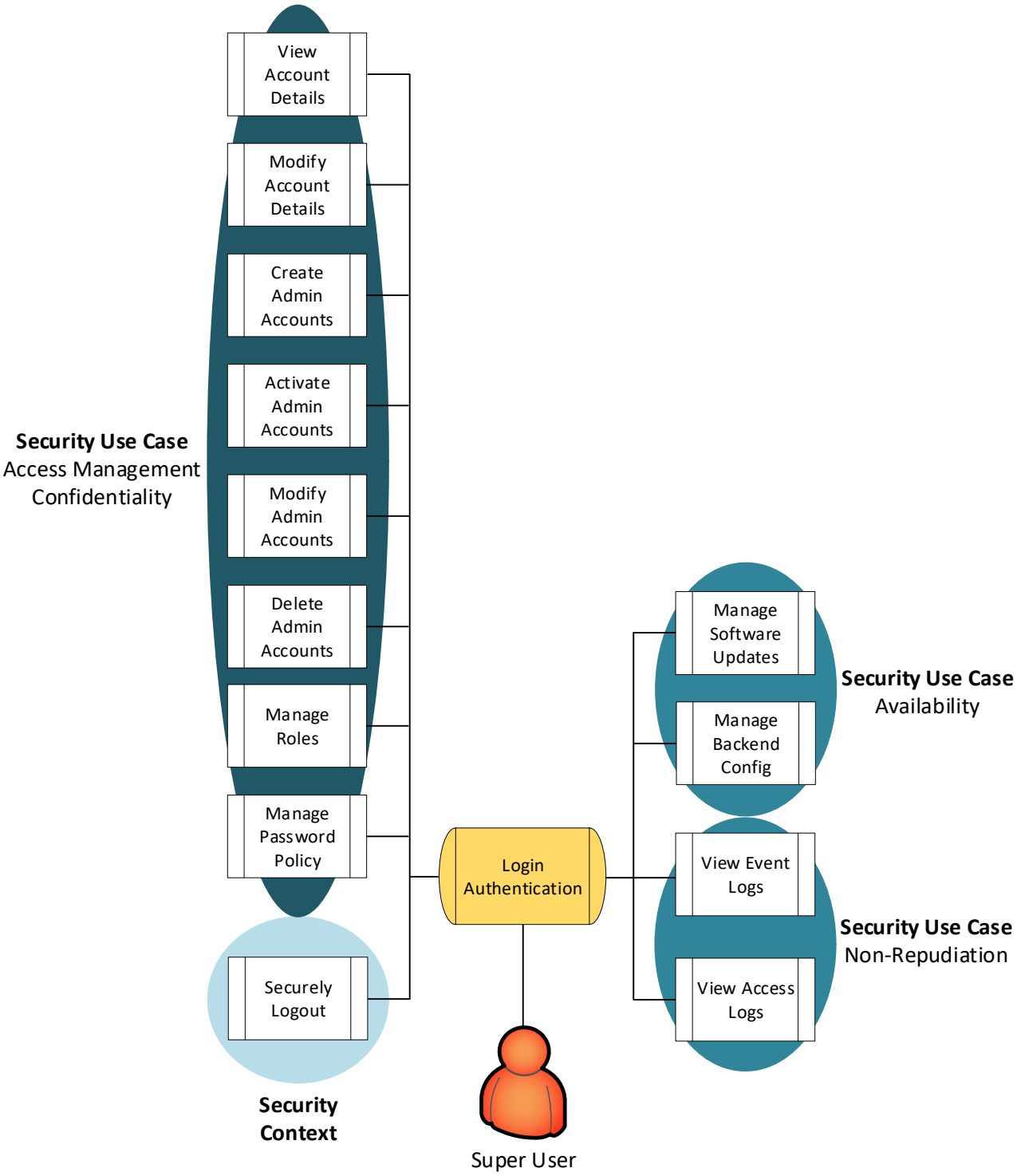


Diagram 5. Super Users Role Use Case[45]

[45] Taaffe, Jonathon [2019] Diagram 5. Super Users Role Use Case [Created 1st August 2018]

Implementation

As this secure enterprise blog application is built on the Laravel PHP Framework there are numerous framework files included for the application to function. The following section details the production security code configuration for the secure enterprise blog application

Connection Security

Function: Enforce HTTPS for front-end access
File: SEB\.htaccess

```
16      ## Uncomment following lines to force HTTPS.
17      RewriteCond %{HTTPS} off
18      RewriteRule (.*) https://%{SERVER_NAME}/$1 [R,L]
```

Function: Enforce HTTPS for back-end access
File: SEB\config\cms.php

```
42      |--------------------------------------------------------------------------
43      | Back-end force HTTPS security
44      |--------------------------------------------------------------------------
45      | Use this setting to force a secure protocol when accessing any back-end
46      | pages, including the authentication pages. If set to null, this setting
47      | is enabled when debug mode (app.debug) is disabled.
48      */
49      'backendForceSecure' => null,
```

Access Controls

Function: Application Environment Configuration
File: SEB\config\environment.php

```
|--------------------------------------------------------------------------
| Default Application Environment
|--------------------------------------------------------------------------
| This value determines the "environment" your application is currently
| running in. This may determine how you prefer to configure various
| services your application utilizes. Set this in your ".env" file.
*/
'default' => 'development',
```

Function: Enable Application Debug Mode
File: SEB\config\app.php

```
|--------------------------------------------------------------------------
| Application Debug Mode
|--------------------------------------------------------------------------
|
| When your application is in debug mode, detailed error messages with
| stack traces will be shown on every error that occurs within your
| application. If disabled, a simple generic error page is shown.
|
| You can create a CMS page with route "/error" to set the contents
| of this page. Otherwise a default error page is shown.
|
*/

'debug' => true,
```

Function: Base Directory Restriction Configuration
File: SEB\config\cms.php

```
|--------------------------------------------------------------------------
| Base Directory Restriction
|--------------------------------------------------------------------------
| Restricts loading backend template and config files to within the base
| directory of the application.
| WARNING: This should always be enabled for security reasons. However, in
| some cases you may need to disable this; for instance when developing
| plugins that are stored elsewhere in the filesystem for organizational
| reasons and then symlinked into the application plugins/ directory.
| NEVER have this disabled in production.
*/
'restrictBaseDir' => true,
```

Function: PHP Files Whitelist
File: SEB\.htaccess

```
46      ## Block all PHP files, except index
47      ##
48      RewriteCond %{REQUEST_FILENAME} -f
49      RewriteCond %{REQUEST_FILENAME} \.php$
50      RewriteRule !^index.php index.php [L,NC]
```

Function: Whitelist for folders and files
File: SEB\.htaccess

```
33      ## White listed folders
34      ##
35      RewriteCond %{REQUEST_FILENAME} -f
36      RewriteCond %{REQUEST_FILENAME} !/.well-known/*
37      RewriteCond %{REQUEST_FILENAME} !/storage/app/uploads/.*
38      RewriteCond %{REQUEST_FILENAME} !/storage/app/media/.*
39      RewriteCond %{REQUEST_FILENAME} !/storage/temp/public/.*
40      RewriteCond %{REQUEST_FILENAME} !/themes/.*/(assets|resources)/.*
41      RewriteCond %{REQUEST_FILENAME} !/plugins/.*/(assets|resources)/.*
42      RewriteCond %{REQUEST_FILENAME} !/modules/.*/(assets|resources)/.*
43      RewriteRule !^index.php index.php [L,NC]
```

Function: Blacklist for folders and files
File: SEB\.htaccess

```
21      ## Black listed folders
22      ##
23      RewriteRule ^bootstrap/.* index.php [L,NC]
24      RewriteRule ^config/.* index.php [L,NC]
25      RewriteRule ^vendor/.* index.php [L,NC]
26      RewriteRule ^storage/cms/.* index.php [L,NC]
27      RewriteRule ^storage/logs/.* index.php [L,NC]
28      RewriteRule ^storage/framework/.* index.php [L,NC]
29      RewriteRule ^storage/temp/protected/.* index.php [L,NC]
30      RewriteRule ^storage/app/uploads/protected/.* index.php [L,NC]
```

Authentication

Function: Authentication Controller – Authenticate User
File: SEB\modules\backend\controllers\Auth.php

```
88          $user = BackendAuth::authenticate([
89              'login' => post('login'),
90              'password' => post('password')
91          ], $remember);
```

Function: Authentication Manager
File: SEB\modules\backend\classes\AuthManager.php
Class: AuthManager

```
7    * Authentication manager.
8    *
9    */
10   class AuthManager extends RainAuthManager
11   {
12       protected static $instance;
13       protected $sessionKey = 'admin_auth';
14       protected $userModel = 'Backend\Models\User';
15       protected $groupModel = 'Backend\Models\UserGroup';
16       protected $throttleModel = 'Backend\Models\UserThrottle';
17       protected $requireActivation = false;
```

Function: Laravel Authentication Configuration – Authentication Manager
File: SEB\vendor\laravel\framework\src\Illuminate\Auth\AuthManager.php
Note: Multiple Authentication Options are configured in this file and too many to list all code here. The configurable settings available in this file include:

- Create a new Auth manager instance.
- Resolve the given guard.
- Create a session-based authentication guard.
- Create a token-based authentication guard.

Function: Authentication Controller – Configuration
File: SEB\modules\backend\controllers\Auth.php

```
18  class Auth extends Controller
19  {
20      protected $publicActions = ['index', 'signin', 'signout', 'restore', 'reset'
21
22      public function __construct()
23      {
24          parent::__construct();
25
26          $this->middleware(function ($request, $response) {
27              // Clear Cache and any previous data to fix Invalid security token i
28              $response->headers->set('Cache-Control', 'no-cache, no-store, must-r
29          })->only('signin');
30
31          // Only run on HTTPS connections
32          if (isset($_SERVER["HTTPS"]) && $_SERVER["HTTPS"] === "on") {
33              $this->middleware(function ($request, $response) {
34                  // Add HTTP Header 'Clear Site Data' to remove all Sensitive Dat
35                  $response->headers->set('Clear-Site-Data', 'cache, cookies, stor
36              })->only('signout');
```

Function: Authentication Controller – Display Login Page
File: SEB\modules\backend\controllers\Auth.php

```
55      public function signin()
56      {
57          $this->bodyClass = 'signin';
58
59          try {
60              if (post('postback')) {
61                  return $this->signin_onSubmit();
62              }
63
64              $this->bodyClass .= ' preload';
65          }
66          catch (Exception $ex) {
67              Flash::error($ex->getMessage());
68          }
69      }
```

Function: Authentication Controller – Login Function
File: SEB\modules\backend\controllers\Auth.php

```
71      public function signin_onSubmit()
72      {
73          $rules = [
74              'login'    => 'required|between:2,255',
75              'password' => 'required|between:4,255'
76          ];
77
78          $validation = Validator::make(post(), $rules);
79          if ($validation->fails()) {
80              throw new ValidationException($validation);
81          }
82
83          if (($remember = config('cms.backendForceRemember', true)) === null) {
84              $remember = (bool) post('remember');
85          }
```

Function: Authentication Controller – Log the Sign-In Event
File: SEB\modules\backend\controllers\Auth.php

```
102           AccessLog::add($user);
```

Function: Authentication Controller – Redirect to Login Page
File: SEB\modules\backend\controllers\Auth.php

```
47      public function index()
48      {
49          return Backend::redirect('backend/auth/signin');
50      }
```

Function: Login Page – Form Configuration
File: SEB\modules\backend\models\AccessLog.php

```
1   <h2><?= e(Backend\Models\BrandSetting::get('app_tagline')) ?></h2>
2
3   <?= Form::open() ?>
4       <input type="hidden" name="postback" value="1" />
5
6       <div class="form-elements" role="form">
7           <div class="form-group text-field horizontal-form october">
```

Function: Login Page – Username Input
File: SEB\modules\backend\controllers\auth\signin.htm

```
<input
    type="text"
    name="login"
    value="<?= e(post('login')) ?>"
    class="form-control icon user"
    placeholder="<?= e(trans('backend::lang.account.login_placeholder')) ?>"
    autocomplete="off"
    maxlength="255" />
```

Function: Login Page – Password Input
File: SEB\modules\backend\controllers\auth\signin.htm

```
<input
    type="password"
    name="password"
    value=""
    class="form-control icon lock"
    placeholder="<?= e(trans('backend::lang.account.password_placeholder')) ?>"
    autocomplete="off"
    maxlength="255" />
```

Function: Login Page – Login Button
File: SEB\modules\backend\controllers\auth\signin.htm

```
<button type="submit" class="btn btn-primary login-button">
    <?= e(trans('backend::lang.account.login')) ?>
</button>
```

Function: User Roles Controller – Configuration
File: SEB\modules\backend\controllers\UserRoles.php

```
class UserRoles extends Controller
{
    public $implement = [
        \Backend\Behaviors\FormController::class,
        \Backend\Behaviors\ListController::class
    ];
    public $formConfig = 'config_form.yaml';
    public $listConfig = 'config_list.yaml';
    public $requiredPermissions = ['backend.manage_users'];
```

Function: User Roles Controller – Access Restricted to Super Users
File: SEB\modules\backend\controllers\UserRoles.php

```
$this->bindEvent('page.beforeDisplay', function() {
    if (!$this->user->isSuperUser()) {
        return Response::make(View::make('backend::access_denied'), 403);
    }
```

Function: Laravel Authentication Configuration – User Identifier, Name and Password Request
Folder: SEB\vendor\laravel\framework\src\Illuminate\Auth
File: SEB\vendor\laravel\framework\src\Illuminate\Auth\Authenticatable.php

```
19      public function getAuthIdentifierName()
20      {
21          return $this->getKeyName();
22      }
23      public function getAuthIdentifier()
24      {
25          return $this->{$this->getAuthIdentifierName()};
26      }
27      public function getAuthPassword()
28      {
29          return $this->password;
30      }
```

Function: Laravel Authentication Configuration – Database Connection
File: SEB\vendor\laravel\framework\src\Illuminate\Auth\DatabaseUserProvider.php

```
11  class DatabaseUserProvider implements UserProvider
12  {
13      protected $conn;
14      protected $hasher;
15      protected $table;
```

Function: Laravel Authentication Configuration – Retrieve User Account Details
File: SEB\vendor\laravel\framework\src\Illuminate\Auth\DatabaseUserProvider.php

```
38      public function retrieveById($identifier)
39      {
40          $user = $this->conn->table($this->table)->find($identifier);
41
42          return $this->getGenericUser($user);
43      }
```

Function: Laravel Authentication Configuration – Retrieve User by Given Credentials
File: SEB\vendor\laravel\framework\src\Illuminate\Auth\DatabaseUserProvider.php

```
82      public function retrieveByCredentials(array $credentials)
83      {
84          if (empty($credentials) ||
85             (count($credentials) === 1 &&
86              array_key_exists('password', $credentials))) {
87              return;
```

Function: Laravel Authentication Configuration – Validate User against Given Credentials
File: SEB\vendor\laravel\framework\src\Illuminate\Auth\DatabaseUserProvider.php

```
129     public function validateCredentials(UserContract $user, array $credentials)
130     {
131         return $this->hasher->check(
132             $credentials['password'], $user->getAuthPassword()
133         );
134     }
```

Function: Authentication Controller – Log Out User
File: SEB\modules\backend\controllers\Auth.php

```
111     public function signout()
112     {
113         if (BackendAuth::isImpersonator()) {
114             BackendAuth::stopImpersonate();
115         } else {
116             BackendAuth::logout();
117         }
118
119         return Backend::redirect('backend');
```

Function: Access Logs Controller
File: SEB\modules\backend\controllers\AccessLogs.php

```
1   <?php namespace Backend\Controllers;
2
3   use Backend;
4   use BackendMenu;
5   use Backend\Classes\Controller;
6   use System\Classes\SettingsManager;
7
8   class AccessLogs extends Controller
9   {
10      public $implement = [
11          \Backend\Behaviors\ListController::class
12      ];
13
14      public $listConfig = 'config_list.yaml';
15
16      public $requiredPermissions = ['system.access_logs'];
17
18      public function __construct()
```

Database

Function: Database Connection Configuration
File: SEB\config\database.php

```
16      |--------------------------------------------------------------------------
17      | Default Database Connection Name
18      |--------------------------------------------------------------------------
19      | Here you may specify which of the database connections below you wish
20      | to use as your default connection for all database work. Of course
21      | you may use many connections at once using the Database library.
22      */
23      'default' => 'mysql',
```

```
26      |--------------------------------------------------------------------------
27      | Database Connections
28      |--------------------------------------------------------------------------
29      | Here are each of the database connections setup for your application.
30      | Of course, examples of configuring each database platform that is
31      | supported by Laravel is shown below to make development simple.
32      | All database work in Laravel is done through the PHP PDO facilities
33      | so make sure you have the driver for your particular database of
34      | choice installed on your machine before you begin development.
35      */
36      'connections' => [
44          'mysql' => [
45              'driver'    => 'mysql',
46              'host'      => 'localhost',
47              'port'      => 3306,
48              'database'  => 'SEB',
49              'username'  => 'seb_dba',
50              'password'  => 'WsfkVVAcr5DL5i91',
51              'charset'   => 'utf8mb4',
52              'collation' => 'utf8mb4_unicode_ci',
53              'prefix'    => '',
```

Function: Database Connection File
File: SEB\vendor\laravel\framework\src\Illuminate\Database\Connection.php

```
21  class Connection implements ConnectionInterface
22  {
23      use DetectsDeadlocks,
24          DetectsLostConnections,
25          Concerns\ManagesTransactions;
26
27      protected $pdo;
28      protected $readPdo;
29      protected $database;
30      protected $tablePrefix = '';
31      protected $config = [];
32      protected $reconnector;
33      protected $queryGrammar;
34      protected $schemaGrammar;
35      protected $postProcessor;
36      protected $events;
37      protected $fetchMode = PDO::FETCH_OBJ;
38      protected $transactions = 0;
39      protected $recordsModified = false;
40      protected $queryLog = [];
41      protected $loggingQueries = false;
42      protected $pretending = false;
43      protected $doctrineConnection;
44      protected static $resolvers = [];
```

Function: MySQL Database Connection
File: SEB\vendor\laravel\framework\src\Illuminate\Database\MySqlConnection.php

```php
class MySqlConnection extends Connection
{
    protected function getDefaultQueryGrammar()
    {
        return $this->withTablePrefix(new QueryGrammar);
    }

    public function getSchemaBuilder()
    {
        if (is_null($this->schemaGrammar)) {
            $this->useDefaultSchemaGrammar();
        }
        return new MySqlBuilder($this);
    }

    protected function getDefaultSchemaGrammar()
    {
        return $this->withTablePrefix(new SchemaGrammar);
    }

    protected function getDefaultPostProcessor()
    {
        return new MySqlProcessor;
    }
```

Function: Create Database Configuration – Default Administrator Account Configuration
File: SEB\modules\backend\database\seeds\SeedSetupAdmin.php

```php
class SeedSetupAdmin extends Seeder
{
    public static $email = 'admin@domain.tld';
    public static $login = 'admin';
    public static $password = 'admin';
    public static $firstName = 'Admin';
    public static $lastName = 'Person';
```

Function: Create Database Configuration – Admin Account Configuration
File: SEB\modules\backend\database\seeds\SeedSetupAdmin.php

```
        $user = User::create([
            'email'                 => static::$email,
            'login'                 => static::$login,
            'password'              => static::$password,
            'password_confirmation' => static::$password,
            'first_name'            => static::$firstName,
            'last_name'             => static::$lastName,
            'permissions'           => [],
            'is_superuser'          => true,
            'is_activated'          => true,
            'role_id'               => $role->id
```

Function: Create Database Configuration – Role Configuration
File: SEB\modules\backend\database\seeds\SeedSetupAdmin.php

```
    public function run()
    {
        UserRole::create([
            'name' => 'Publisher',
            'code' => UserRole::CODE_PUBLISHER,
            'description' => 'Site editor with access to publishing tools.',
        ]);

        $role = UserRole::create([
            'name' => 'Developer',
            'code' => UserRole::CODE_DEVELOPER,
            'description' => 'Site administrator with access to developer tools.',
        ]);

        $group = UserGroup::create([
            'name' => 'Owners',
            'code' => UserGroup::CODE_OWNERS,
            'description' => 'Default group for website owners.',
            'is_new_user_default' => false
        ]);
```

Function: PDO Fetch Style for stdClass Object Configuration
File: SEB\config\database.php

```
    |--------------------------------------------------------------------------
    | PDO Fetch Style
    |--------------------------------------------------------------------------
    | By default, database results will be returned as instances of the PHP
    | stdClass object; however, you may desire to retrieve records in an
    | array format for simplicity. Here you can tweak the fetch style.
    */
    'fetch' => PDO::FETCH_CLASS,
```

Function: Database Interaction
File: SEB\vendor\laravel\framework\src\Illuminate\Database\ConnectionInterface.php

```php
interface ConnectionInterface
{
    public function table($table);
    public function raw($value);
    public function selectOne($query, $bindings = []);
    public function select($query, $bindings = []);
    public function insert($query, $bindings = []);
    public function update($query, $bindings = []);
    public function delete($query, $bindings = []);
    public function statement($query, $bindings = []);
    public function affectingStatement($query, $bindings = []);
    public function unprepared($query);
    public function prepareBindings(array $bindings);
```

