User Manual

Organisation: `https://github.com/Hyperperform`

Developers:
Rohan Chhipa *14188377*
Avinash Singh *14043778*
Jason Gordon *14405025*
Claudio Da Silva *14205892*

Updated October 23, 2016

Client

MagnaBC
`http://www.magnabc.co.za/`

Contents

1 System Overview **3**

2 System Configuration **3**
2.1 Docker installation . . . 3
2.2 Manual Installation . . . 3
2.3 Event Gathering . . . 4
2.4 Miscellaneous . . . 5

3 Installation **5**
3.1 Docker Installation . . . 5
3.2 Manual Installation . . . 5
3.2.1 WildFly . . . 5
3.2.2 PostgreSQL . . . 6
3.2.3 ActiveMQ . . . 9
3.3 Notifications . . . 10
3.3.1 Deploying to WildFly . . . 12
3.3.2 Front-end Dashboard . . . 12

4 Getting Started/Using the System **14**

1 System Overview

Many different tools are available for measuring the quality of products made, but very few tools exist which assess the quality of the people making said products. People play a huge role in a project, and trying to monitor each and every one becomes a tedious task which diverts man power away from other more critical tasks. Whether it be for an end of year evaluation, or attempting to assess the current status of a project, generating a report on a staff member can help keep up productivity, as well as get them any help they need in order to resume quality performance. By ensuring that there is constant quality performance from each individual on a project, one can increase project quality as well as reduce project risks such as loss of an important team member during a critical stage of a project's life-cycle.

The HyperPerform system has the ability to automatically gather information from multiple integrations such as GitHub and Travis. This alleviates the responsibility of the manager and HR having to manually monitor each employee by filling in time sheets and performance documents. The system aims to simplify management by allowing a manager to view all relevant employee data in a simple and easy to understand manner. This data is represented in the form of a report. These reports can be a summarised report or a more detailed report on an employees' activities.
In order to allow the system to be used in many different environments, we have developed the system to be highly pluggable in nature, so that one could easily increase the number of integrations from which information can be pulled.

2 System Configuration

This guide has been created for users who are using a Linux based operating system. To install the HyperPerform system on the machine you will be required to have an active connection to the internet. Please note that high amounts of data might be consumed during this process.

2.1 Docker installation

To install the system with ease and to avoid all the system configurations you can download Docker to handle this for you. Docker can be found at `www.docker.com` where guides are made available for installing docker on a particular operating system. If you intend to use Docker to install the HyperPerform system then please ensure you install Docker on your machine.

2.2 Manual Installation

The manual installation requires you to download the source code from the GitHub repository. The newest release is highly recommended. To carry out a manual installation please ensure you have Maven and the WildFly application server on your machine.

Maven can be downloaded from: `maven.apache.org`

WildFly can be downloaded from: `wildfly.org`

Please ensure you download WildFly 10. The HyperPerform system was fully tested on this version of WildFly. Any other version might produce unexpected behaviour.

For the front-end Dashboard please ensure you have Nodejs (version 6.4.0 or higher) installed on your machine. Nodejs can be found at `https://nodejs.org/en/`.

2.3 Event Gathering

The system gathers information through webhook technology. Thus to be able to receive any events the computer on which the system will be installed **must** be connected to the internet. When configuring the integrations you will need to provide the URL for that integration to send events to. The following figure shows the GitHub webhook.

