# Programming Reference

## Controllers

- AJAX
- Users
- Reservations
- Timeoff
- Settings

## Models

- User
- Reservation
- Timeoff
- Setting
- Roles

## PDF Generation

PDFs are generated using dompdf.

#### Modifying Report Layouts

The views for each report can be found in the `pdf` folder in the respective Template directories. For example, the view file for the Reservation report is found at `Template/Reservations/pdf/report.ctp`.

## Time Off API

The `api` action in the Time Off Controller exposes an API for retrieving JSON-formatted Time Off requests. The Time Off calendar uses this API to retrieve requests for loading into the calendar.

#### Routing

Because the API is served by the `api` function of `TimeoffController`, you may think that the API is accessed at `/timeoff/api`. Instead, the API is accessed via `/api/`. You can see the route configuration in `config/routes.php`:

```
Router::scope('/api/', function($routes) {
    $routes->extensions(['json']);
    $routes->connect('/timeoff/*', ['controller' => 'Timeoff', 'action' => 'api',
'_ext' => 'json']);
});
```

#### URLs

`api/timeoff/approved`

`api/timeoff/pending`

Display all approved requests for user with `id` of 2

`api/timeoff/approved/2`

Display all pending requests for user with `id` of 2

`api/timeoff/pending/2`

If a non-scheduler user accesses any of these URLs, they will see only their own requests, regardless of the `id` specified in the URL. For example, an `Hourly` employee with an `id` of 4 will see the same thing whether they vist `api/timeoff/pending`, `api/timeoff/pending/4`, or even `api/timeoff/pending/999`

#### View

`Template/Timeoff/json/index.ctp`

## Time Off Calendar

The Time Off calendar is used for displaying and making Time Off requests. It is a View Element which can be inserted into any CakePHP view.

#### Library Documentation

FullCalendar

#### Using the Calendar

To insert a blank calendar, add the following line to your view

```
<?= $this->element('Calendar/base') ?>
```

Event data sources can be added to the calendar through the `options` array.

```
<?= $this->element('Calendar/base', [
    'sources' => ['approved', 'pending'],
]) ?>
```

Creates a calendar that shows all approved and pending requests for the currently logged in user.

Modals can also be added to the calendar through the `options` array.

```
<?= $this->element('Calendar/base', [
    'sources' => ['approved', 'pending'],
    'modal' => 'Calendar/modal_approve'
]) ?>
```

#### Modifying the Calendar

Base calendar: `Element/Calendar/base.ctp`

Approval Modal - `Element/Calendar/modal_approve.ctp` - The modal that is displayed when schedulers click on an event

Request Modal - `Element/Calendar/modal_request.ctp` - Displayed when users create a new Time Off request option - allows them to specify the times of the request

## Pagination

Pagination is handled by CakePHP's PaginationHelper.

#### View

The layout file for the pagination page links is `Element/pagination.ctp`.

#### Changing the default number of elements per page

The default number of elements per page is 15. To change the default value, find where the paginator is configured in the controller and change the value.

In the case of the UsersController, the `$paginate` field configures the paginator. The `limit` option determines the number of elements per page.

```
public $paginate = [
    'limit' => 15,
    'order' => [
        'Users.email' => 'asc'
    ],
    'contain' => ['Roles'],
    'sortWhitelist' => [
        'email', 'clientID', 'Roles.name', 'expiration_date'
    ]
];
```

## CAT Certification Expiration

The User model (`Model/Entity/User.php`) contains the `checkCATCertification` function for checking the status of the associated user's CAT certification. The function takes two parameters: a comparison operator - either `'<'` or `'<='` and a boolean indicating whether or not the system should notify the user via email if the user's certification is expiring soon. We chose to put the CAT

certification logic in the model rather than the controller because it eliminated code duplication while allowing the logic to be accessible from within the CakePHP shell script.

### Daily Checks

A CakePHP shell script runs every 24 hours to check for users whose CAT certifications are expiring in exactly 30, 60, or 90 days. If a user's certification is expiring on such an interval, the script will notify the user via email.

```
$user->checkCATCertification('<', true);
```

### Check after editing Client ID

Because the daily job only checks to see if there are any users whose CAT certifications are expiring in *exactly* 30, 60, or 90 days, there is a possibility that we may miss a new user whose certification expires in less than 30 days. To resolve this, when a user is edited, the system checks to see if the user's certification expires in *less than or equal to* 30, 60, or 90 days.

```
$user->checkCATCertification('<=', true);
```

## Settings

### Adding or Modifying Settings

All settings and their possible values are held in the `$setting_info` variable within the `index` function of `SettingsController.php`. `$setting_info` is formatted as follows:

```
$setting_info = [
    'setting_name' => [
        'description' => 'Short description what the setting does',
        'options' => ['val1' => 'val1', 'val2' => 'val2', ...]
    ],
    ...
];
```

Modifying the possible values for a setting is as simple as editing the values in the `options` array for that setting.

To add a setting, simply add another setting to the `$setting_info` array. When you add a setting, you must be sure to insert the new setting into the `Settings` table.

For example, let's imagine we want to add a setting for the maximum number of email/password attempts allowed by the system. First we would add the setting to the `$setting_info` array.

```
$setting_info = [
    ...
    'password_attempts' => [
        'description' => 'Maximum number of email/password attempts allowed',
```

```
        'options' => ['5' => '5', '10' => '10', 'unlimited' => 'unlimited']
    ]
];
```

Next, add the new setting to the database

```
INSERT INTO Settings (name, value) VALUES ('password_attempts', '5');
```

# Metrics

System metrics are generated by the `metrics` action in the `UsersController`.

### Graphs

StarPort uses Chart.js to display metrics graphically. To pass metrics data to Chart.js, the `metrics` function JSON-encodes the metrics and inserts them into the DOM as a `data-points` attribute for each tab.

At run time, the JSON-encoded metrics are pulled out of the DOM and loaded into Chart.js

```
var data = JSON.parse(tab.attr("data-points"));
```

### AJAX

When metrics filters are applied, the client makes an AJAX request to the `AjaxController`. The `AjaxController` generates an HTML response and sends it to the client. The client then inserts the response into the DOM