Encryption

Function: Encryption Key and Cipher selection
File: SEB\config\app.php

```
94      |--------------------------------------------------------------------------
95      | Encryption Key
96      |--------------------------------------------------------------------------
97      |
98      | This key is used by the Illuminate encrypter service and should be set
99      | to a random, 32 character string, otherwise these encrypted strings
100     | will not be safe. Please do this before deploying an application!
101     |
102     */
103
104     'key' => 'XMxrLfEE8PM7gxKkXkVi80a0jG1Z0Twm',
105
106     'cipher' => 'AES-256-CBC',
```

Function: Database Data Encryption
File: SEB\vendor\laravel\framework\src\Illuminate\Encryption\Encrypter.php

```
10  class Encrypter implements EncrypterContract
11  {
12      protected $key;
13      protected $cipher;
14
15      public function __construct($key, $cipher = 'AES-128-CBC')
16      {
17          $key = (string) $key;
18
19          if (static::supported($key, $cipher)) {
20              $this->key = $key;
21              $this->cipher = $cipher;
22          } else {
23              throw new RuntimeException('The only supported ciphers are AES-128-C
24          }
```

Function: Database Data Encryption – Create a new encryption key with given cipher
File: SEB\vendor\laravel\framework\src\Illuminate\Encryption\Encrypter.php

```
48      public static function generateKey($cipher)
49      {
50          return random_bytes($cipher == 'AES-128-CBC' ? 16 : 32);
51      }
```

Function: Database Data Encryption – Encrypt a new encryption key with OpenSSL and encode with Base64
File: SEB\vendor\laravel\framework\src\Illuminate\Encryption\Encrypter.php

```
62      public function encrypt($value, $serialize = true)
63      {
64          $iv = random_bytes(openssl_cipher_iv_length($this->cipher));
65          $value = \openssl_encrypt(
66              $serialize ? serialize($value) : $value,
67              $this->cipher, $this->key, 0, $iv
68          );
69
70          if ($value === false) {
71              throw new EncryptException('Could not encrypt the data.');
72          }
73
74          $mac = $this->hash($iv = base64_encode($iv), $value);
75
76          $json = json_encode(compact('iv', 'value', 'mac'));
77
78          if (json_last_error() !== JSON_ERROR_NONE) {
79              throw new EncryptException('Could not encrypt the data.');
80          }
81
82          return base64_encode($json);
```

Function: Database Data Encryption - Determine if given key and cipher combination is valid
File: SEB\vendor\laravel\framework\src\Illuminate\Encryption\Encrypter.php

```
34      public static function supported($key, $cipher)
35      {
36          $length = mb_strlen($key, '8bit');
37
38          return ($cipher === 'AES-128-CBC' && $length === 16) ||
39                 ($cipher === 'AES-256-CBC' && $length === 32);
```

Hashing

Function: Database Hashing Function – Check given plain value against a hash
File: SEB\vendor\laravel\framework\src\Illuminate\Hashing\BcryptHasher.php

```
47      public function check($value, $hashedValue, array $options = [])
48      {
49          if (strlen($hashedValue) === 0) {
50              return false;
51          }
52
53          return password_verify($value, $hashedValue);
54      }
```

Function: Database Hashing Function – Check if value has been hashed using hashing options
File: SEB\vendor\laravel\framework\src\Illuminate\Hashing\BcryptHasher.php

```
63      public function needsRehash($hashedValue, array $options = [])
64      {
65          return password_needs_rehash($hashedValue, PASSWORD_BCRYPT, [
66              'cost' => $this->cost($options),
67          ]);
```

Function: Database Hashing Function – Hash a given value
File: SEB\vendor\laravel\framework\src\Illuminate\Hashing\BcryptHasher.php

```
8   class BcryptHasher implements HasherContract
9   {
10      protected $rounds = 10;
11      public function make($value, array $options = [])
12      {
13          $hash = password_hash($value, PASSWORD_BCRYPT, [
14              'cost' => $this->cost($options),
15          ]);
16
17          if ($hash === false) {
18              throw new RuntimeException('Bcrypt hashing not supported.');
19          }
20
21          return $hash;
```

Session Management

Function: Session Configuration
Folder: SEB\vendor\laravel\framework\src\Illuminate\Session
Note: All session configuration contained in this location. Too many options and lines of code to include in this document. Sore Session management options as follows

Function: Session Encryption
File: SEB\config\session.php

```
34      | Session Encryption
35      |--------------------------------------------------------------------------
36      | This option allows you to easily specify that all of your session data
37      | should be encrypted before it is stored. All encryption will be run
38      | automatically by Laravel and you can use the Session like normal.
39      */
40      'encrypt' => true,
```

Function: Session Authentication – AuthenticateSession Class
File: SEB\vendor\laravel\framework\src\Illuminate\Session\Middleware\AuthenticateSession.php

```
 9  class AuthenticateSession
10  {
11      protected $auth;
12      public function __construct(AuthFactory $auth)
13      {
14          $this->auth = $auth;
15      }
```

Function: Session Authentication – Store Users password Hash in Session
File: SEB\vendor\laravel\framework\src\Illuminate\Session\Middleware\AuthenticateSession.php

```
57      protected function storePasswordHashInSession($request)
58      {
59          if (! $request->user()) {
60              return;
61          }
62
63          $request->session()->put([
64              'password_hash' => $request->user()->getAuthPassword(),
```

Function: Session Authentication – Securely Log User out of Session
File: SEB\vendor\laravel\framework\src\Illuminate\Session\Middleware\AuthenticateSession.php

```
76      protected function logout($request)
77      {
78          $this->auth->logout();
79
80          $request->session()->flush();
81
82          throw new AuthenticationException;
```

Function: Session Database Connection Configuration
File: SEB\config\session.php

```
56      |--------------------------------------------------------------------------
57      | Session Database Connection
58      |--------------------------------------------------------------------------
59      | When using the "database" or "redis" session drivers, you may specify a
60      | connection that should be used to manage these sessions. This should
61      | correspond to a connection in your database configuration options.
62      */
63      'connection' => mysql,
```

Function: Session Database Table Configuration
File: SEB\config\session.php

```
66      |--------------------------------------------------------------------------
67      | Session Database Table
68      |--------------------------------------------------------------------------
69      | When using the "database" session driver, you may specify the table we
70      | should use to manage the sessions. Of course, a sensible default is
71      | provided for you; however, you are free to change this as needed.
72      */
73      'table' => 'sessions',
```

Function: Session Cookie Name
File: SEB\config\session.php

```
89      |--------------------------------------------------------------------------
90      | Session Cookie Name
91      |--------------------------------------------------------------------------
92      | Here you may change the name of the cookie used to identify a session
93      | instance by ID. The name specified here will get used every time a
94      | new session cookie is created by the framework for every driver.
95      */
96      'cookie' => 'seb_session',
```

Function: Session Cookie Storage Location
File: SEB\storage\framework\sessions\...

Function: Session Cookie Path
File: SEB\config\session.php

```
99      |--------------------------------------------------------------------------
100     | Session Cookie Path
101     |--------------------------------------------------------------------------
102     | The session cookie path determines the path for which the cookie will
103     | be regarded as available. Typically, this will be the root path of
104     | your application but you are free to change this when necessary.
105     */
106     'path' => '/',
```

Function: Session Cookie Domain
File: SEB\config\session.php

```
109       |--------------------------------------------------------------------------
110       | Session Cookie Domain
111       |--------------------------------------------------------------------------
112       | Here you may change the domain of the cookie used to identify a session
113       | in your application. This will determine which domains the cookie is
114       | available to in your application. A sensible default has been set.
115       */
116       'domain' => www.seb.com,
```

Function: Session Cookie HTTPS Only
File: SEB\config\session.php

```
119       |--------------------------------------------------------------------------
120       | HTTPS Only Cookies
121       |--------------------------------------------------------------------------
122       | By setting this option to true, session cookies will only be sent back
123       | to the server if the browser has a HTTPS connection. This will keep
124       | the cookie from being sent to you if it can not be done securely.
125       */
126       'secure' => true,
```

Function: Session Cookie SameSite
File: SEB\config\session.php

```
130       | Same-Site Cookies
131       |--------------------------------------------------------------------------
132       | This option determines how your cookies behave when cross-site requests
133       | take place, and can be used to mitigate CSRF attacks. By default, we
134       | do not enable this as other CSRF protection services are in place.
135       | In the strict mode, the cookie is not sent with any cross-site usage
136       | even if the user follows a link to another website. Lax cookies are
137       | only sent with a top-level get request.
138       | Supported: "lax", "strict"
139       */
140       'same_site' => strict,
```

Function: Back-end Session Duration Configuration
File: SEB\config\cms.php

```
53        | Back-end login remember
54        |--------------------------------------------------------------------------
55        | Define live duration of backend sessions :
56        | true  - session never expire (cookie expiration in 5 years)
57        | false - session have a limited time (see session.lifetime)
58        | null  - The form login display a checkbox that allow user to choose
59        |         wanted behavior
60        */
61        'backendForceRemember' => true,
```

Function: Session Lifetime Configuration
File: SEB\config\session.php

```
22      |--------------------------------------------------------------------------
23      | Session Lifetime
24      |--------------------------------------------------------------------------
25      | Here you may specify the number of minutes that you wish the session
26      | to be allowed to remain idle for it is expired. If you want them
27      | to immediately expire when the browser closes, set it to zero.
28      */
29      'lifetime' => 120,
30      'expire_on_close' => true,
```

Cookie Management

Function: Cookie Configuration File
File: SEB\vendor\laravel\framework\src\Illuminate\Cookie\CookieJar.php

```
class CookieJar implements JarContract
{
    use InteractsWithTime;
    protected $path = '/';
    protected $domain;
    protected $secure = false;
    protected $sameSite;
    protected $queued = [];
```

Function: Unencrypted cookie Whitelist
File: SEB\config\cookie.php

```
    |--------------------------------------------------------------------------
    | Cookies that should not be encrypted
    |--------------------------------------------------------------------------
    | Laravel encrypts/decrypts cookies by default. You can specify cookies
    | that should not be encrypted or decrypted here. This is useful, for
    | example, when you want to pass data from frontend to server side backend
    | via cookies, and vice versa.
    */
    'unencryptedCookies' => [
        // 'my_cookie',
```

Function: jQuery Plugin for Reading, Writing and Deleting Cookies
File: SEB\modules\backend\assets\js\vendor\ jquery.cookie.js
URL: https://github.com/carhartl/jquery-cookie
Code: Encoding, Decoding, Stringify and Parse a Cookie

```
23    function encode(s) {
24      return config.raw ? s : encodeURIComponent(s);
25    }
26
27    function decode(s) {
28      return config.raw ? s : decodeURIComponent(s);
29    }
30
31    function stringifyCookieValue(value) {
32      return encode(config.json ? JSON.stringify(value) : String(value));
33    }
34
35    function parseCookieValue(s) {
36      if (s.indexOf('"') === 0) {
37        // This is a quoted cookie as according to RFC2068, unescape...
38        s = s.slice(1, -1).replace(/\\"/g, '"').replace(/\\\\/g, '\\');
39      }
```

Code: Creating a Cookie

```
55    var config = $.cookie = function (key, value, options) {
```

Code: Cookie Configuration

```
59      if (arguments.length > 1 && !$.isFunction(value)) {
60        options = $.extend({}, config.defaults, options);
61
62        if (typeof options.expires === 'number') {
63          var days = options.expires, t = options.expires = new Date();
64          t.setTime(+t + days * 864e+5);
65        }
66
67        return (document.cookie = [
68          encode(key), '=', stringifyCookieValue(value),
69          options.expires ? '; expires=' + options.expires.toUTCString() : '', //
70          options.path    ? '; path=' + options.path : '',
71          options.domain  ? '; domain=' + options.domain : '',
72          options.secure  ? '; secure' : ''
73        ].join(''));
74      }
```

Code: Deleting a Cookie

```
107     $.removeCookie = function (key, options) {
108       if ($.cookie(key) === undefined) {
109         return false;
110       }
111
112       // Must not alter options, thus extending a fresh object...
113       $.cookie(key, '', $.extend({}, options, { expires: -1 }));
114       return !$.cookie(key);
115     };
```

Cross Site Request Forgery (CSRF) Prevention

Function: Enable Cross Site Request Forgery
File: SEB\config\cms.php

```
323         |--------------------------------------------------------------------------
324         | Cross Site Request Forgery (CSRF) Protection
325         |--------------------------------------------------------------------------
326         | If the CSRF protection is enabled, all "postback" requests are checked
327         | for a valid security token.
328         */
329         'enableCsrfProtection' => true,
```

Function: Add CSRF Token to all AJAX Requests
File: SEB\modules\backend\assets\js\backend.js

```
6    * Ensure the CSRF token is added to all AJAX requests.
7    */
8   $.ajaxPrefilter(function(options) {
9       var token = $('meta[name="csrf-token"]').attr('content')
10
11      if (token) {
12          if (!options.headers) options.headers = {}
13          options.headers['X-CSRF-TOKEN'] = token
14      }
15  })
```

Logging

Function: System Logs Location
File: SEB\storage\logs\system.log

Function: Logging level Configuration
File: SEB\config\app.php

```
109       |--------------------------------------------------------------------------
110       | Logging Configuration
111       |--------------------------------------------------------------------------
112       | Here you may configure the log settings for your application. Out of
113       | the box, Laravel uses the Monolog PHP logging library. This gives
114       | you a variety of powerful log handlers / formatters to utilize.
115       | Available Settings: "single", "daily", "syslog", "errorlog"
116       */
117
118       'log' => 'single',
```

Function: Access Logs – Configuration
File: SEB\modules\backend\controllers\accesslogs\config_filter.yaml

```
5   scopes:
6
7       created_at:
8           label: backend::lang.access_log.created_at
9           type: daterange
10          conditions: created_at >= ':after' AND created_at <= ':before'
11
12      user:
13          label: backend::lang.access_log.login
14          modelClass: Backend\Models\User
15          conditions: user_id in (:filtered)
16          nameFrom: login
```

Function: Access Logs – Database Configuration
File: SEB\modules\backend\models\AccessLog.php

```php
class AccessLog extends Model
{
    protected $table = 'backend_access_log';
    public $belongsTo = [
        'user' => User::class
    ];
    public static function add($user)
    {
        $record = new static;
        $record->user = $user;
        $record->ip_address = Request::getClientIp();
        $record->save();

        return $record;
    }
```

Function: Access Logs – Display Entries
File: SEB\modules\backend\models\AccessLog.php

```php
    public static function getRecent($user)
    {
        $records = static::where('user_id', $user->id)
            ->orderBy('created_at', 'desc')
            ->limit(2)
            ->get()
        ;

        if (!count($records)) {
            return null;
        }

        $first = $records->first();

        return !$first->created_at->isToday() ? $first : $records->pop();
```

Function: Event Logs – Controller
File: SEB\modules\system\controllers\EventLogs.php

```php
class EventLogs extends Controller
{
    public $implement = [
        \Backend\Behaviors\FormController::class,
        \Backend\Behaviors\ListController::class
    ];

    public $formConfig = 'config_form.yaml';
    public $listConfig = 'config_list.yaml';
    public $requiredPermissions = ['system.access_logs'];
```

Function: Event Logs – Database Configuration
File: SEB\modules\system\models\EventLog.php

```php
class EventLog extends Model
{
    protected $table = 'system_event_logs';
    protected $jsonable = ['details'];
    public static function useLogging()
    {
        return (
            class_exists('Model') &&
            Model::getConnectionResolver() &&
            App::hasDatabase() &&
            !defined('OCTOBER_NO_EVENT_LOGGING') &&
            LogSetting::get('log_events')
        );
    }
```

Function: Event Logs – Create an Event Log Entry
File: SEB\modules\system\models\EventLog.php

```
36      public static function add($message, $level = 'info', $details = null)
37      {
38          $record = new static;
39          $record->message = $message;
40          $record->level = $level;
41
42          if ($details !== null) {
43              $record->details = (array) $details;
44          }
45
46          try {
47              $record->save();
48          }
49          catch (Exception $ex) {}
50
51          return $record;
```

Function: Event Logs – Request Logs
File: SEB\modules\system\controllers\EventLogs.php

```
16  class RequestLogs extends Controller
17  {
18      public $implement = [
19          \Backend\Behaviors\FormController::class,
20          \Backend\Behaviors\ListController::class
21      ];
22      public $formConfig = 'config_form.yaml';
23      public $listConfig = 'config_list.yaml';
24      public $requiredPermissions = ['system.access_logs'];
```

Availability

Function: System and Software Updates Configuration
File: SEB\modules\system\controllers\Updates.php

```
29   */
30  class Updates extends Controller
31  {
32          public $implement = [
33          \Backend\Behaviors\ListController::class
34      ];
35
36      public $listConfig = [
37          'list' => 'config_list.yaml',
38          'manage' => 'config_manage_list.yaml'
39      ];
40
41      public $requiredPermissions = ['system.manage_updates'];
```

Function: Application Update Configuration
File: SEB\config\cms.php

```
 99      |--------------------------------------------------------------------------
100      | Prevents application updates
101      |--------------------------------------------------------------------------
102      | If using composer or git to download updates to the core files, set this
103      | value to 'true' to prevent the update gateway from trying to download
104      | these files again as part of the application update process. Plugins
105      | and themes will still be downloaded.
106      */
107      'disableCoreUpdates' => false,
```

Password Policy

The password policy applied to the secure enterprise blog application complies with the corporate password policy. Below are details of the configured password policy:

For Registration and Access users will require a unique Username with a Complex Password. User account Usernames will be configured as per their existing Active Directory unique username.

Password Policy

Number of Characters	12
Number of Numbers	2
Number of Lower-Case Characters	2
Number of Upper-Case Characters	2
Number of Special Characters	2
Password Reuse	Disabled
All Account Credentials Error Prompt	“Invalid Username or Password.”
User Account Temporary Suspension	After 3 Failed Logins

Table 6. Password Policy[46]

[46] Taaffe, Jonathon [2019] Table 6. Password Policy [Created 1st August 2019]

Database Design

There is a total of 34 Tables configured as InnoDB and collated using utf8_general_ci. 19 of the 34 tables are operational in the development environment. The remaining 15 tables are configured for future feature releases which will be rolled out when the application has gone live.

Database Tables

The following table details the database tables in operation in the development environment:

Table Name	Category	Details
backend_user_throttle	Access Control	Throttle/Limit User Access
backend_user_roles	Authentication	Role Based Access Control Roles
backend_users	Authentication	User Accounts
backend_user_preferences	Authentication	User Preferences
Migrations	Database	Database Processing
Sessions	Session Management	Session Management
system_event_logs	Logging	System Event Logs
system_request_logs	Logging	System Request Logs
backend_access_log	Logging	Application Access Logs
system_plugin_history	Availability	System and Application Updates
system_plugin_versions	Availability	System and Application Update Version Control
system_revisions	Availability	System Revision Tracking
Cache	Availability	Performance Configuration
system_files	Application	Secure Enterprise Blog Logo
system_parameters	Application	Secure Enterprise Blog Configuration
cms_theme_data	Application	Secure Enterprise Blog User Interface
cms_theme_logs	Application	Secure Enterprise Blog User Interface
rainlab_blog_posts	Application	Secure Enterprise Blog Content
system_settings	System	System Settings

Table 7. Database Tables[47]

Database Table Configuration

The following details the configuration for each of the tables in operation in the development environment:

backend_user_throttle

```
CREATE TABLE `backend_user_throttle` (
  `id` int(10) UNSIGNED NOT NULL,
  `user_id` int(10) UNSIGNED DEFAULT NULL,
  `ip_address` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `attempts` int(11) NOT NULL DEFAULT 0,
  `last_attempt_at` timestamp NULL DEFAULT NULL,
  `is_suspended` tinyint(1) NOT NULL DEFAULT 0,
  `suspended_at` timestamp NULL DEFAULT NULL,
  `is_banned` tinyint(1) NOT NULL DEFAULT 0,
  `banned_at` timestamp NULL DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

[47] Taaffe, Jonathon [2019] Table 7. Database Tables [Created 1st August 2019]

backend_user_roles

```
CREATE TABLE `backend_user_roles` (
  `id` int(10) UNSIGNED NOT NULL,
  `name` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
  `code` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `description` text COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `permissions` text COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `is_system` tinyint(1) NOT NULL DEFAULT 0,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

backend_users

```
CREATE TABLE `backend_users` (
  `id` int(10) UNSIGNED NOT NULL,
  `first_name` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `last_name` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `login` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
  `email` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
  `password` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
  `activation_code` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `persist_code` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `reset_password_code` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT
NULL,
  `permissions` text COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `is_activated` tinyint(1) NOT NULL DEFAULT 0,
  `role_id` int(10) UNSIGNED DEFAULT NULL,
  `activated_at` timestamp NULL DEFAULT NULL,
  `last_login` timestamp NULL DEFAULT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  `deleted_at` timestamp NULL DEFAULT NULL,
  `is_superuser` tinyint(1) NOT NULL DEFAULT 0
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

backend_user_preferences

```
CREATE TABLE `backend_user_preferences` (
  `id` int(10) UNSIGNED NOT NULL,
  `user_id` int(10) UNSIGNED NOT NULL,
  `namespace` varchar(100) COLLATE utf8mb4_unicode_ci NOT NULL,
  `group` varchar(50) COLLATE utf8mb4_unicode_ci NOT NULL,
  `item` varchar(150) COLLATE utf8mb4_unicode_ci NOT NULL,
  `value` text COLLATE utf8mb4_unicode_ci DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

migrations

```
CREATE TABLE `migrations` (
  `id` int(10) UNSIGNED NOT NULL,
  `migration` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
  `batch` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

sessions