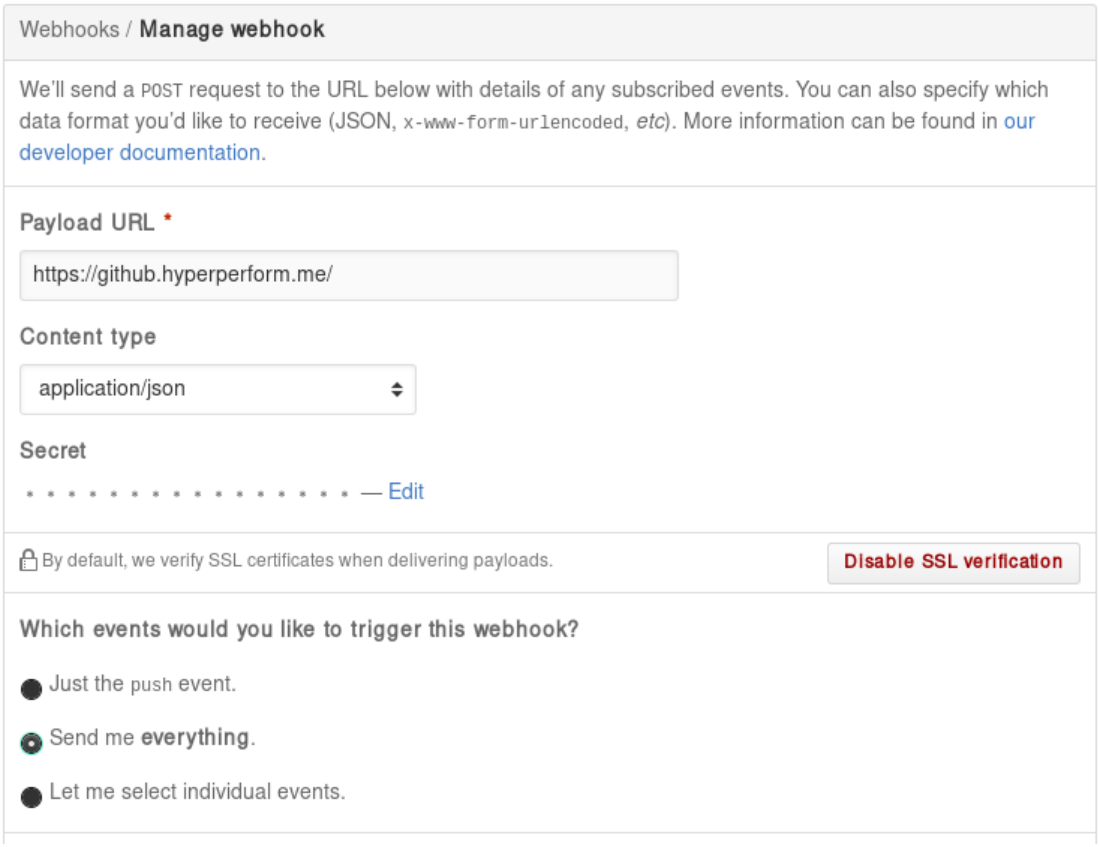

Figure 1: Adding GitHub webhook

An optional feature would be to bind a domain name to the global IP address of the server. However this is merely for readability purposes and work affect system performance in any way.

2.4 Miscellaneous

Certain components of the system require user names which are consistent. The Git event processor is one such component. When using GitHub on your local machine please ensure that your local configurable Git name corresponds to your actual GitHub username.

To check your local Git name open terminal or command prompt and run the following command:

```
git config user.name
```

If the name corresponds to your account name then nothing further needs to be done and you can continue to the installation. However if the two names do not match then run the following command:

```
git config --global user.name "<username>"
```

In the command above the <username> is your actual account name.

3 Installation

3.1 Docker Installation

Assuming you have docker installed on your machine, simply run the following command in terminal:

```
docker run hyperperform/HyperPerform
```

This will download the HyperPerform Docker image from DockerHub and run it on you machine.

The front end component does not have a Docker image at this point in time. To install the front end component please refer to section 3.2.5 for the manual installation.

3.2 Manual Installation

This installation guide assumes a Linux Server running Ubuntu 14 or higher:

3.2.1 WildFly

Once you have downloaded the WildFly application server please carry out the WildFly installations and add a user. Once this is done proceed to installing PostgreSQL.

3.2.2 PostgreSQL

The install PostgreSQL on your machine:

Install via terminal:

```
sudo apt-get update
sudo apt-get install postgresql postgresql-contrib
```

To configure PostgreSQL to connect remotely:

```
sudo nano /etc/postgresql/9.3/main/postgresql.conf
```