```
CREATE TABLE `sessions` (
  `id` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
  `payload` text COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `last_activity` int(11) DEFAULT NULL,
  `user_id` int(10) UNSIGNED DEFAULT NULL,
  `ip_address` varchar(45) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `user_agent` text COLLATE utf8mb4_unicode_ci DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

system_event_logs

```
CREATE TABLE `system_event_logs` (
  `id` int(10) UNSIGNED NOT NULL,
  `level` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `message` text COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `details` mediumtext COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

system_request_logs

```
CREATE TABLE `system_request_logs` (
  `id` int(10) UNSIGNED NOT NULL,
  `status_code` int(11) DEFAULT NULL,
  `url` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `referer` text COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `count` int(11) NOT NULL DEFAULT 0,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

backend_access_log

```
CREATE TABLE `backend_access_log` (
  `id` int(10) UNSIGNED NOT NULL,
  `user_id` int(10) UNSIGNED NOT NULL,
  `ip_address` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

system_plugin_history

```
CREATE TABLE `system_plugin_history` (
  `id` int(10) UNSIGNED NOT NULL,
  `code` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
  `type` varchar(20) COLLATE utf8mb4_unicode_ci NOT NULL,
  `version` varchar(50) COLLATE utf8mb4_unicode_ci NOT NULL,
  `detail` text COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `created_at` timestamp NULL DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

system_plugin_versions

```
CREATE TABLE `system_plugin_versions` (
  `id` int(10) UNSIGNED NOT NULL,
  `code` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
  `version` varchar(50) COLLATE utf8mb4_unicode_ci NOT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `is_disabled` tinyint(1) NOT NULL DEFAULT 0,
  `is_frozen` tinyint(1) NOT NULL DEFAULT 0
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

system_revisions

```
CREATE TABLE `system_revisions` (
  `id` int(10) UNSIGNED NOT NULL,
  `user_id` int(10) UNSIGNED DEFAULT NULL,
  `field` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `cast` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `old_value` text COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `new_value` text COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `revisionable_type` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
  `revisionable_id` int(11) NOT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

cache

```
CREATE TABLE `cache` (
  `key` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
  `value` longtext COLLATE utf8mb4_unicode_ci NOT NULL,
  `expiration` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

system_files

```
CREATE TABLE `system_files` (
  `id` int(10) UNSIGNED NOT NULL,
  `disk_name` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
  `file_name` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
  `file_size` int(11) NOT NULL,
  `content_type` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
  `title` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `description` text COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `field` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `attachment_id` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `attachment_type` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `is_public` tinyint(1) NOT NULL DEFAULT 1,
  `sort_order` int(11) DEFAULT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

system_parameters

```
CREATE TABLE `system_parameters` (
  `id` int(10) UNSIGNED NOT NULL,
  `namespace` varchar(100) COLLATE utf8mb4_unicode_ci NOT NULL,
  `group` varchar(50) COLLATE utf8mb4_unicode_ci NOT NULL,
  `item` varchar(150) COLLATE utf8mb4_unicode_ci NOT NULL,
  `value` text COLLATE utf8mb4_unicode_ci DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

cms_theme_data

```
CREATE TABLE `cms_theme_data` (
  `id` int(10) UNSIGNED NOT NULL,
  `theme` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `data` mediumtext COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

cms_theme_logs

```
CREATE TABLE `cms_theme_logs` (
  `id` int(10) UNSIGNED NOT NULL,
  `type` varchar(20) COLLATE utf8mb4_unicode_ci NOT NULL,
  `theme` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `template` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `old_template` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `content` longtext COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `old_content` longtext COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `user_id` int(11) DEFAULT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

rainlab_blog_posts

```
CREATE TABLE `rainlab_blog_posts` (
  `id` int(10) UNSIGNED NOT NULL,
  `user_id` int(10) UNSIGNED DEFAULT NULL,
  `title` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `slug` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
  `excerpt` text COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `content` longtext COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `content_html` longtext COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `published_at` timestamp NULL DEFAULT NULL,
  `published` tinyint(1) NOT NULL DEFAULT 0,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  `metadata` mediumtext COLLATE utf8mb4_unicode_ci DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

system_settings

```
CREATE TABLE `system_settings` (
  `id` int(10) UNSIGNED NOT NULL,
  `item` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `value` mediumtext COLLATE utf8mb4_unicode_ci DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

Security Implementation

MySQL Password Policy

The default MySQL credentials have been disabled and a new Database Administrator Account has been configured as follows

Database Name	Seb
Database Type	MySQL
Hostname	Localhost
MySQL Port	3306
Encoding	UFT8_General_CI
Authentication	Native MySQL
Password Policy	16 Characters including: 2 Numbers, 2 Lower Case, 2 Upper Case
User account	'seb_dba'@'localhost'
MySQL Login	seb_dba
Encryption Key	32 Character Random Key

Table 8. MySQL Database Administrator Account Configuration[48]

HTTPS Configuration

The Front-end Apache2 Web Server has been configured with a Symantec Verisign SSL SHA256RSA (4096-Bit) Certificate[49] to allow secure HTTPS connections. The ..\tomcat\conf\server.xml file has been configured to enable SSL communication as follows:

```
87      <Connector port="8443" protocol="org.apache.coyote.http11.Http11Protocol"
88                 maxThreads="150" SSLEnabled="true" scheme="https" secure="true"
89                 clientAuth="false" sslProtocol="TLS" />
```

HTTP Strict Transport Security (HSTS)[50]

HSTS has been configured by the HTTPS response header field "Strict-Transport-Security". HSTS prevents protocol downgrade attacks and cookie hijacking. With HSTS configured the front-end web server will request all browsers connect using only secure HTTPS connection.

Apache HSTS Configuration[51]

The following line of code has been added to the Apache HTTPS VirtualHost file:

```
Header always set Strict-Transport-Security "max-age=31536000; includeSubDomains"
```

[48] Taaffe, Jonathon [2019] Table 8. MySQL Database Administrator Account Configuration [Created 1st August 2019]
[49] Symantec Verisign [2019] SSL/TLS Certificates https://www.websecurity.symantec.com/ssl-certificate [Accessed 1st August 2019]
[50] Wikipedia.com [2019] HTTP Strict Transport Security https://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security [Accessed 1st August 2019]
[51] LinuxTogether.org [2019] How to Setup HTTP Strict Transport Security (HSTS) for Apache https://linuxtogether.org/how-to-setup-hsts-for-apache/ [Accessed 1st August 2019]

HSTS Perpetual Requirements[52]

The following additional configurations have been configured to enable HSTS:

- All HTTP traffic redirected to HTTPS with 301 Permanent Redirect
- All subdomains have been included in the Symantec Verisign SSL Certificate
- Serve an HSTS header on the base domain for HTTPS requests.
- Max-age configured for 10886400 seconds.
- includeSubDomains directive configured
- Preload directive configured

Server-side Input Validation

The following PHP Server-side Input Validation[53] code has been added to the Login page served by the Front-end Web Server:

```
<?php
if(isset($_POST['Submit'])){

  $uname=trim($_POST["uname"]);
  $upass=trim($_POST["upass"]);

  if($uname =="") {
    $errorMsg=  " Invalid User Name or Password.";
    $code= "1" ;
  }
  elseif($upass == "") {
    $errorMsg=  " Invalid User Name or Password.";
    $code= "2";
  }
}
else{
  echo "Success";
?>
```

[52] GlobalSign.com [2019] What Is HSTS and How Do I Implement It? https://www.globalsign.com/en/blog/what-is-hsts-and-how-do-i-use-it/ [Accessed 1st August 2019]

[53] W3Schools.com [2019] PHP 5 Form Validation https://www.w3schools.com/php/php_form_validation.asp [Accessed 1st August 2019]

Session Management

Session management for the Secure Enterprise Blog application is managed by the Laravel Framework. Session Management Configuration files are located at:

SEB\vendor\laravel\framework\src\Illuminate\Session

The table below details the Session Management Configuration for the secure enterprise blog application:

Function	File	Value
Session Encryption	SEB\config\session.php	Enabled
Session Authentication – AuthenticateSession Class	SEB\vendor\laravel\framework\src \Illuminate\Session\Middleware \AuthenticateSession.php	Configured
Session Authentication – Store Users password Hash in Session	SEB\vendor\laravel\framework\src \Illuminate\Session\Middleware \AuthenticateSession.php	Configured
Session Authentication – Securely Log User out of Session	SEB\vendor\laravel\framework\src \Illuminate\Session\Middleware \AuthenticateSession.php	Configured
Session Database Connection Configuration	SEB\config\session.php	MySQL
Session Database Table Configuration	SEB\config\session.php	session
Session Cookie Name	SEB\config\session.php	seb_session
Session Cookie Storage Location	SEB\storage\framework\sessions\...	File on disk
Session Cookie Path	SEB\config\session.php	/
Session Cookie Domain	SEB\config\session.php	www.SEB.com
Session Cookie HTTPS Only	SEB\config\session.php	Enabled
Session Cookie SameSite	SEB\config\session.php	Strict
Session Duration Configuration	SEB\config\cms.php	Dev: true - session never expires Prod: false - session is time limited
Session Lifetime Configuration	SEB\config\session.php	lifetime => 120 minutes
Session Lifetime Configuration	SEB\config\session.php	expire_on_close => true

Table 9. Session Management Configuration[54]

[54] Taaffe, Jonathon [2019] Table 9. Session Management Configuration [Created 1st August 2019]

Configuration Code

The following screen shots show the function, file and associated code to enable the Session Management configuration documented in the table above

Session Encryption

SEB\config\session.php

```
34      | Session Encryption
35      |--------------------------------------------------------------------------
36      | This option allows you to easily specify that all of your session data
37      | should be encrypted before it is stored. All encryption will be run
38      | automatically by Laravel and you can use the Session like normal.
39      */
40      'encrypt' => true,
```

Session Authentication – AuthenticateSession Class

SEB\vendor\laravel\framework\src\Illuminate\Session\Middleware\AuthenticateSession.php

```
 9  class AuthenticateSession
10  {
11      protected $auth;
12      public function __construct(AuthFactory $auth)
13      {
14          $this->auth = $auth;
15      }
```

Session Authentication – Store Users password Hash in Session

SEB\vendor\laravel\framework\src\Illuminate\Session\Middleware\AuthenticateSession.php

```
57      protected function storePasswordHashInSession($request)
58      {
59          if (! $request->user()) {
60              return;
61          }
62
63          $request->session()->put([
64              'password_hash' => $request->user()->getAuthPassword(),
```

Session Authentication – Securely Log User out of Session

SEB\vendor\laravel\framework\src\Illuminate\Session\Middleware\AuthenticateSession.php

```
76      protected function logout($request)
77      {
78          $this->auth->logout();
79
80          $request->session()->flush();
81
82          throw new AuthenticationException;
```

Session Database Connection Configuration
SEB\config\session.php

```
56      |--------------------------------------------------------------------------
57      | Session Database Connection
58      |--------------------------------------------------------------------------
59      | When using the "database" or "redis" session drivers, you may specify a
60      | connection that should be used to manage these sessions. This should
61      | correspond to a connection in your database configuration options.
62      */
63      'connection' => mysql,
```

Session Database Table Configuration
SEB\config\session.php

```
66      |--------------------------------------------------------------------------
67      | Session Database Table
68      |--------------------------------------------------------------------------
69      | When using the "database" session driver, you may specify the table we
70      | should use to manage the sessions. Of course, a sensible default is
71      | provided for you; however, you are free to change this as needed.
72      */
73      'table' => 'sessions',
```

Session Cookie Name
SEB\config\session.php

```
89      |--------------------------------------------------------------------------
90      | Session Cookie Name
91      |--------------------------------------------------------------------------
92      | Here you may change the name of the cookie used to identify a session
93      | instance by ID. The name specified here will get used every time a
94      | new session cookie is created by the framework for every driver.
95      */
96      'cookie' => 'seb_session',
```

Session Cookie Storage Location
SEB\storage\framework\sessions\...

Session Cookie Path
SEB\config\session.php

```
99      |--------------------------------------------------------------------------
100     | Session Cookie Path
101     |--------------------------------------------------------------------------
102     | The session cookie path determines the path for which the cookie will
103     | be regarded as available. Typically, this will be the root path of
104     | your application but you are free to change this when necessary.
105     */
106     'path' => '/',
```

Session Cookie Domain

SEB\config\session.php

```
109        |--------------------------------------------------------------------------
110        | Session Cookie Domain
111        |--------------------------------------------------------------------------
112        | Here you may change the domain of the cookie used to identify a session
113        | in your application. This will determine which domains the cookie is
114        | available to in your application. A sensible default has been set.
115        */
116        'domain' => www.seb.com,
```

Session Cookie HTTPS Only

SEB\config\session.php

```
119        |--------------------------------------------------------------------------
120        | HTTPS Only Cookies
121        |--------------------------------------------------------------------------
122        | By setting this option to true, session cookies will only be sent back
123        | to the server if the browser has a HTTPS connection. This will keep
124        | the cookie from being sent to you if it can not be done securely.
125        */
126        'secure' => true,
```

Session Cookie SameSite

SEB\config\session.php

```
130        | Same-Site Cookies
131        |--------------------------------------------------------------------------
132        | This option determines how your cookies behave when cross-site requests
133        | take place, and can be used to mitigate CSRF attacks. By default, we
134        | do not enable this as other CSRF protection services are in place.
135        | In the strict mode, the cookie is not sent with any cross-site usage
136        | even if the user follows a link to another website. Lax cookies are
137        | only sent with a top-level get request.
138        | Supported: "lax", "strict"
139        */
140        'same_site' => strict,
```

Session Duration Configuration

SEB\config\cms.php

```
53         | Back-end login remember
54         |--------------------------------------------------------------------------
55         | Define live duration of backend sessions :
56         | true  - session never expire (cookie expiration in 5 years)
57         | false - session have a limited time (see session.lifetime)
58         | null  - The form login display a checkbox that allow user to choose
59         |         wanted behavior
60         */
61         'backendForceRemember' => true,
```

Session Lifetime Configuration

SEB\config\session.php

```
22      |--------------------------------------------------------------------------
23      | Session Lifetime
24      |--------------------------------------------------------------------------
25      | Here you may specify the number of minutes that you wish the session
26      | to be allowed to remain idle for it is expired. If you want them
27      | to immediately expire when the browser closes, set it to zero.
28      */
29      'lifetime' => 120,
30      'expire_on_close' => true,
```

Logging

System, Access and Event actions are all being logged. This data can be used to determine if any malicious activity is being directed at the application. Only Administrators and Super Users have access to the log data.

The Logging configuration is as follows:

Function	File	Value
System Logs Location	SEB\storage\logs\system.log	
Logging level Configuration	SEB\config\app.php	single
Access Logs – Configuration	SEB\modules\backend\controllers \accesslogs\config_filter.yaml	scopes
Access Logs – Database Configuration	SEB\modules\backend\models \AccessLog.php	class AccessLog
Access Logs – Display Entries	SEB\modules\backend\models \AccessLog.php	getRecent
Event Logs – Controller	SEB\modules\system\controllers \EventLogs.php	class EventLogs
Event Logs – Database Configuration	SEB\modules\system\models \EventLog.php	system_event_logs
Event Logs – Create an Event Log Entry	SEB\modules\system\models \EventLog.php	add($message, $level = 'info', $details=null)
Event Logs – Request Logs	SEB\modules\system\controllers \EventLogs.php	RequestLogs

Table 10. Logging Configuration[55]

[55] Taaffe, Jonathon [2019] Table 10. Logging Configuration [Created 1st August 2019]

Configuration Code

The following screen shots show the function, file and associated code to enable the Logging configuration documented in the table above.

System Logs Location
SEB\storage\logs\system.log

Logging level Configuration
SEB\config\app.php

```
109        |--------------------------------------------------------------------------
110        | Logging Configuration
111        |--------------------------------------------------------------------------
112        | Here you may configure the log settings for your application. Out of
113        | the box, Laravel uses the Monolog PHP logging library. This gives
114        | you a variety of powerful log handlers / formatters to utilize.
115        | Available Settings: "single", "daily", "syslog", "errorlog"
116        */
117
118        'log' => 'single',
```

Access Logs – Configuration
SEB\modules\backend\controllers\accesslogs\config_filter.yaml

```
5   scopes:
6
7       created_at:
8           label: backend::lang.access_log.created_at
9           type: daterange
10          conditions: created_at >= ':after' AND created_at <= ':before'
11
12      user:
13          label: backend::lang.access_log.login
14          modelClass: Backend\Models\User
15          conditions: user_id in (:filtered)
16          nameFrom: login
```

Access Logs – Database Configuration

SEB\modules\backend\models\AccessLog.php

```
12  class AccessLog extends Model
13  {
14      protected $table = 'backend_access_log';
15      public $belongsTo = [
16          'user' => User::class
17      ];
18      public static function add($user)
19      {
20          $record = new static;
21          $record->user = $user;
22          $record->ip_address = Request::getClientIp();
23          $record->save();
24
25          return $record;
```

Access Logs – Display Entries

SEB\modules\backend\models\AccessLog.php

```
33      public static function getRecent($user)
34      {
35          $records = static::where('user_id', $user->id)
36              ->orderBy('created_at', 'desc')
37              ->limit(2)
38              ->get()
39          ;
40
41          if (!count($records)) {
42              return null;
43          }
44
45          $first = $records->first();
46
47          return !$first->created_at->isToday() ? $first : $records->pop();
```

Event Logs – Controller

SEB\modules\system\controllers\EventLogs.php

```php
class EventLogs extends Controller
{
    public $implement = [
        \Backend\Behaviors\FormController::class,
        \Backend\Behaviors\ListController::class
    ];

    public $formConfig = 'config_form.yaml';
    public $listConfig = 'config_list.yaml';
    public $requiredPermissions = ['system.access_logs'];
```

Event Logs – Database Configuration

SEB\modules\system\models\EventLog.php

```php
class EventLog extends Model
{
    protected $table = 'system_event_logs';
    protected $jsonable = ['details'];
    public static function useLogging()
    {
        return (
            class_exists('Model') &&
            Model::getConnectionResolver() &&
            App::hasDatabase() &&
            !defined('OCTOBER_NO_EVENT_LOGGING') &&
            LogSetting::get('log_events')
        );
    }
```

Event Logs – Create an Event Log Entry

SEB\modules\system\models\EventLog.php

```
36      public static function add($message, $level = 'info', $details = null)
37      {
38          $record = new static;
39          $record->message = $message;
40          $record->level = $level;
41
42          if ($details !== null) {
43              $record->details = (array) $details;
44          }
45
46          try {
47              $record->save();
48          }
49          catch (Exception $ex) {}
50
51          return $record;
```

Event Logs – Request Logs

SEB\modules\system\controllers\EventLogs.php

```
16  class RequestLogs extends Controller
17  {
18      public $implement = [
19          \Backend\Behaviors\FormController::class,
20          \Backend\Behaviors\ListController::class
21      ];
22      public $formConfig = 'config_form.yaml';
23      public $listConfig = 'config_list.yaml';
24      public $requiredPermissions = ['system.access_logs'];
```

Testing

Introduction

This test plan identifies the secure enterprise blog application components and features to be tested, the types of testing, the individuals responsible for testing, the resources required for a successful test and includes the testing schedule.

In Scope

All features of the application as outlined below are to be tested:

Area	Action	Role	Account
Security	Login Authentication	Bloggers Administrators Super Users	Blogger01 seb_admin01 seb_superuser01
Security	View Account Details	Bloggers Administrators Super Users	Blogger01 seb_admin01 seb_superuser01
Security	Modify Account Details	Bloggers Administrators Super Users	Blogger01 seb_admin01 seb_superuser01
Blog Content	View All Blog Posts	Bloggers Administrators	Blogger01 seb_admin01
Blog Content	View Own Blog Posts	Bloggers Administrators	Blogger01 seb_admin01
Blog Content	Create New Blog Post	Bloggers Administrators	Blogger01 seb_admin01
Blog Content	Modify Own Blog Posts	Bloggers Administrators	Blogger01 seb_admin01
Blog Content	Publish New Blog Post	Bloggers Administrators	Blogger01 seb_admin01
Blog Content	Update All Blog Posts	Administrators	seb_admin01
Blog Content	Delete All Blog Posts	Administrators	seb_admin01
Blog Content	Export All Blog Posts	Administrators	seb_admin01
Blog Content	Import Blog Posts	Administrators	seb_admin01
Security	Create User Accounts	Administrators	seb_admin01
Security	Activate User Accounts	Administrators	seb_admin01
Security	Modify User Accounts	Administrators	seb_admin01
Security	Delete User Accounts	Administrators	seb_admin01
Security	View Access Logs	Administrators Super Users	seb_admin01 seb_superuser01
Security	View Event Logs	Administrators Super Users	seb_admin01 seb_superuser01
Security	Create Administrator Accounts	Super Users	seb_superuser01
Security	Activate Administrator Accounts	Super Users	seb_superuser01
Security	Modify Administrator Accounts	Super Users	seb_superuser01
Security	Delete Administrator Accounts	Super Users	seb_superuser01

Security	Manage Roles	Super Users	seb_superuser01
Security	Manage Password Policy	Super Users	seb_superuser01
System	Manage Backend Configuration	Super Users	seb_superuser01
System	Manage Software Updates	Super Users	seb_superuser01
Security	Securely Logout	Bloggers Administrators Super Users	Blogger01 seb_admin01 seb_superuser01

Table 11. Application Features Testing[56]

Out of Scope

The following features are not in scope for testing:

- User Interface
- Hardware
- Software Framework
- Database Structure
- Communications Methods

Quality Objective

This test plan will verify the functionality of the secure enterprise blog application. The key features to be tested include

- Authentication and Authorisation
- Content Management
- Account Management
- System Management

Test Strategy

3 functional tests will be performed to verify application quality and reliability and to identify any errors. The functional tests are:

- Basic Test: Test functionality of the application
- System Test: Test behaviour of the application
- Security Test: Test security of the application

[56] Taaffe, Jonathon [2019] Table 11. Application Features Testing [Created 1st August 2019]

Application Functional Test[57]

A total of 25 functional tests were successfully completed against all features of the application. Below are the summary test results of all tests completed:

Test ID	Area	Test Activity	Role
SEB000001	Security	Login Authentication	Blogger
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Connect to application URL	Connect to application URL	As expected
2	View application login page	View application login page	As expected