Edit the following lines:

```
listen_addresses = "*"
```

Create database hyperperform and the tables
Run the following commands in terminal:

```
psql -c 'CREATE DATABASE hyperperform;' -U postgres

psql -d hyperperform -c 'CREATE TABLE public."GitPush" ( id
    integer NOT NULL, repository character varying(255), "
    timestamp" timestamp without time zone, username character
    varying(255), commitsize integer, url character varying(255)
    , message character varying(255), CONSTRAINT "GitPush_pkey"
    PRIMARY KEY (id) ); CREATE SEQUENCE public.
    hibernate_sequence INCREMENT 1 MINVALUE 1 MAXVALUE
    9223372036854775807 START 1 CACHE 1;' -U postgres

psql -d hyperperform -c 'CREATE TABLE public."TravisEvent" ( id
     integer NOT NULL, branch character varying(255), commiter
    character varying(255), repo character varying(255), status
    character varying(255), "timestamp" timestamp without time
    zone, CONSTRAINT "TravisEvent_pkey" PRIMARY KEY (id));' -U
    postgres

psql -d hyperperform -c 'CREATE TABLE public."CalendarProject"
    ( projectid integer NOT NULL, calendarid character varying
    (255), collaborators bytea, creator character varying(255),
    duedate timestamp without time zone, eventid character
    varying(255), reponame character varying(255), "timestamp"
    timestamp without time zone, CONSTRAINT "
    CalendarProject_pkey" PRIMARY KEY (projectid));' -U postgres

psql -d hyperperform -c 'CREATE TABLE public."CalendarMeeting"
    ( meetingid integer NOT NULL, calendarid character varying
    (255), creator character varying(255), duedate timestamp
    without time zone, eventid character varying(255), location
    character varying(255), "timestamp" timestamp without time
    zone, CONSTRAINT "CalendarMeeting_pkey" PRIMARY KEY (
    meetingid));' -U postgres
```

```
psql -d hyperperform -c 'CREATE TABLE public."
    CalendarMeeting_attendees" ( "CalendarMeeting_meetingID"
    integer NOT NULL, attendees integer, attendees_key character
     varying(255) NOT NULL, CONSTRAINT "
    CalendarMeeting_attendees_pkey" PRIMARY KEY ("
    CalendarMeeting_meetingID", attendees_key), CONSTRAINT
    fkn4q1pmj9vx3tfsaw9irp9voax FOREIGN KEY ("
    CalendarMeeting_meetingID") REFERENCES public."
    CalendarMeeting" (meetingid) MATCH SIMPLE ON UPDATE NO
    ACTION ON DELETE NO ACTION);' -U postgres

psql -d hyperperform -c 'CREATE TABLE public."GitIssue"(id
    integer NOT NULL, action character varying(255), assignee
    character varying(255), createdby character varying(255),
    issueid bigint, repository character varying(255), "
    timestamp" timestamp without time zone, title character
    varying(255), url character varying(255), CONSTRAINT "
    GitIssue_pkey" PRIMARY KEY (id));' -U postgres

psql -d hyperperform -c 'CREATE TABLE public."User" (email
    character varying(255) NOT NULL, gitusername character
    varying(255), name character varying(255), "position"
    integer, profilepicture bytea,  role integer, surname
    character varying(255), username character varying(255),
    password character varying(255), CONSTRAINT "User_pkey"
    PRIMARY KEY (email));' -U postgres

psql -d hyperperform -c 'CREATE TABLE "AccessEvent"(id integer
    NOT NULL, email character varying(255), day bigint, deviceid
     character varying(255), employeeid character varying(255),
    name character varying(255), surname character varying(255),
     "timestamp" timestamp without time zone, CONSTRAINT "
    AccessEvent_pkey" PRIMARY KEY (id) );' -U postgres

psql -d hyperperform -c 'CREATE TABLE public."ForecastData"(
    data character varying(10485760) NOT NULL, CONSTRAINT "
    ForecastData_pkey" PRIMARY KEY (data));' -U postgres
```

3.2.3 ActiveMQ

To setup ActiveMQ on your server:

- Start up your WildFly application server
- Navigate to WildFly management console on localhost:9990
- Navigate to configurations tab and click on sub-systems
- Scroll down and search for Messaging-ActiveMQ and click on it
- Click on default, select queues/topics