3	Enter login credentials	Enter login credentials	As expected
4	Login to application	Login to application	As expected

Test ID	Area	Test Activity	Role
SEB000002	Security	View Account Details	Blogger
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Access My Account	Access My Account details	As expected
3	View My Account Details	View My Account details	As expected

Test ID	Area	Test Activity	Role
SEB000003	Security	Modify Account Details	Blogger
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Access My Account	Access My Account details	As expected
3	View My Account Details	View My Account details	As expected
4	Change your Last Name	Change Last Name	As expected
5	Save Changes	Successfully save change	As expected

Test ID	Area	Test Activity	Role
SEB000004	Security	Securely Logout	Blogger
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Click Sign Out	Log out of application	As expected
3	Log out of application	Log out of application	As expected

[57] Taaffe, Jonathon [2019] Table 12. Application Functional Test Tables [Created 1st August 2019]

Test ID	Area	Test Activity	Role
SEB000005	Blog Content	View Own Blog Posts	Blogger
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Click Sign Out	Log out of application	As expected
3	Log out of application	Log out of application	As expected

Test ID	Area	Test Activity	Role
SEB000006	Blog Content	Create New Blog Post	Blogger
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Create New Blog Post	Create New Blog Post	As expected
3	Save New Blog Post	Save New Blog Post	As expected

Test ID	Area	Test Activity	Role
SEB000007	Blog Content	Modify Own Blog Posts	Blogger
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Modify Own Blog Posts	Modify Own Blog Posts	As expected
3	Save Modified Blog Post	Save Modified Blog Post	As expected

Test ID	Area	Test Activity	Role
SEB000008	Blog Content	Modify Own Blog Posts	Blogger
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Modify Own Blog Posts	Modify Own Blog Posts	As expected
3	Save Modified Blog Post	Save Modified Blog Post	As expected

Test ID	Area	Test Activity	Role
SEB000009	Blog Content	Update Other Users Blog Post	Administrator
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Update Other Users Blog Posts	Update Other Users Blog Posts	As expected
3	Save Modified Blog Post	Save Modified Blog Post	As expected

Test ID	Area	Test Activity	Role
SEB000010	Blog Content	Export All Blog Posts	Administrator
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Export All Blog Posts	Export All Blog Posts	As expected

Test ID	Area	Test Activity	Role
SEB000011	Blog Content	Import Blog Posts	Administrator
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Import Blog Posts	Import Blog Posts	As expected

Test ID	Area	Test Activity	Role
SEB000012	Account Management	Create New User Account	Administrator
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Create New User Account	Create New User Account	As expected
3	Save New User Account	Save New User Account	As expected

Test ID	Area	Test Activity	Role
SEB000013	Account Management	Modify Existing User Account	Administrator
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Modify Existing User Account	Modify Existing User Account	As expected
3	Save Modified User Account	Save Modified User Account	As expected

Test ID	Area	Test Activity	Role
SEB000014	Account Management	Delete Existing User Account	Administrator
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Delete Existing User Account	Delete Existing User Account	As expected

Test ID	Area	Test Activity	Role
SEB000015	System Administration	View Access Logs	Administrator
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	View Access Logs	View Access Logs	As expected

Test ID	Area	Test Activity	Role
SEB000016	System Administration	View Event Logs	Administrator
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	View Event Logs	View Access Logs	As expected

Test ID	Area	Test Activity	Role
SEB000017	Account Management	Create New Admin Account	Super User
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Create New Admin Account	Create New Admin Account	As expected
3	Save New Admin Account	Save New Admin Account	As expected

Test ID	Area	Test Activity	Role
SEB000018	Account Management	Modify Existing Admin Account	Super User
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Modify Existing Admin Account	Modify Existing Admin Account	As expected
3	Save Modified Admin Account	Save Modified Admin Account	As expected

Test ID	Area	Test Activity	Role
SEB000019	Account Management	Delete Existing Admin Account	Super User
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Delete Existing Admin Account	Delete Existing Admin Account	As expected

Test ID	Area	Test Activity	Role
SEB000020	System Administration	View Access Logs	Super User
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	View Access Logs	View Access Logs	As expected

Test ID	Area	Test Activity	Role
SEB000021	System Administration	View Event Logs	Super User
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	View Event Logs	View Access Logs	As expected

Test ID	Area	Test Activity	Role
SEB000022	System Administration	Manage Roles	Super User
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Manage Roles	Manage Roles	As expected
3	Save Changes	Save Changes	As expected

Test ID	Area	Test Activity	Role
SEB000023	System Administration	Manage Password Policy	Super User
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Manage Password Policy	Manage Password Policy	As expected
3	Save Changes	Save Changes	As expected

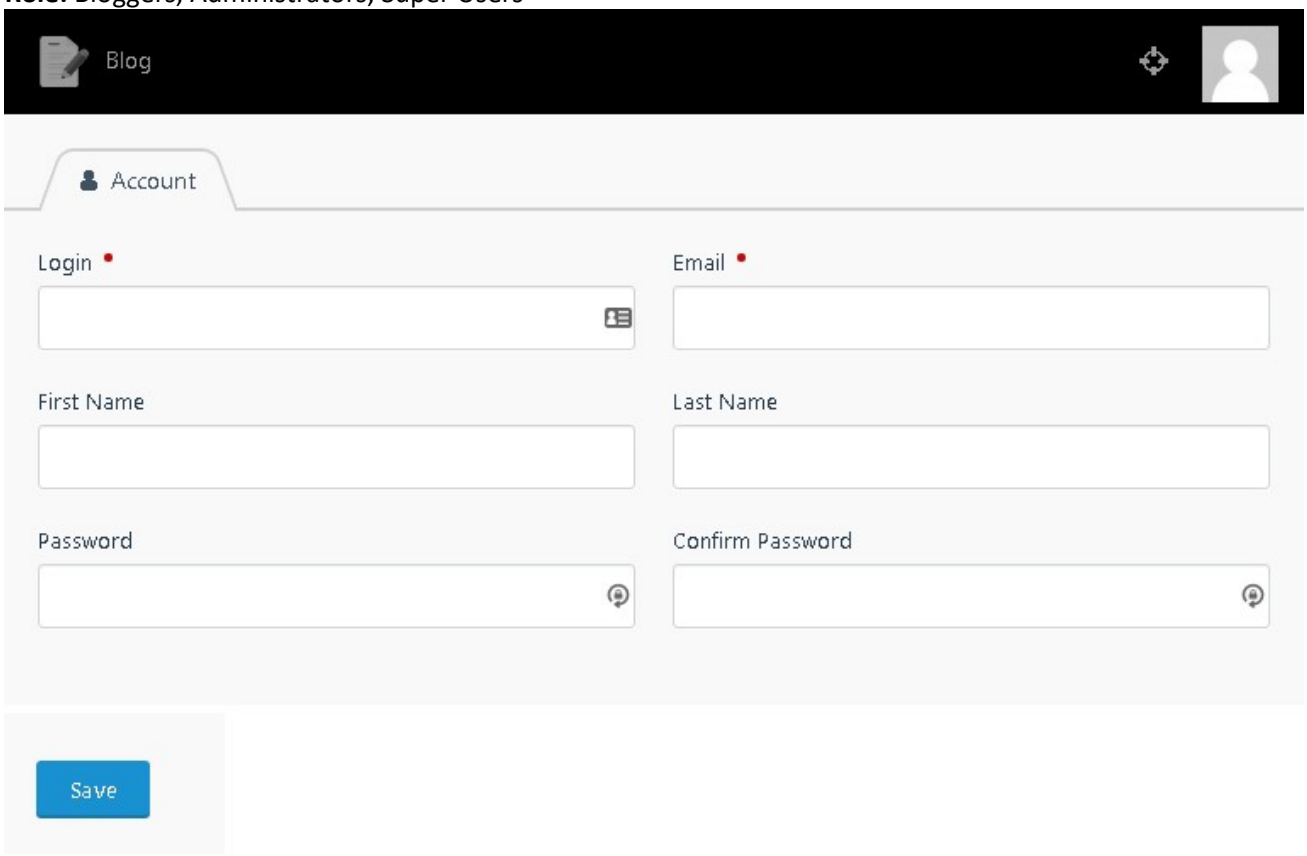
Test ID	Area	Test Activity	Role
SEB000024	System Administration	Manage Backend Configuration	Super User
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Manage Backend Configuration	Manage Backend Configuration	As expected
3	Save Changes	Save Changes	As expected

Test ID	Area	Test Activity	Role
SEB000025	System Administration	Manage Software Updates	Super User
Tester	**Reviewer**	**Test Date**	**Test Result**
Taaffe, J	Taaffe, J	09/08/2019	Pass
Step #	**Steps Details**	**Expected Results**	**Actual Results**
1	Login to application	Login to application	As expected
2	Manage Software Updates	Manage Software Updates	As expected
3	Save Changes	Save Changes	As expected

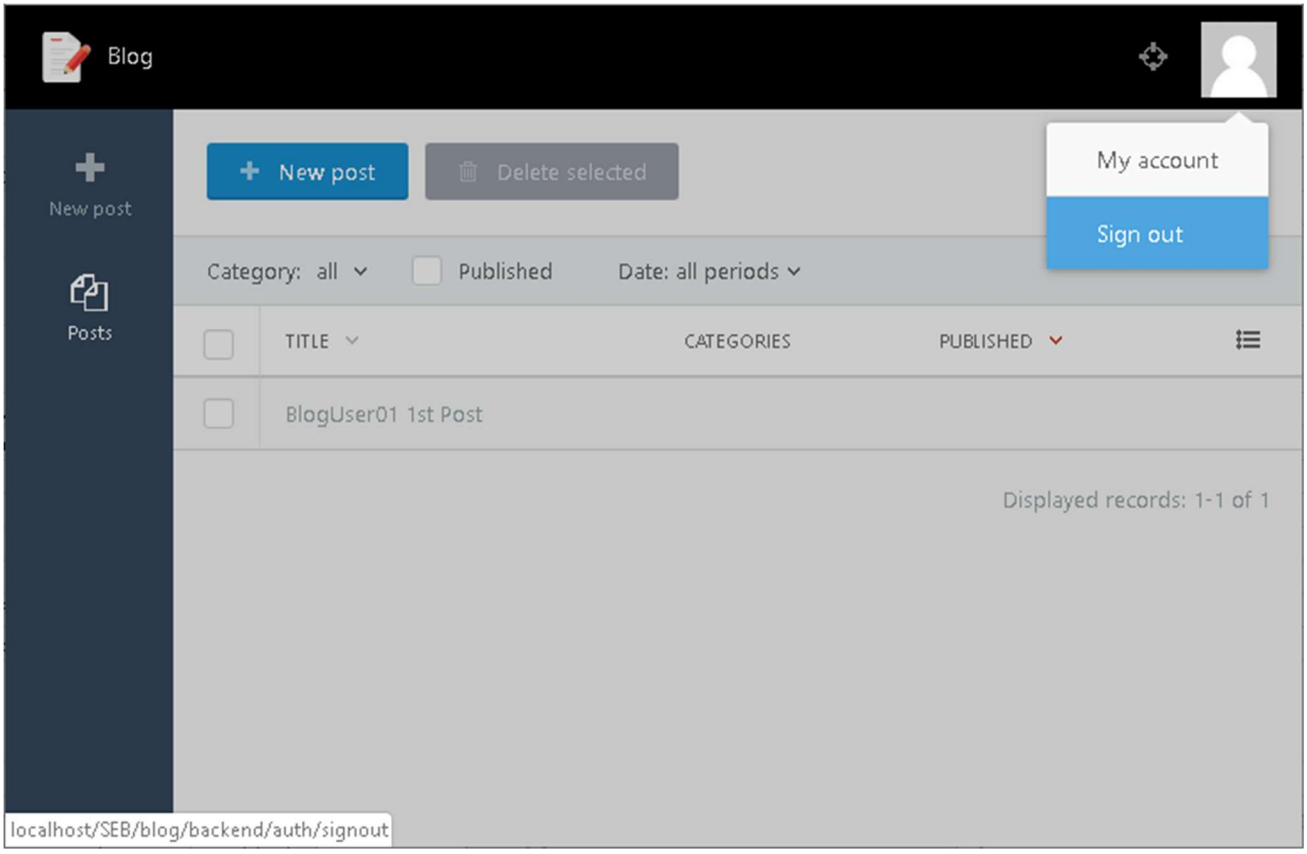
User Interface Design

Interface: Login
Roles: All Users

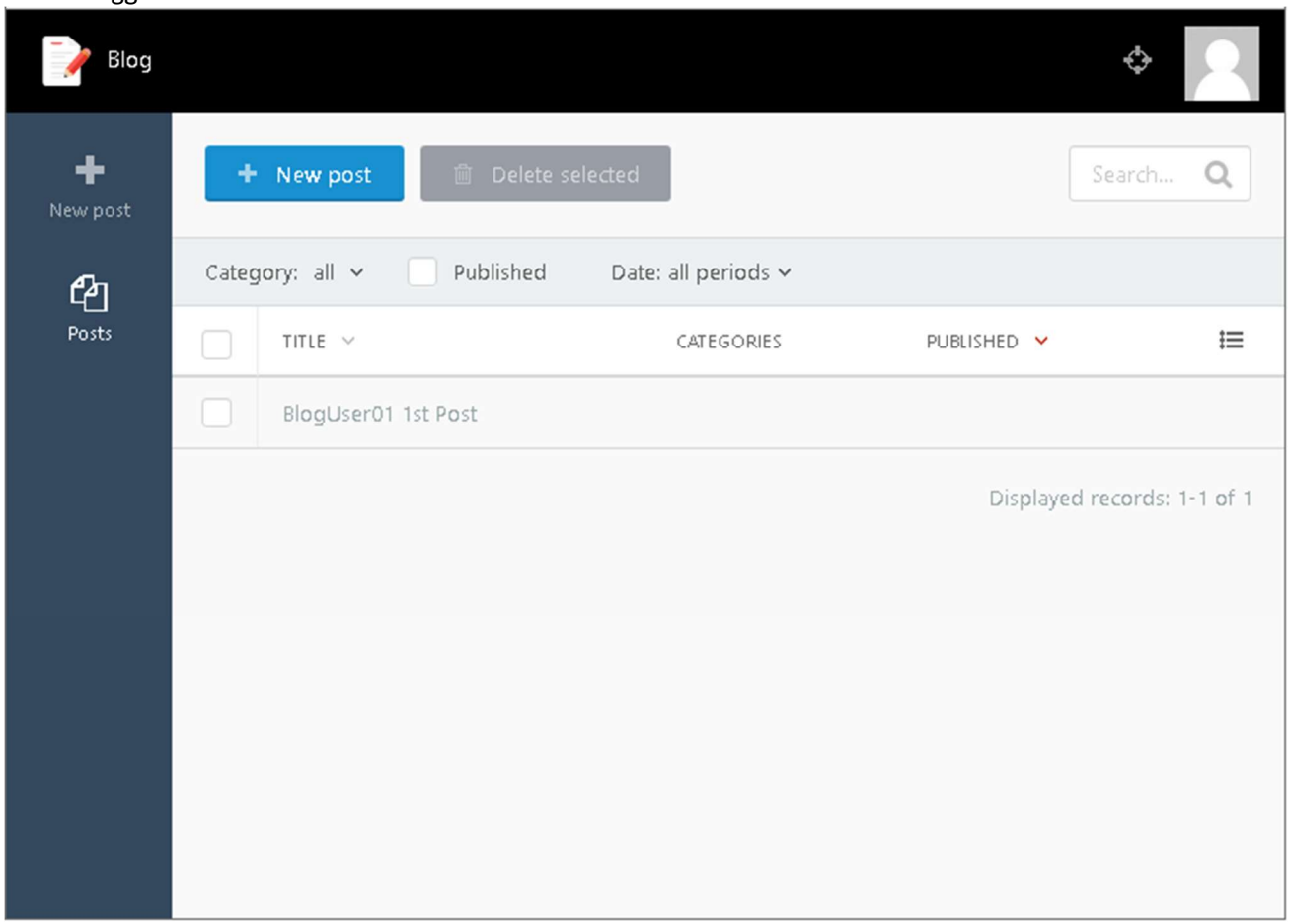
Interface: Account Management
Role: Bloggers, Administrators, Super Users



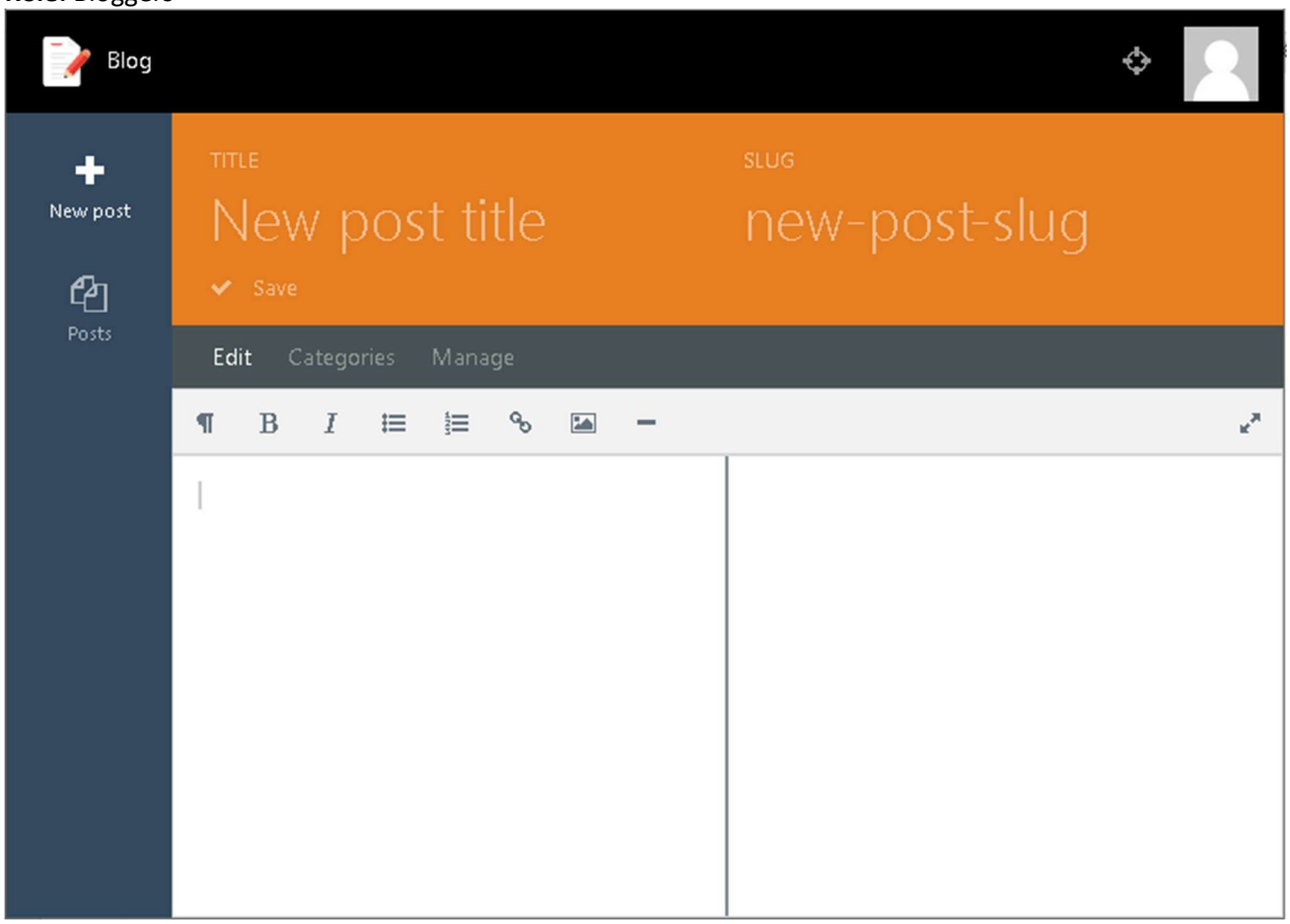
Interface: Securely Log Out
Role: Bloggers, Administrators, Super Users



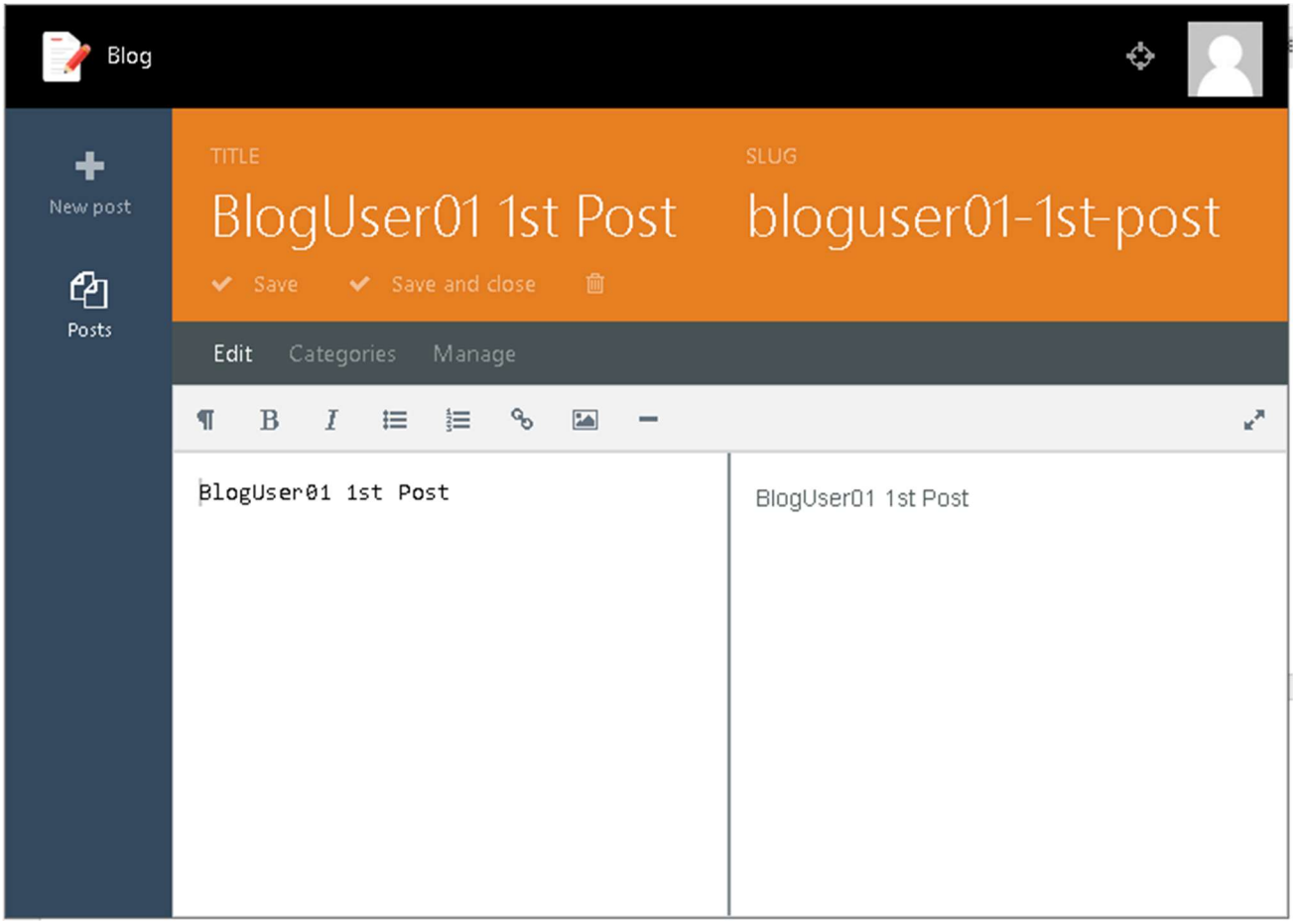
Interface: Blog Content Management
Role: Bloggers



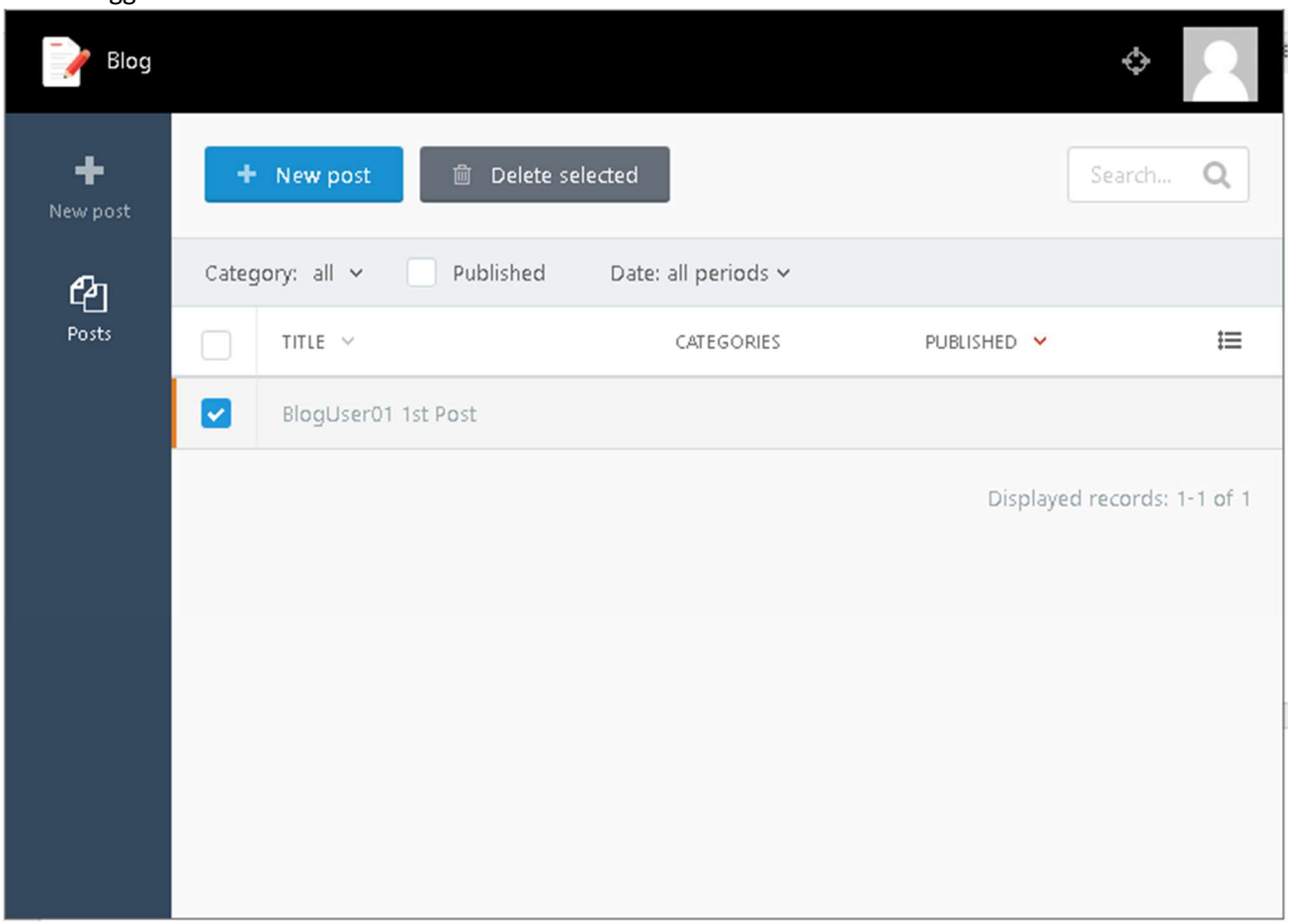
Interface: Blog Content Management – Create New Blog Post
Role: Bloggers



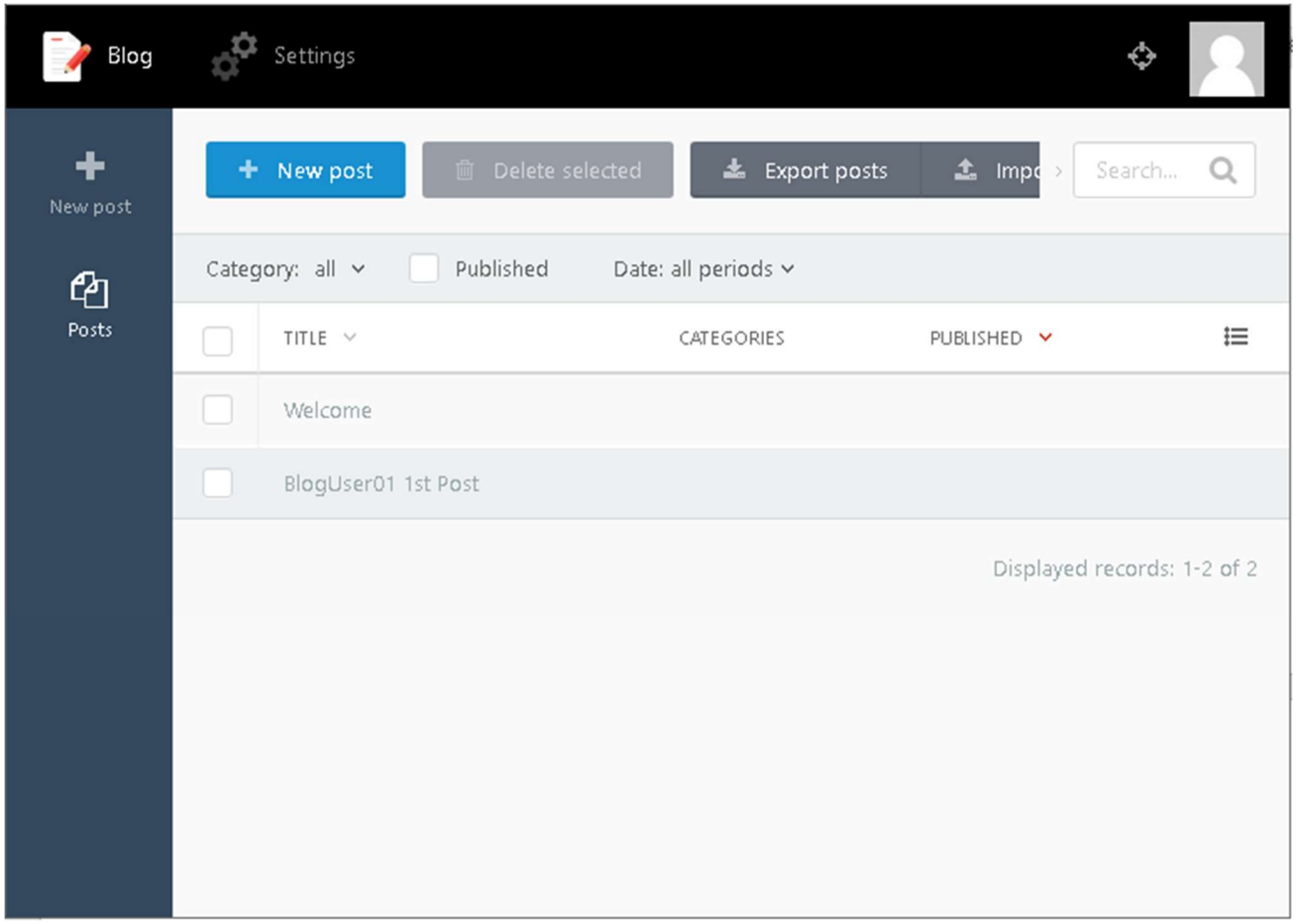
Interface: Blog Content Management – Modify Own Blog Post
Role: Bloggers



Interface: Blog Content Management – Delete Own Post
Role: Bloggers

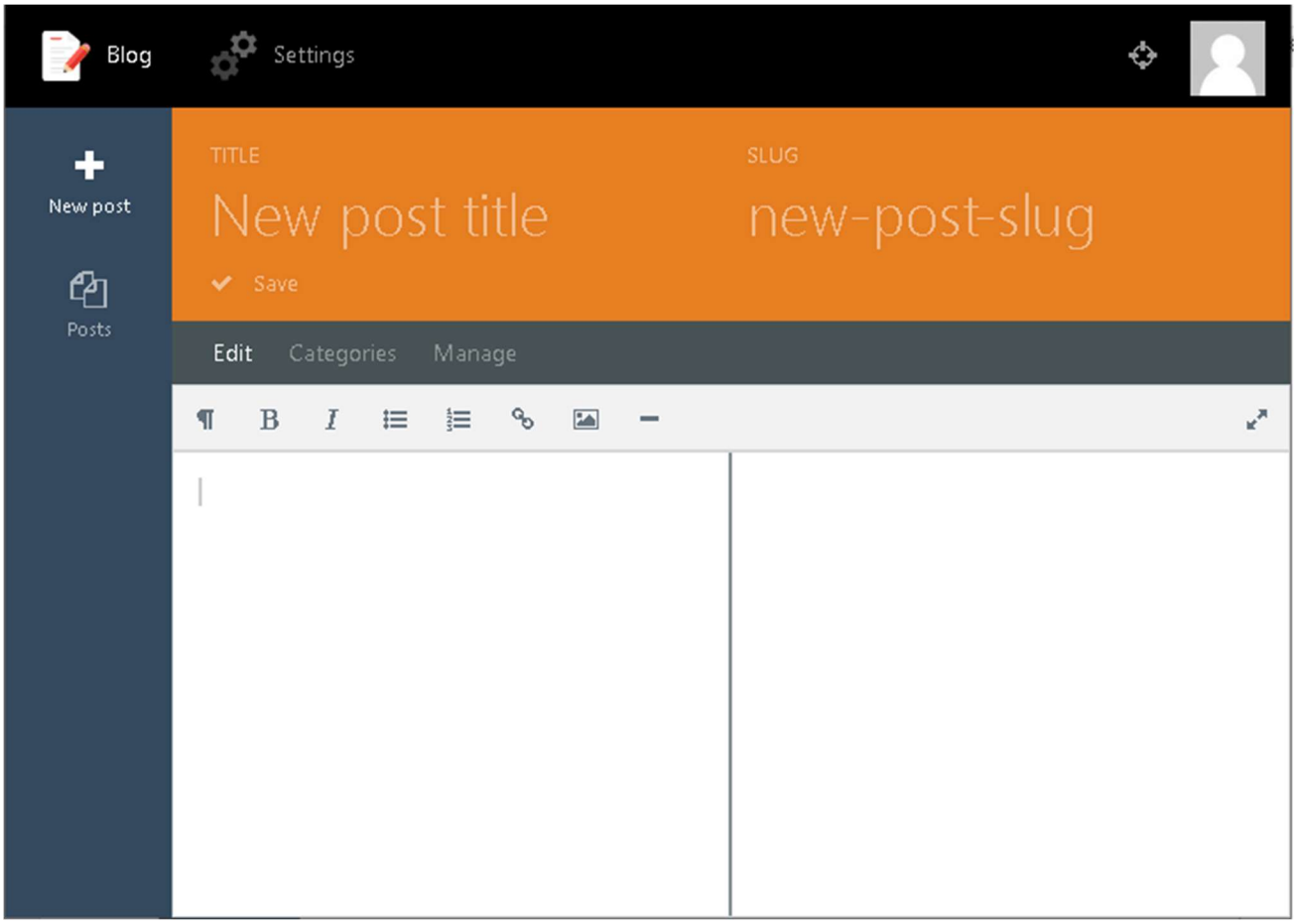


Interface: Blog Content Management
Role: Administrators



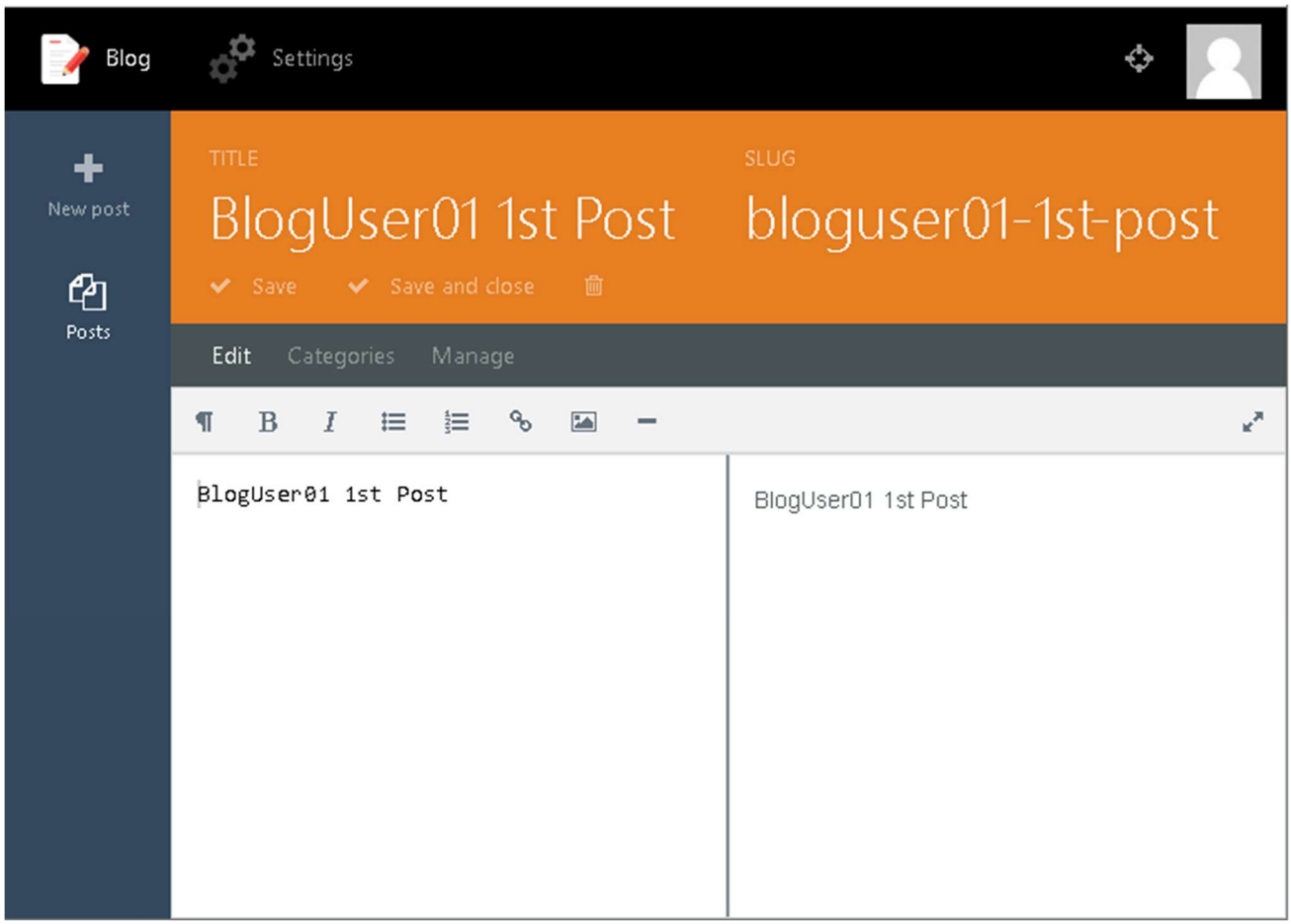
Interface: Blog Content Management – Create New Blog Post

Role: Administrators



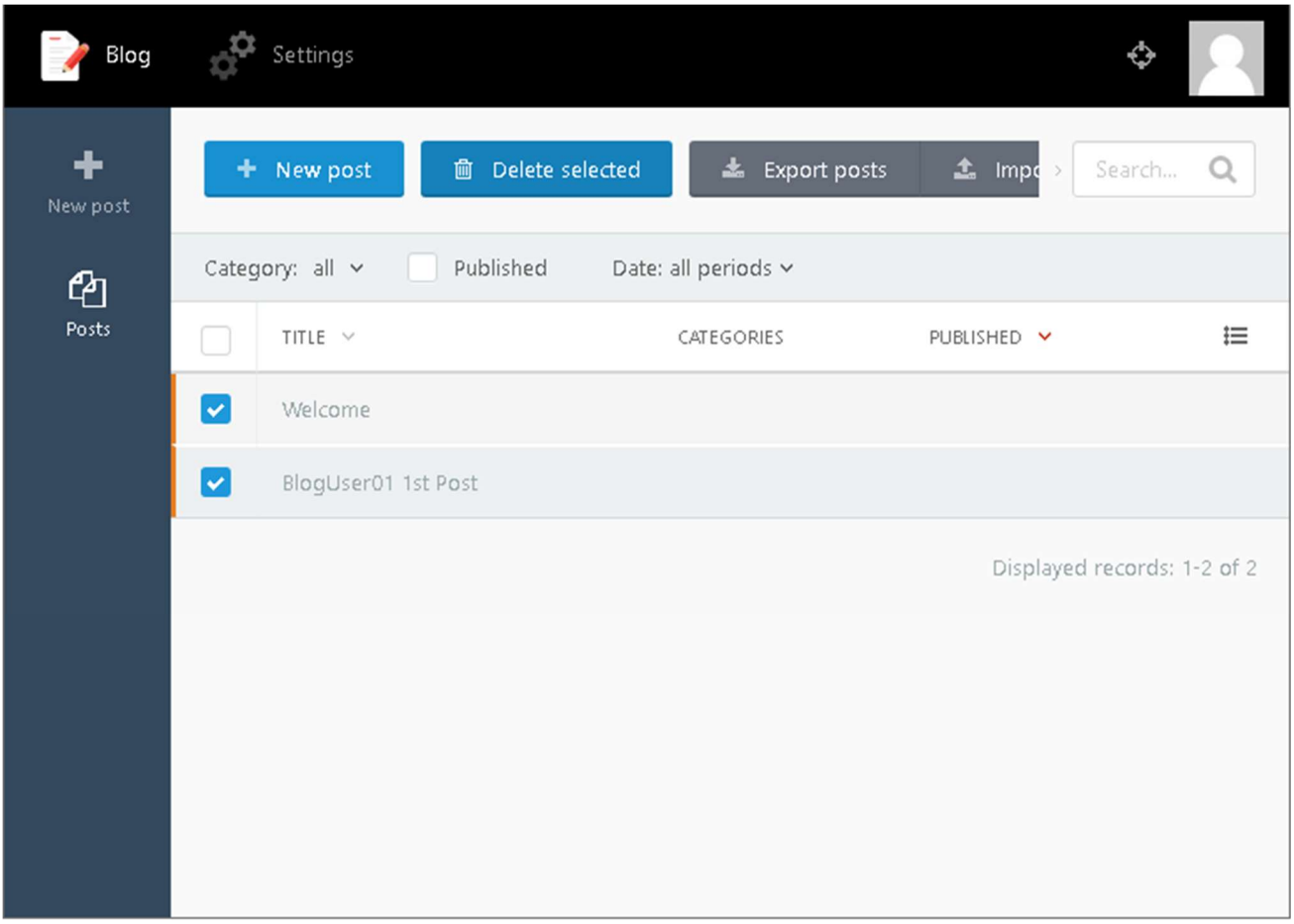
Interface: Blog Content Management – Modify All Blog Posts

Role: Administrators



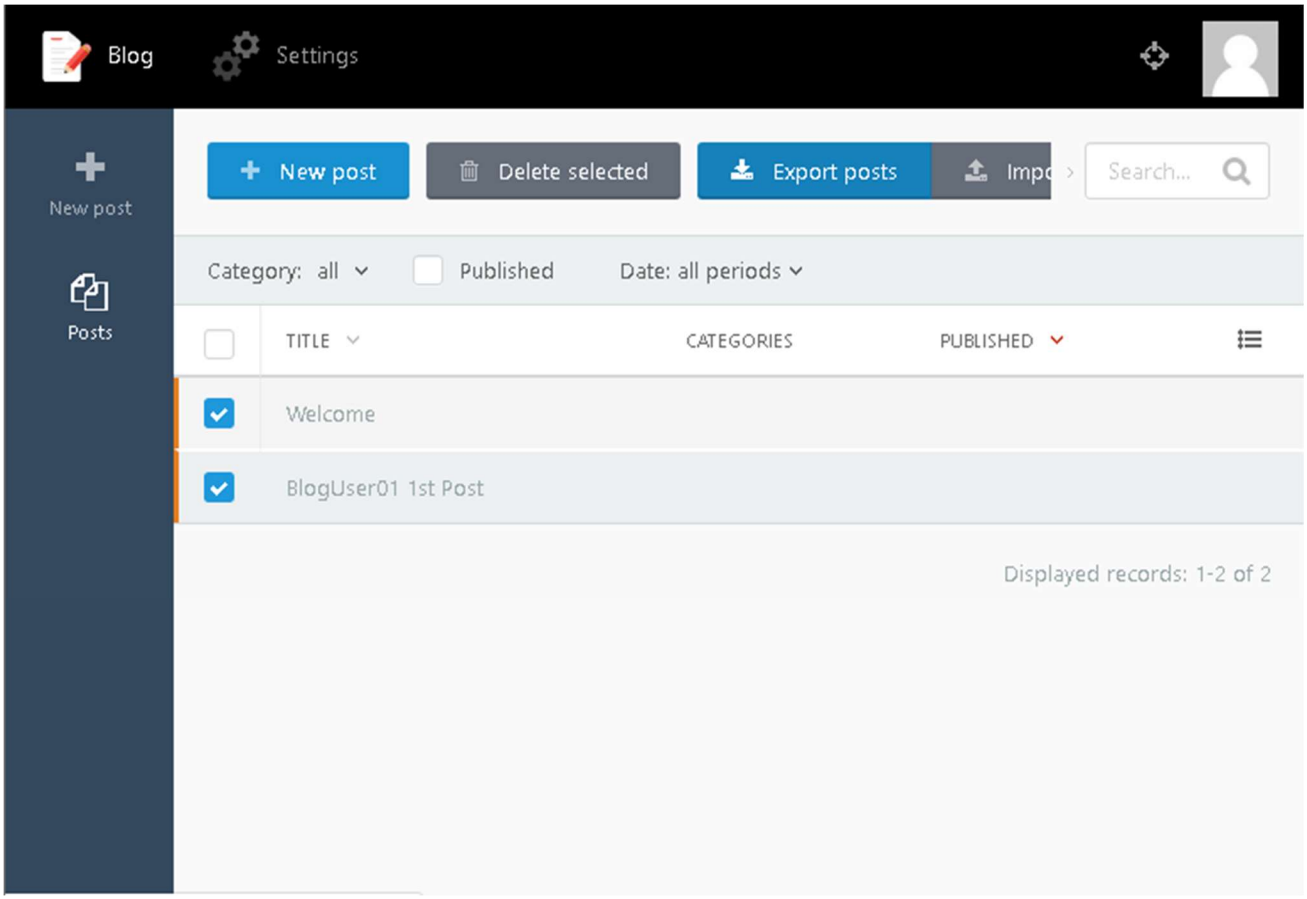
Interface: Blog Content Management – Delete Any/All Blog Post

Role: Administrators



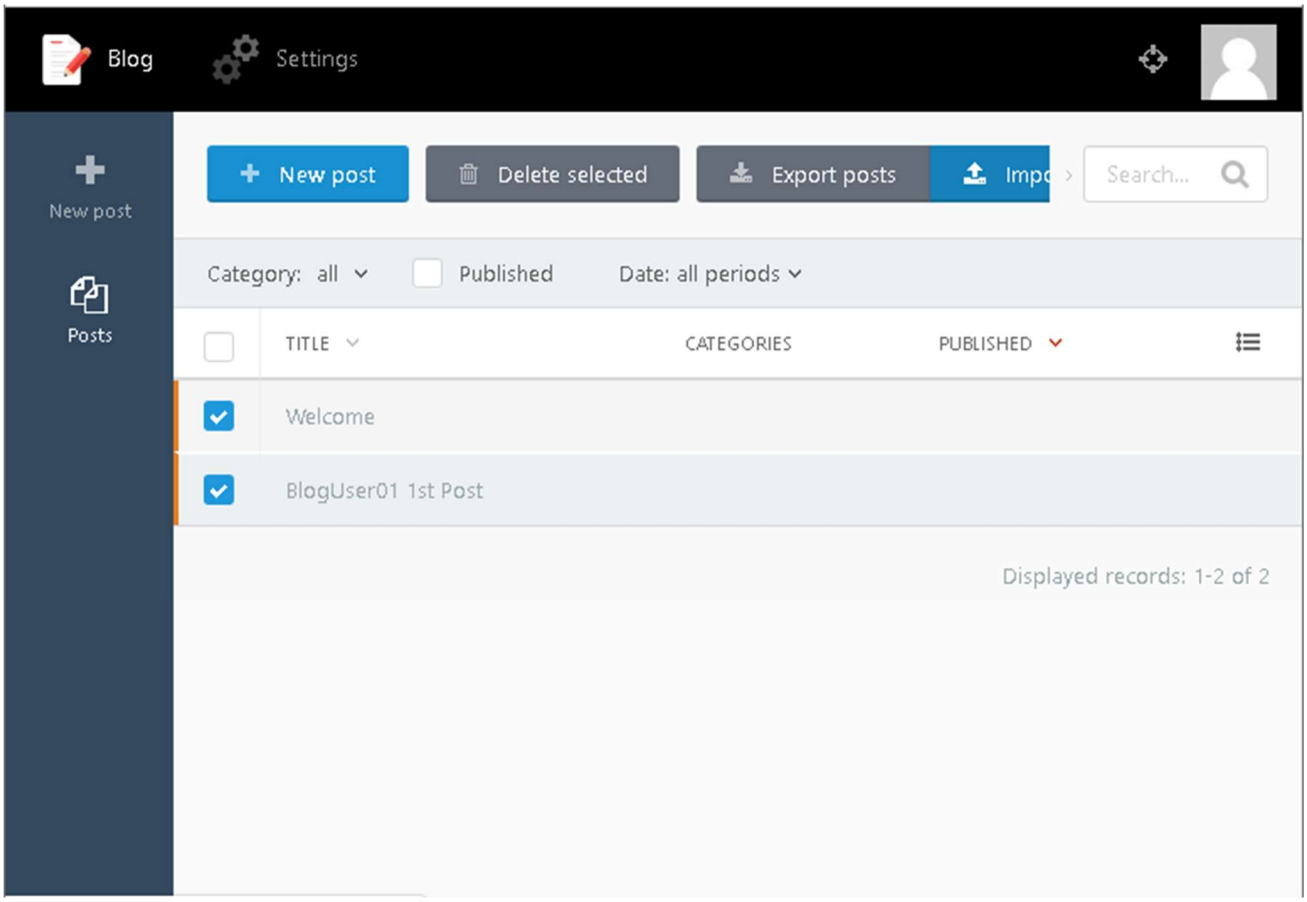
Interface: Blog Content Management – Export Blog Posts

Role: Administrators

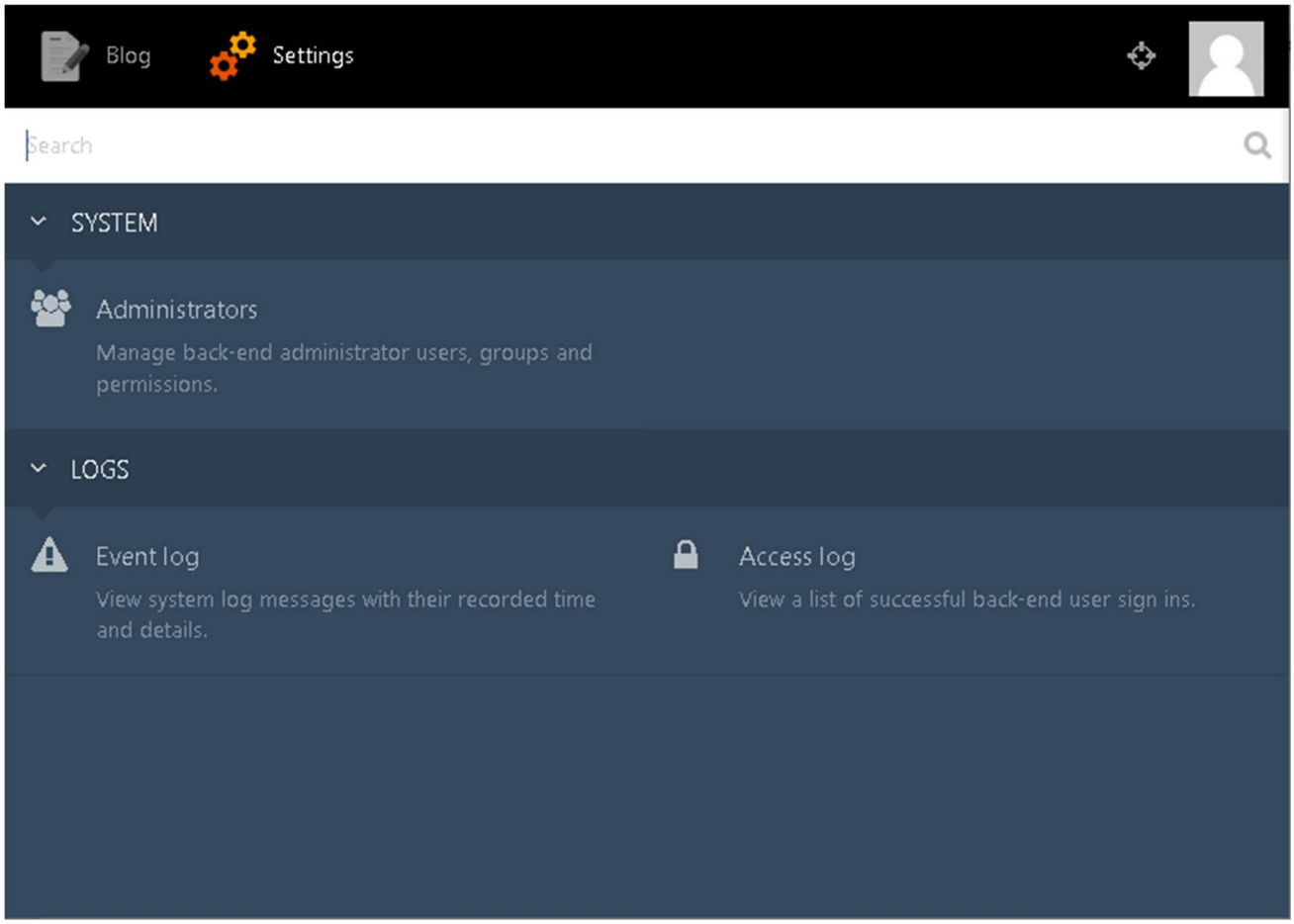


Interface: Blog Content Management – Import Blog Posts

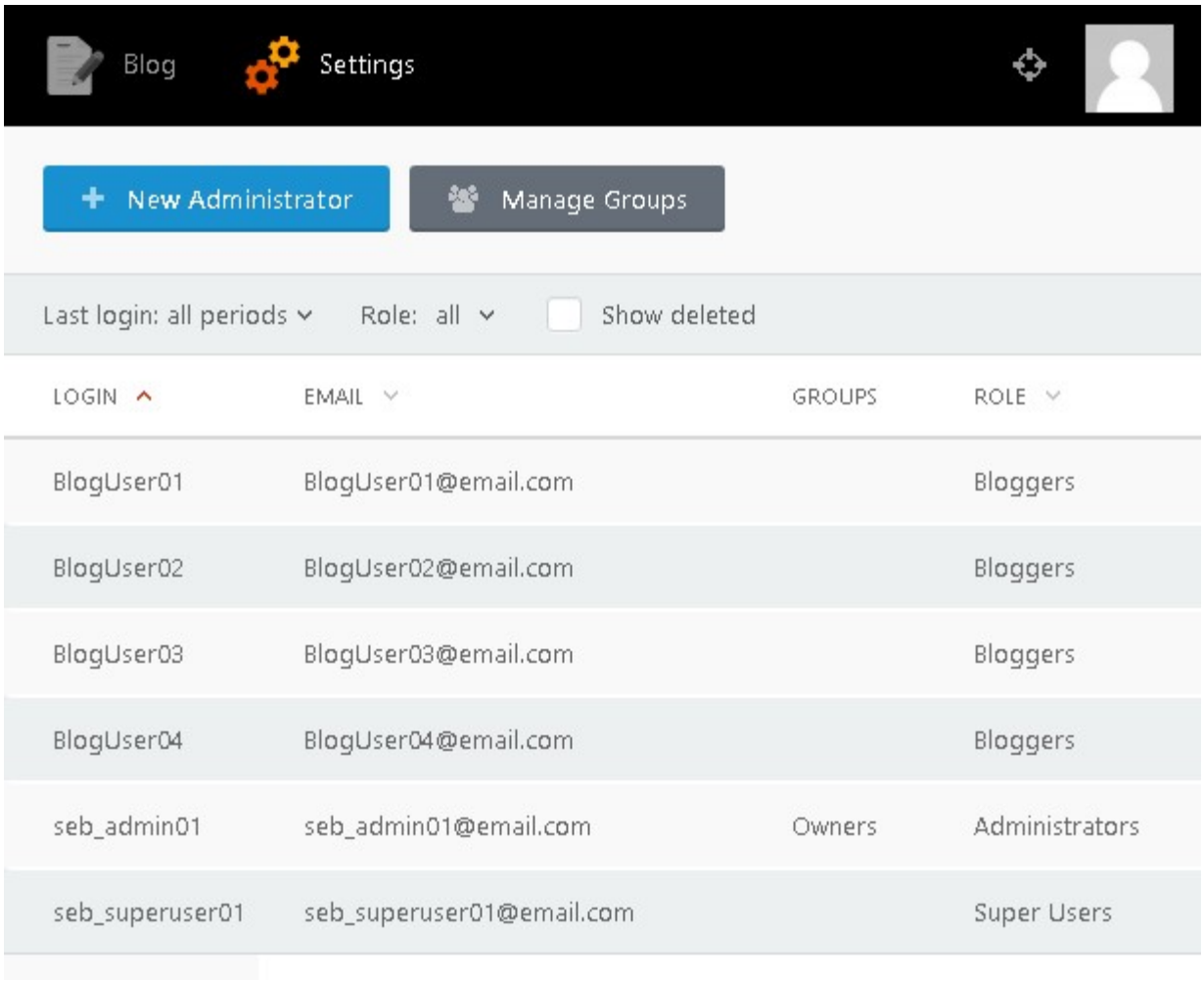
Role: Administrators



Interface: System Administration Console
Role: Administrators

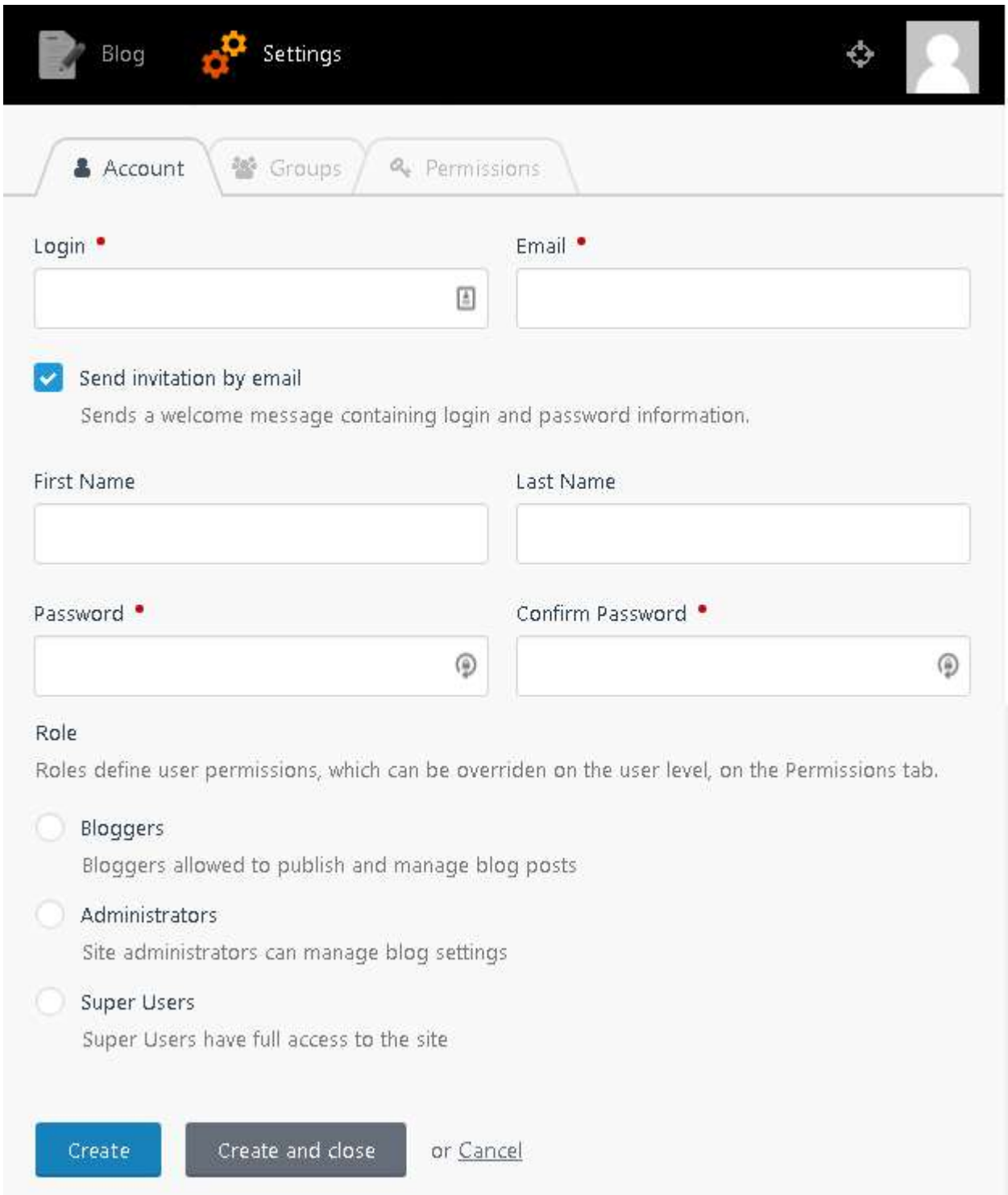


Interface: Account Management
Role: Administrators



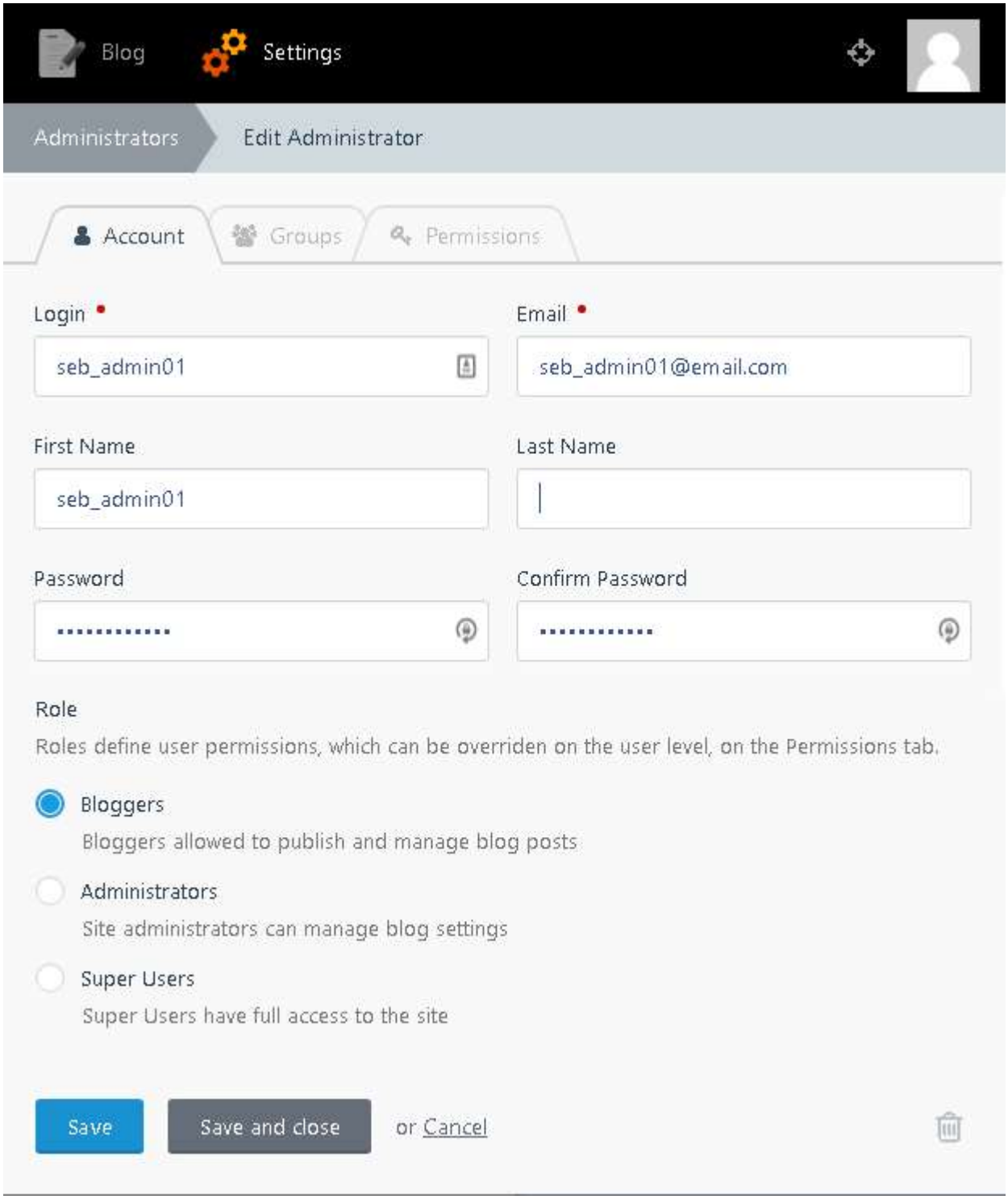
Interface: Account Management – Create a New User Account

Role: Administrators