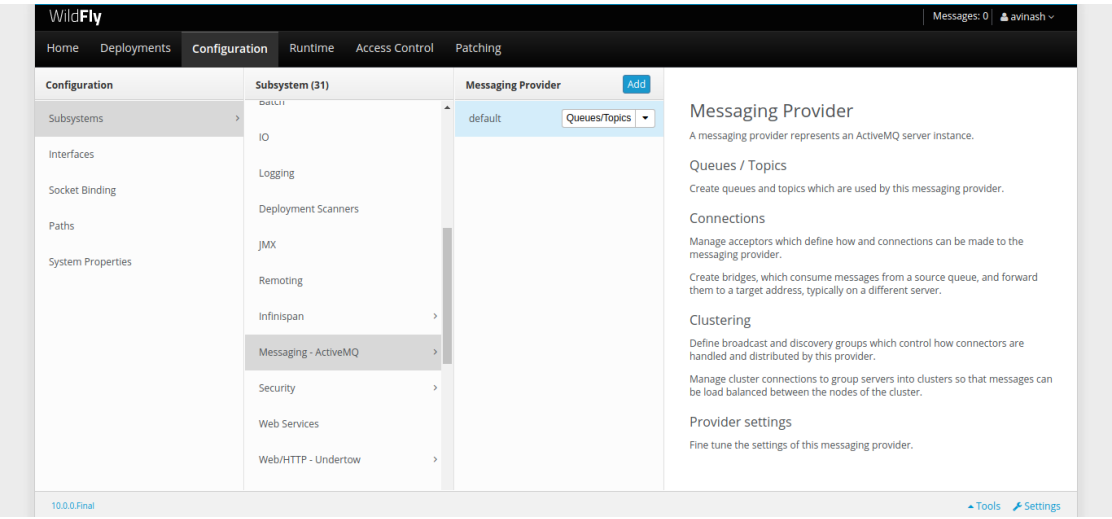

Figure 2: Subsystems Configuration

- Click add and input the following information:
  - Name*: hyperperform
  - JNDI Names*: java:/jms/queue/hyperperform

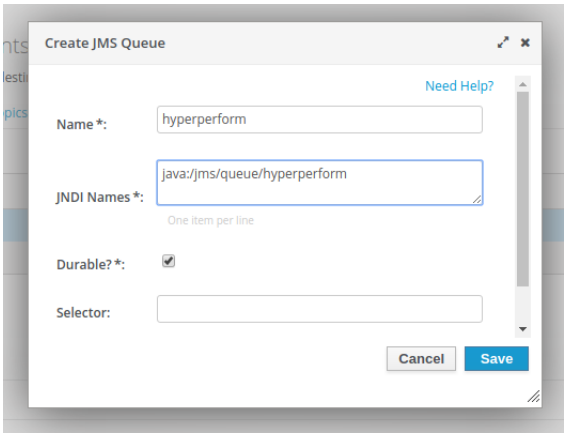

Figure 3: Queue Configuration

- Click save

3.3 Notifications

Please Note: These settings are for a Gmail configuration if you are running your own stmp server you need to adjust the information appropriately. To setup Notifications via Email on the server:

- Start up your WildFly application server
- Navigate to WildFly management console on localhost:9990
- Navigate to configurations tab and click on sub-systems
- Scroll down and search for Email and click on it

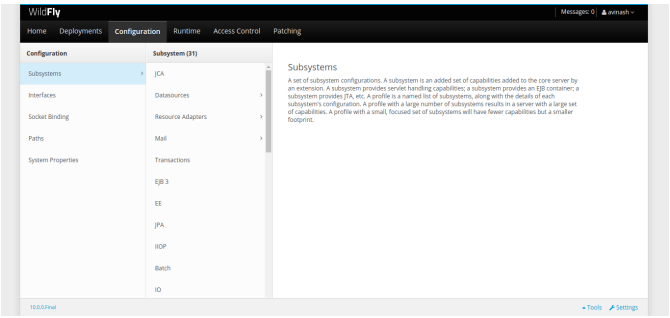

Figure 4: Subsystem Email

- Click add and input the following information:
  - Name*: Gmail
  - JNDI: java:/jboss/mail/gmail
- Click save

- There after View the new configuration and Add a new type
  - Socket-binding: mail-smtp
  - Type: smtp
  - Username: "Your@gmail account"
  - Password: "Gmail password"
  - SSL: enable
- There after reload the server
- After reloading navigate to configurations tab and the Socket Bindings