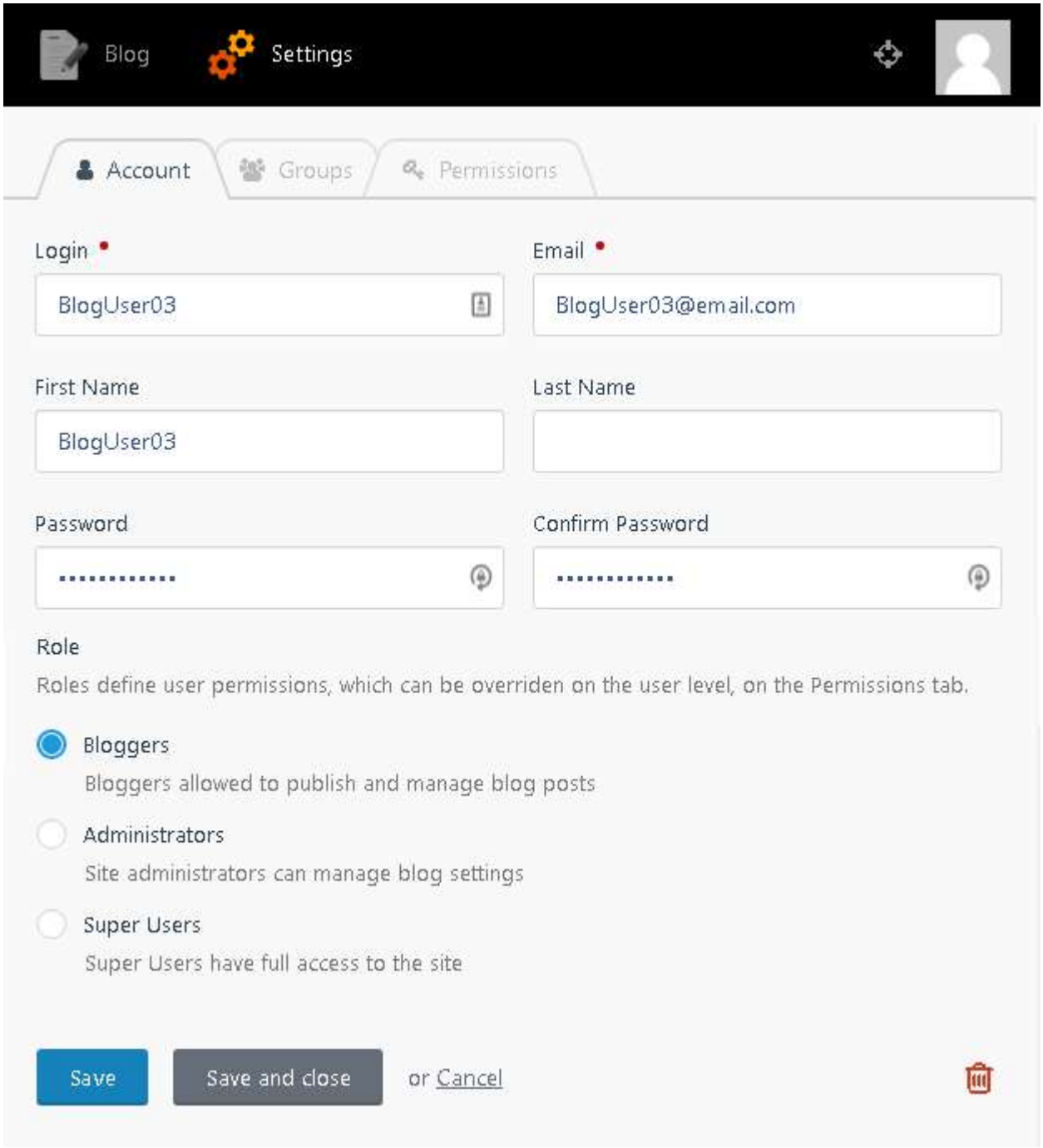
Interface: Account Management – Modify a User Account

Role: Administrators

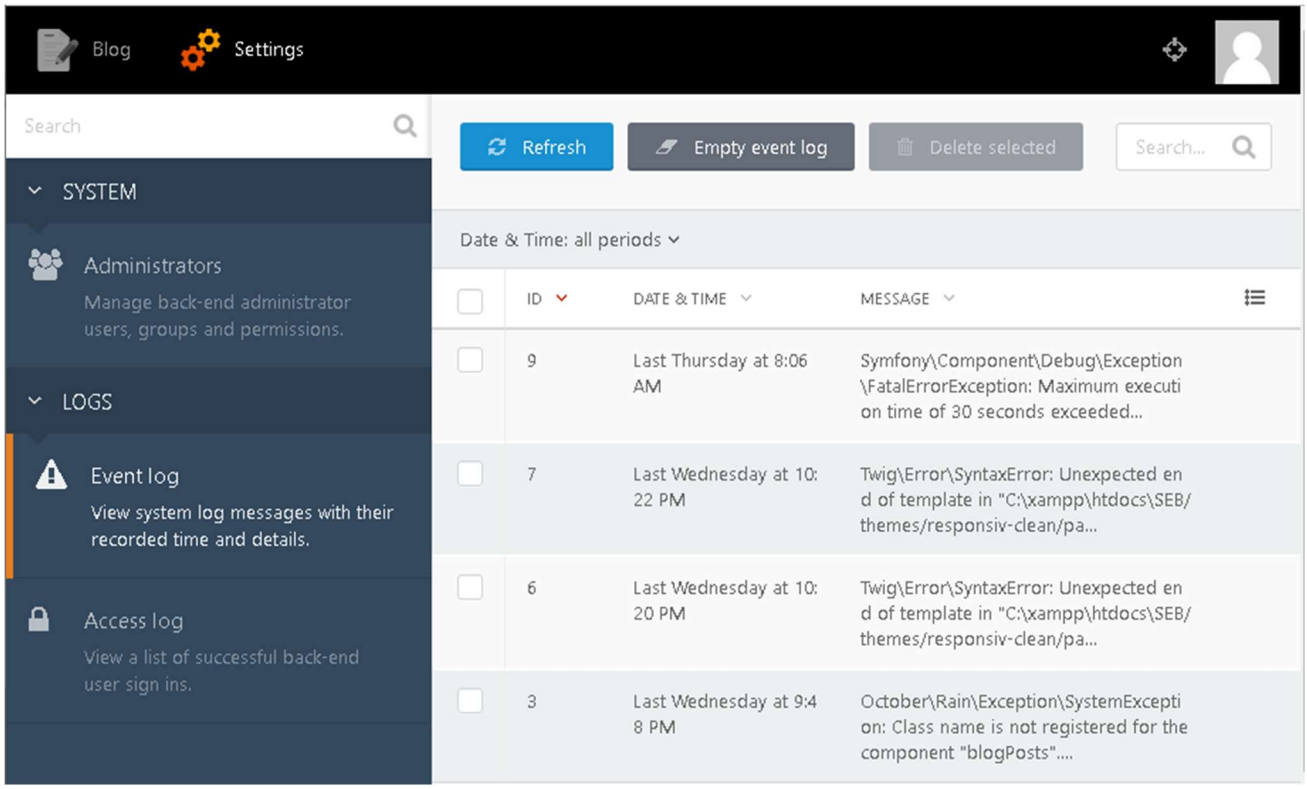


Interface: Account Management – Delete a User Account

Role: Administrators

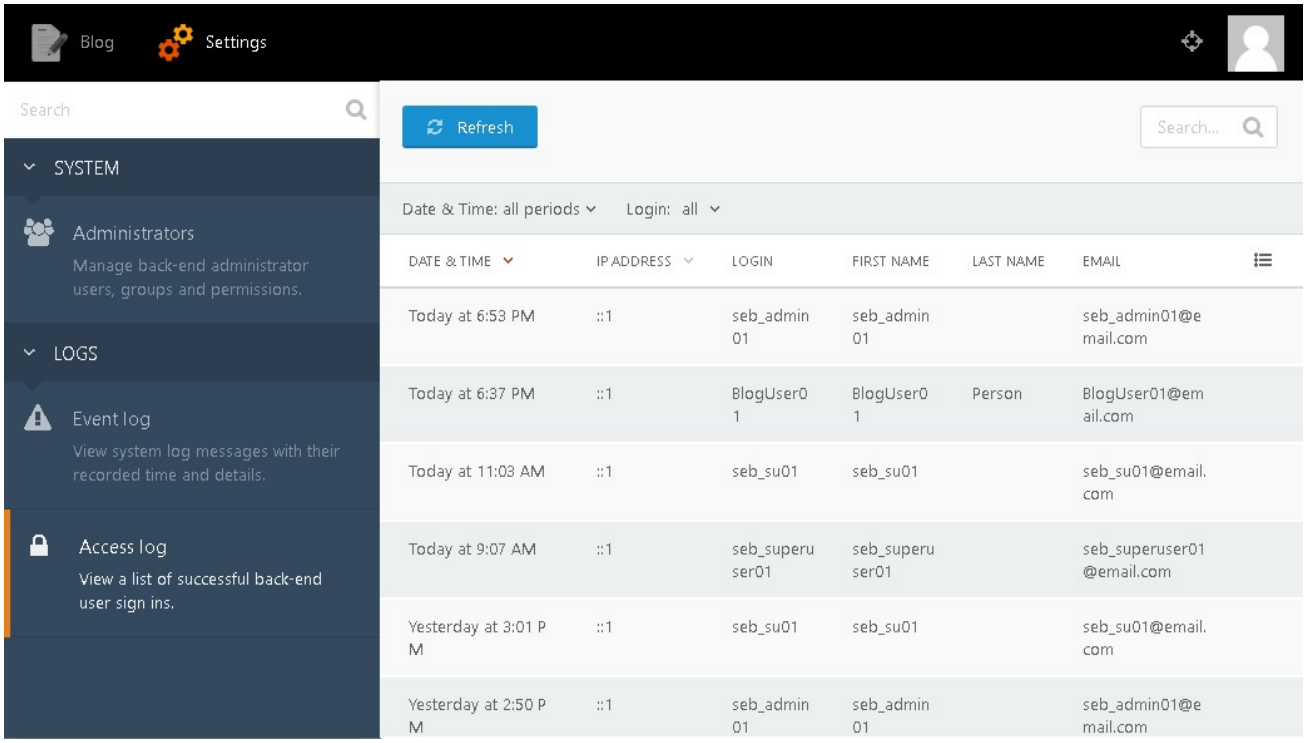


Interface: System Management – Event Log Management
Role: Administrators, Super Users

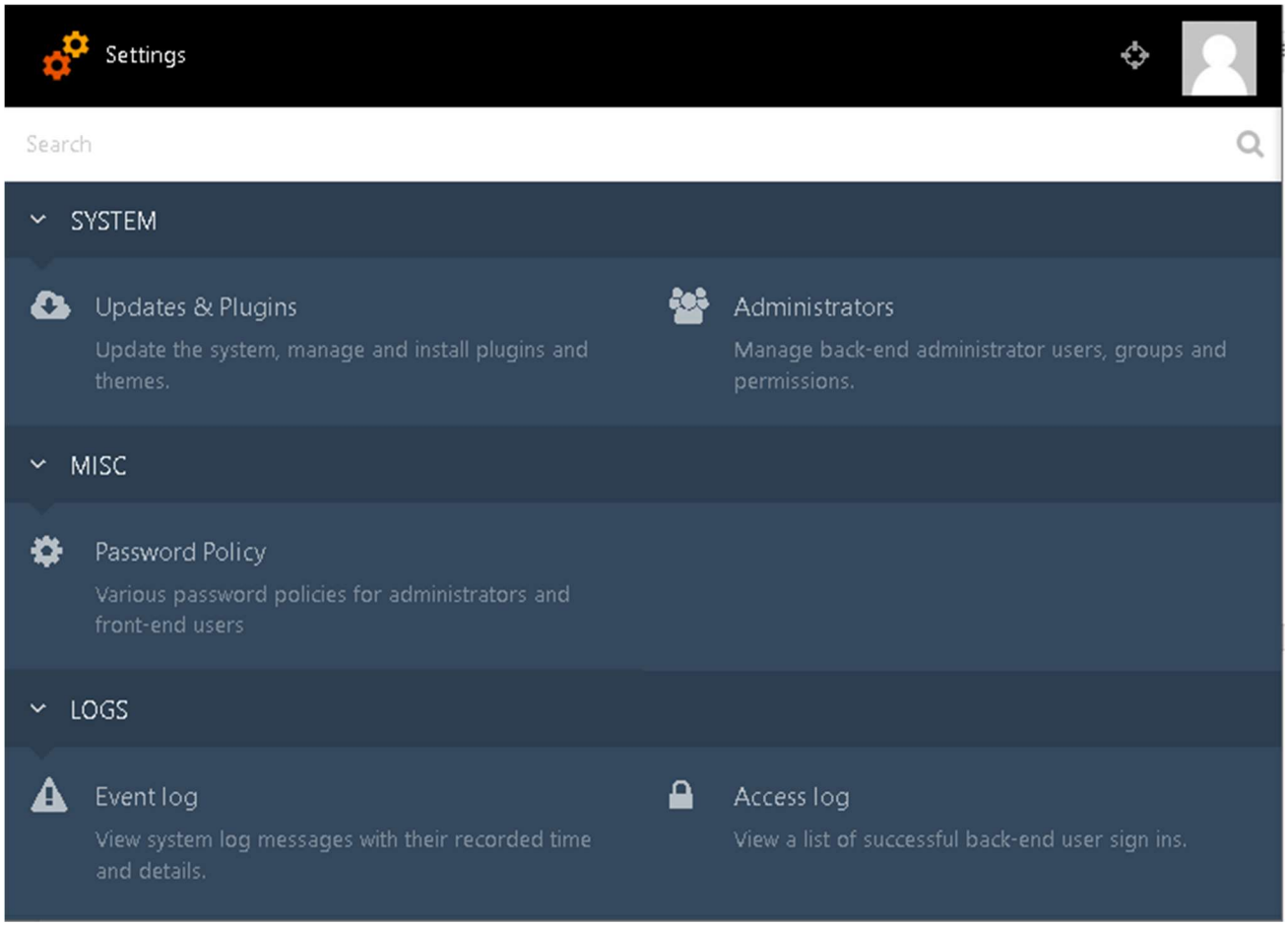


Interface: System Management – Access Log Management

Role: Administrators, Super Users

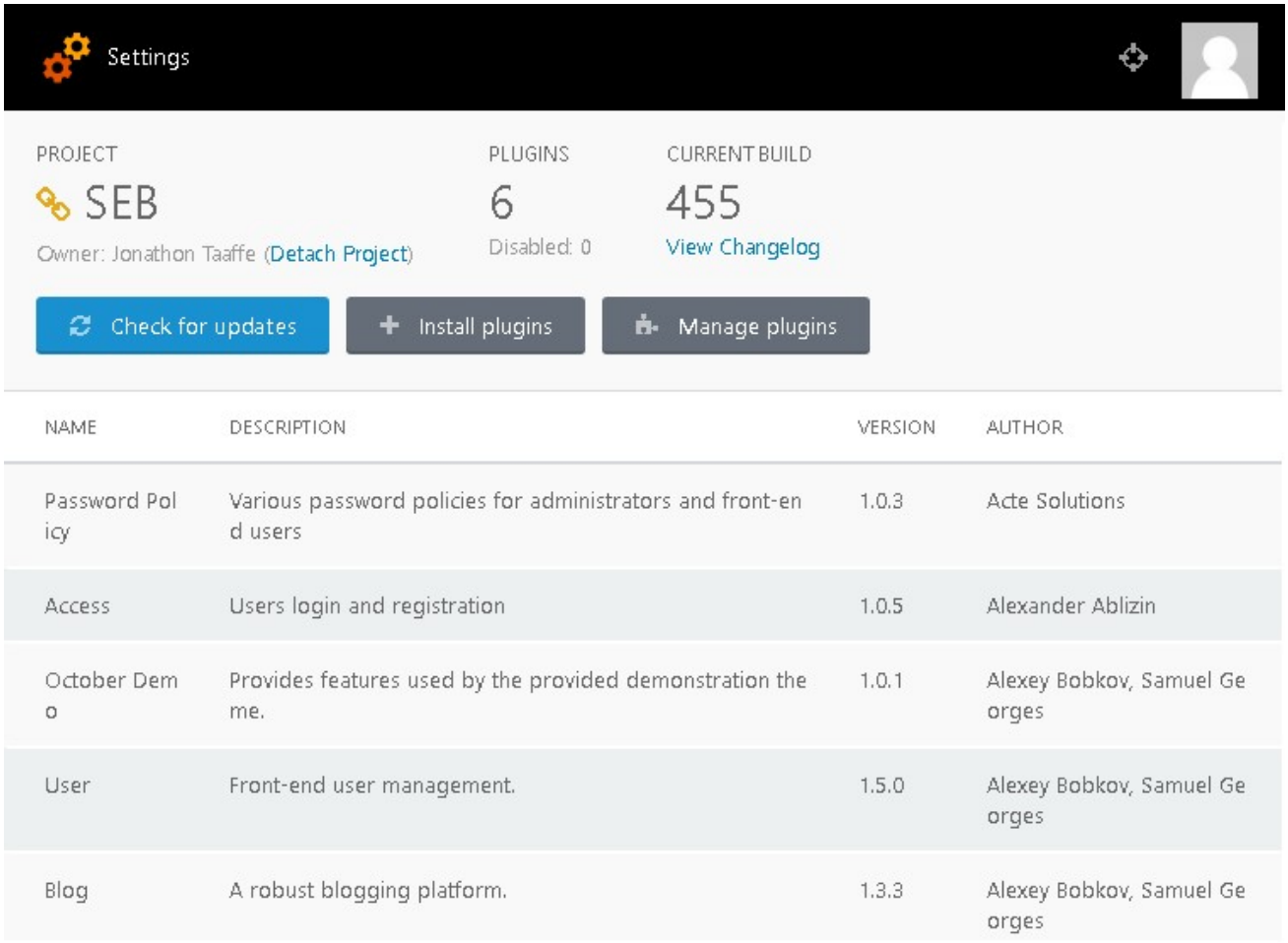


Interface: System Administration Console
Role: Super Users



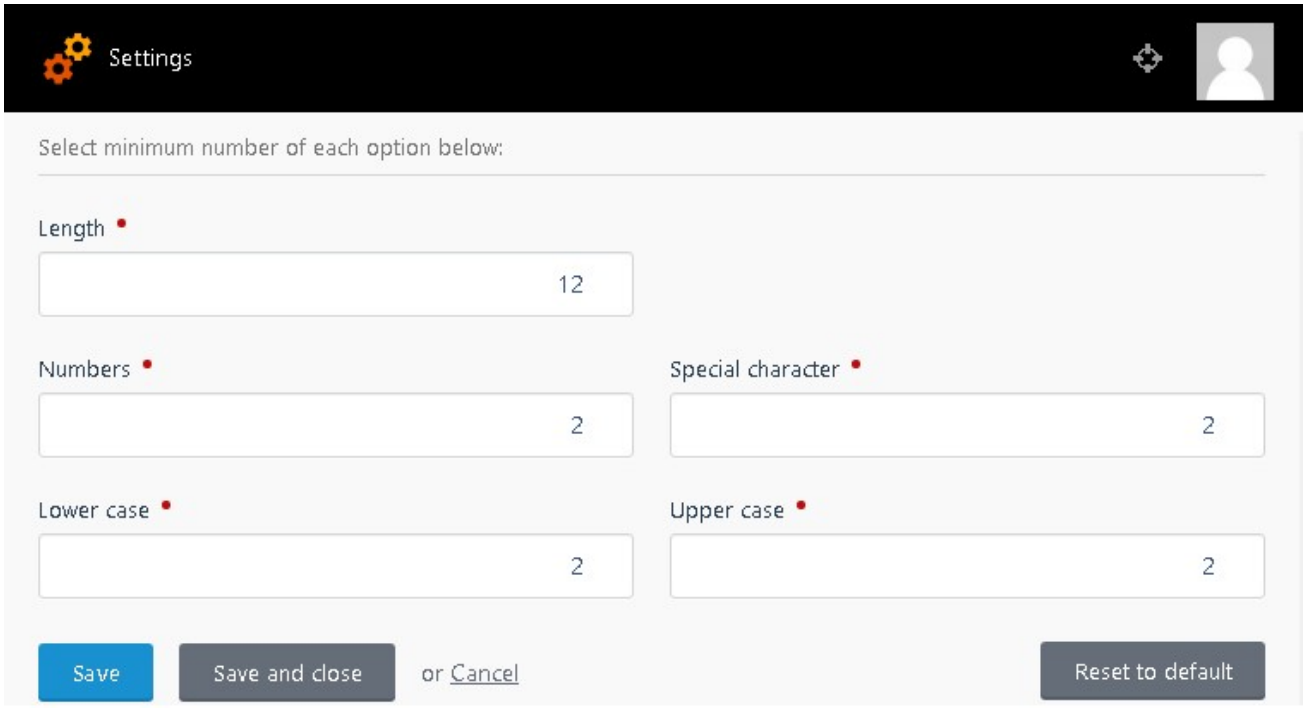
Interface: System Management – Updates & Plugins Management

Role: Super Users

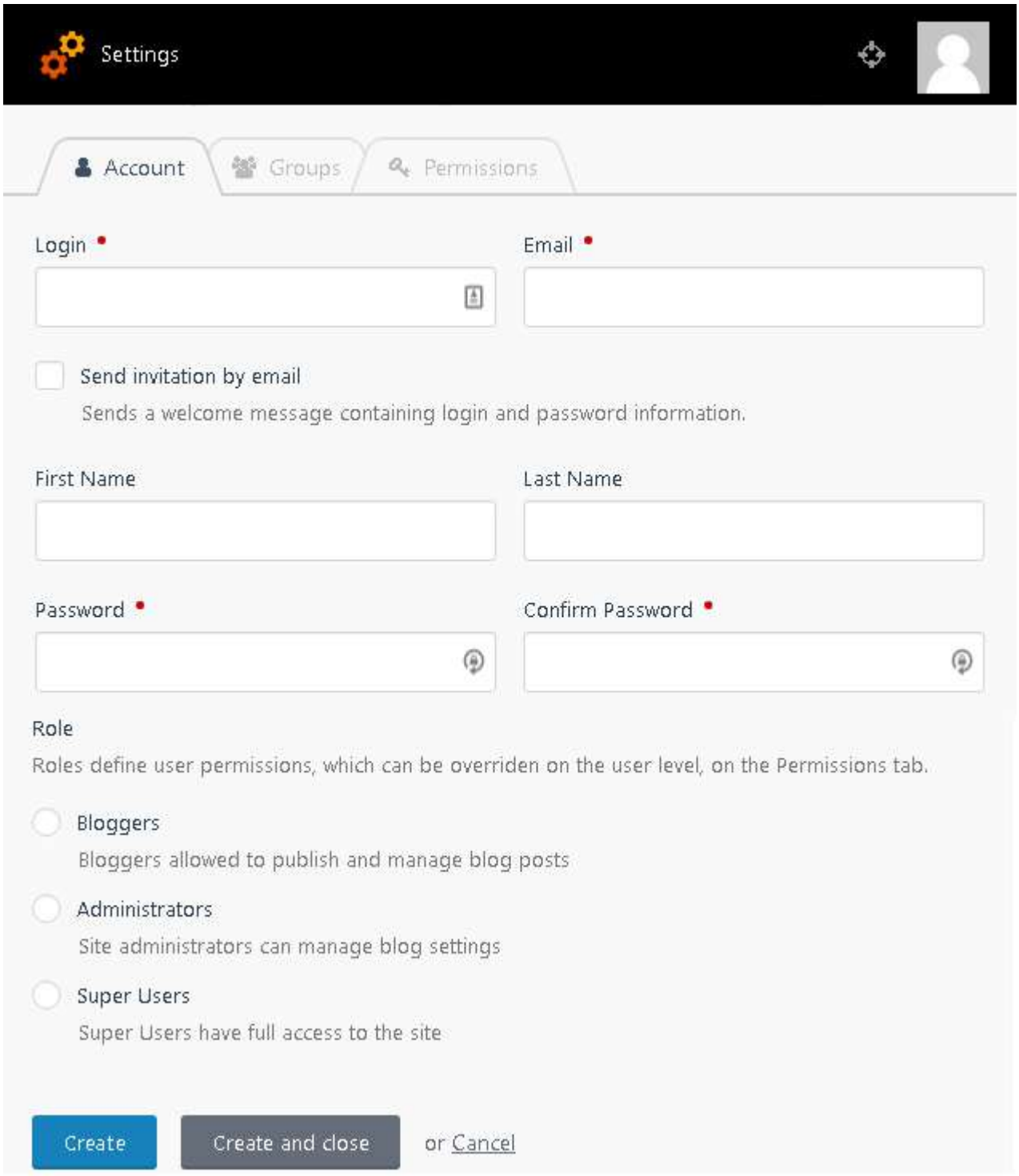


Interface: System Management – Password Policy Management

Role: Super Users

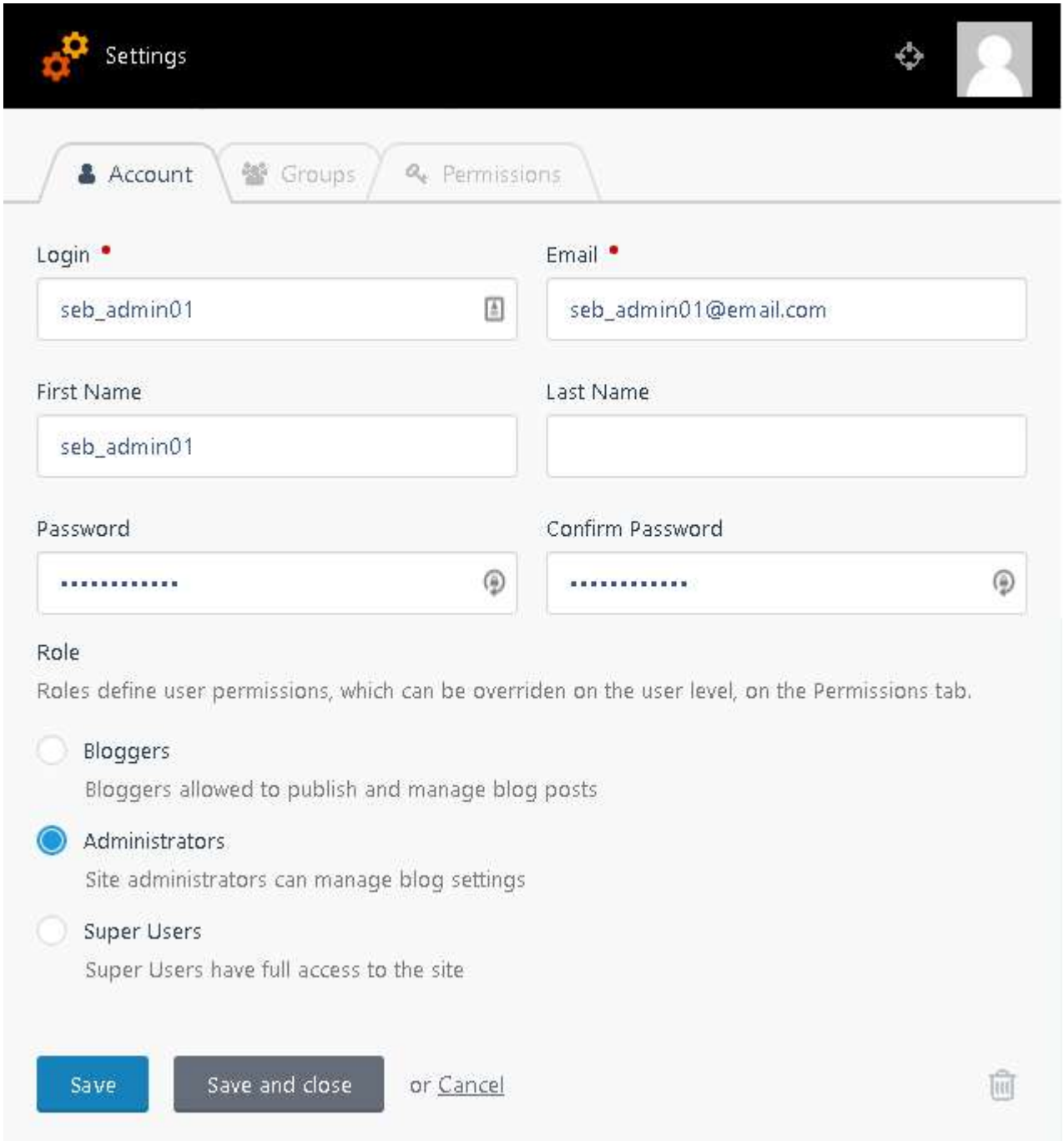


Interface: Account Management
Action: Create a New Administrator Account
Role: Super Users



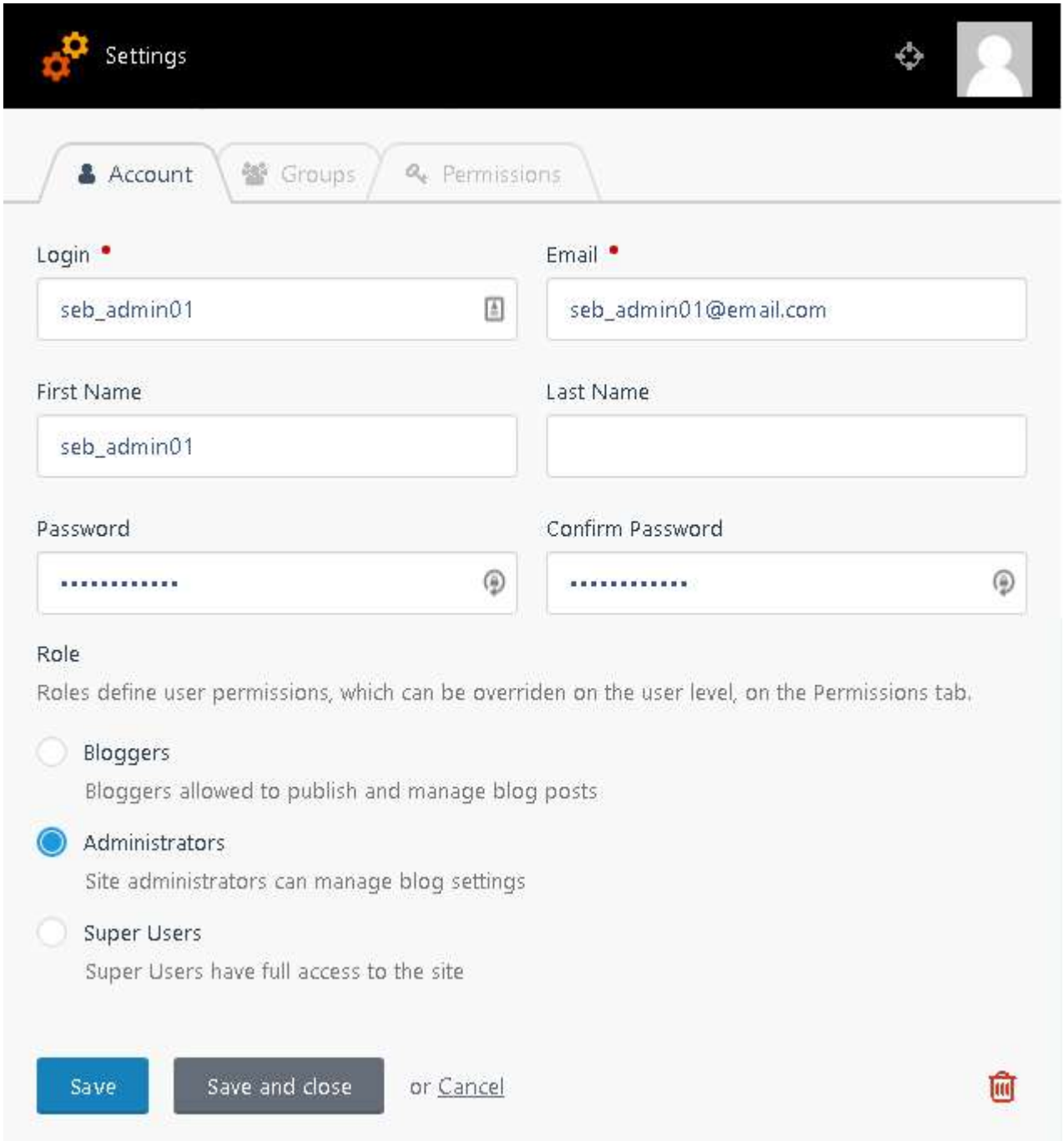
Interface: Account Management – Modify an Administrator Account

Role: Super Users



Interface: Account Management – Delete an Administrator Account

Role: Super Users



Conclusion

This project has delivered a fully functional, feature rich Secure Enterprise Blog Application.

The security components of this application ensure Confidentiality, Integrity and Availability.

The User Interface is intuitive, and the core features of Blog Content Management, Account Management and System Management have all been successfully tested.

The application is accessible from any web browser globally, but a mobile version of the application will be developed and released at a future date.

Additional features that will also be added to this application in a future release include File Management, Email Notifications and Per User Throttling.

References

1	Wikipedia.org	[2019]	Role-Based Access Control	https://en.wikipedia.org/wiki/Role-based_access_control	[Accessed 1st August 2019]
2	ApacheFriends.org	[2019]	XAMPP Download	https://www.apachefriends.org/download.html	[Accessed 1st August 2019]
3	Laravel.com	[2019]	Laravel.com	https://laravel.com	[Accessed 1st August 2019]
4	Taaffe, Jonathon	[2019]	Table 1. Development Environment Configuration		[Created 1st August 2019]
5	Taaffe, Jonathon	[2019]	Table 2. Development Database Configuration		[Created 1st August 2019]
6	Symantec Verisign	[2019]	SSL/TLS Certificates	https://www.websecurity.symantec.com/ssl-certificate	[Accessed 1st August 2019]
7	Cisco	[2019]	Cisco ASA Next Generation Firewalls	https://www.cisco.com/c/en/us/products/security/asa-firepower-services/index.html	[Accessed 1st August 2019]
8	Palo Alto Networks	[2019]	Palo Alto Networks Next-Gen Firewalls	https://www.paloaltonetworks.com/products/secure-the-network/next-generation-firewall	[Accessed 1st August 2019]
9	Cloudflare	[2019]	Cloud Web Application Firewall	https://www.cloudflare.com/waf/	[Accessed 1st August 2019]
10	Cloudflare	[2019]	Advanced DDoS Protection	https://www.cloudflare.com/ddos/	[Accessed 1st August 2019]
11	Ubuntu.com	[2019]	Download Ubuntu Server	https://ubuntu.com/download/server	[Accessed 1st August 2019]
12	Apache.org	[2019]	Apache Tomcat	https://tomcat.apache.org	[Accessed 1st August 2019]
13	Taaffe, Jonathon	[2019]	Table 3. Front-end Web Server Configuration		[Created 1st August 2019]
14	MySQL.com	[2019]	MySQL.com	https://www.mysql.com	[Accessed 1st August 2019]
15	Taaffe, Jonathon	[2019]	Table 4. Back-end Database Server Configuration		[Created 1st August 2019]
16	PHP.net	[2019]	PHP.net	https://www.php.net	[Accessed 1st August 2019]
17	Laravel.com	[2019]	Laravel.com	https://laravel.com	[Accessed 1st August 2019]
18	Taaffe, Jonathon	[2019]	Table 5. Back-end Application Server Configuration		[Created 1st August 2019]
19	Wikipedia.org	[2019]	Laravel	https://en.wikipedia.org/wiki/Laravel	[Accessed 1st August 2019]
20	Laravel.com	[2019]	Eloquent	https://laravel.com/docs/5.8/eloquent	[Accessed 1st August 2019]
21	Laravel.com	[2019]	Middleware	https://laravel.com/docs/5.8/middleware	[Accessed 1st August 2019]
22	Laravel.com	[2019]	Authentication	https://laravel.com/docs/5.8/authentication	[Accessed 1st August 2019]
23	Laravel.com	[2019]	Authentication Quickstart	https://laravel.com/docs/5.8/authentication#authentication-quickstart	[Accessed 1st August 2019]
24	Laravel.com	[2019]	Events	https://laravel.com/docs/5.8/authentication#events	[Accessed 1st August 2019]
25	Laravel.com	[2019]	Authorisation	https://laravel.com/docs/5.8/authorization	[Accessed 1st August 2019]
26	Laravel.com	[2019]	Database Considerations	https://laravel.com/docs/5.8/authentication#introduction-database-considerations	[Accessed 1st August 2019]
27	Laravel.com	[2019]	Encryption	https://laravel.com/docs/5.8/encryption	[Accessed 1st August 2019]
28	Laravel.com	[2019]	Hashing	https://laravel.com/docs/5.8/hashing	[Accessed 1st August 2019]
29	Laravel.com	[2019]	CSRF Protection	https://laravel.com/docs/5.8/csrf	[Accessed 1st August 2019]
30	Wikipedia.org	[2019]	Model-View-Controller	https://en.wikipedia.org/wiki/Model–view–controller	[Accessed 1st August 2019]
31	W3.org	[2019]	Cascading Style Sheets	https://www.w3.org/Style/CSS/Overview.en.html	[Accessed 1st August 2019]
32	Wikipedia.org	[2019]	HTML	https://en.wikipedia.org/wiki/HTML	[Accessed 1st August 2019]
33	PHP.net	[2019]	What is PHP?	https://www.php.net/manual/en/intro-whatis.php	[Accessed 1st August 2019]
34	Wikipedia.org	[2019]	JavaScript	https://en.wikipedia.org/wiki/JavaScript	[Accessed 1st August 2019]
35	W3Schools.com	[2019]	JSON – Introduction	https://www.w3schools.com/js/js_json_intro.asp	[Accessed 1st August 2019]
36	Symfony.com	[2019]	Twig	https://twig.symfony.com	[Accessed 1st August 2019]
37	Wikipedia.org	[2019]	AJAX Programming	https://en.wikipedia.org/wiki/Ajax_(programming)	[Accessed 1st August 2019]
38	Google.com	[2019]	Chrome	https://www.google.com/chrome/	[Accessed 1st August 2019]
39	Wikipedia.org	[2019]	Advance Encryption Standard	https://en.wikipedia.org/wiki/Advanced_Encryption_Standard	[Accessed 1st August 2019]
40	Wikipedia.org	[2019]	Bcrypt	https://en.wikipedia.org/wiki/Bcrypt	[Accessed 1st August 2019]

41	Taaffe, Jonathon	[2019]	Diagram 1. Network and Server Infrastructure Architecture		[Created 1st August 2019]
42	Taaffe, Jonathon	[2019]	Diagram 2. Authentication Process Flow		[Created 1st August 2019]
43	Taaffe, Jonathon	[2019]	Diagram 3. Blogger Role Use Case		[Created 1st August 2019]
44	Taaffe, Jonathon	[2019]	Diagram 4. Administrator Role Use Case		[Created 1st August 2019]
45	Taaffe, Jonathon	[2019]	Diagram 5. Super Users Role Use Case		[Created 1st August 2018]
46	Taaffe, Jonathon	[2019]	Table 6. Password Policy		[Created 1st August 2019]
47	Taaffe, Jonathon	[2019]	Table 7. Database Tables		[Created 1st August 2019]
48	Taaffe, Jonathon	[2019]	Table 8. MySQL Database Administrator Account Configuration		[Created 1st August 2019]
49	Symantec Verisign	[2019]	SSL/TLS Certificates	https://www.websecurity.symantec.com/ssl-certificate	[Accessed 1st August 2019]
50	Wikipedia.com	[2019]	HTTP Strict Transport Security	https://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security	[Accessed 1st August 2019]
51	LinuxTogether.org	[2019]	How to Setup http Strict Transport Security (HSTS) for Apache	https://linuxtogether.org/how-to-setup-hsts-for-apache/	[Accessed 1st August 2019]
52	GlobalSign.com	[2019]	What Is HSTS and How Do I Implement It?	https://www.globalsign.com/en/blog/what-is-hsts-and-how-do-i-use-it/	[Accessed 1st August 2019]
53	W3Schools.com	[2019]	PHP 5 Form Validation	https://www.w3schools.com/php/php_form_validation.asp	[Accessed 1st August 2019]
54	Taaffe, Jonathon	[2019]	Table 9. Session Management Configuration		[Created 1st August 2019]
55	Taaffe, Jonathon	[2019]	Table 10. Logging Configuration		[Created 1st August 2019]
56	Taaffe, Jonathon	[2019]	Table 11. Application Features Testing		[Created 1st August 2019]
57	Taaffe, Jonathon	[2019]	Table 12. Application Functional Test Tables		[Created 1st August 2019]