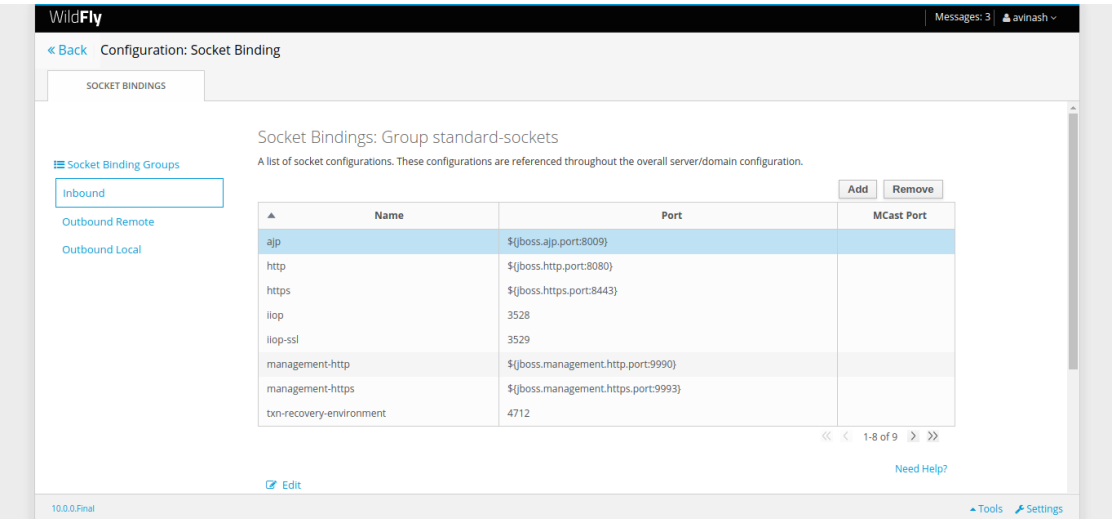

Figure 5: Socket Bindings

- Click on View and there after under standard-sockets click view
- Navigate to Outbound Remote and edit the mail-smtp socket bindings
  - Host: smpt.gmail.com

– Port: 465

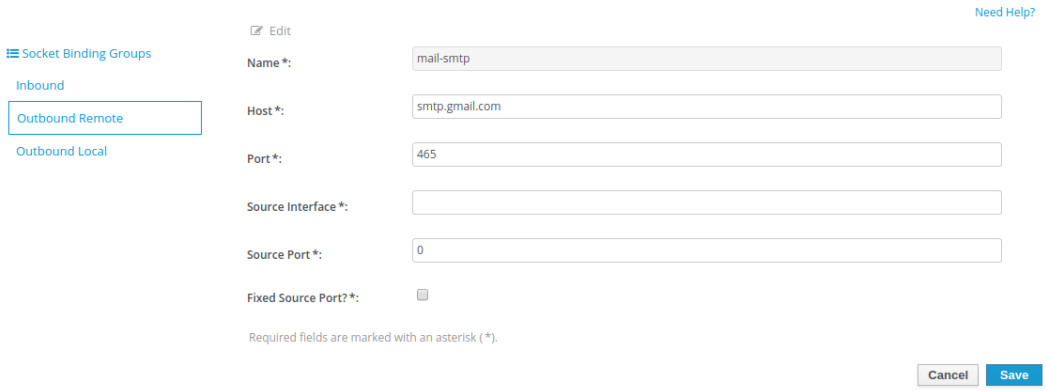

Figure 6: Outbound Socket Configuration

- Reload server

3.3.1 Deploying to WildFly

To deploy the HyperPerform system to the application you will need to build the system using from the source code.

- Ensure the WildFly server is running.
- Navigate to `https://github.com/HyperPerform/hyper-perform-server/releases` and download the newest release source code.
- Extract the source code
- Navigate to the root directory of the source code. A file named pom.xml should be clearly visible.
- Run the following command: mvn clean wildfly:deploy
- Maven will then ask you to provide your user name and password for the Wildfly Server.
- Thereafter Maven will automatically deploy the compiled code (war) to WildFly

3.3.2 Front-end Dashboard

Please notee that there is no release yet for the dashboard and there might be a few bugs, or limitations to the software.

To start up the front end please ensure you have Node 6.4.0 or higher installed on your machine. Node can be found at `https://nodejs.org/en/`.

Please make sure that these commands execute successfully before attempting to run the system:

```
npm install -g gulp
npm install -g bower
npm install -g sass
```

Once that has completed with no errors do the following.

- Download the Dashboard source code from `https://github.com/HyperPerform/hyper-perform-web-application`
- Navigate to the root directory of the source code

Run the following commands in terminal:

```
npm install
gulp build
gulp serve
```

The front-end system will auto launch in your default browser in order to view the data in the front-end system the Wildfly application server must be running.

4 Getting Started/Using the System

Once the front-end Dashboard is served your default browser should automatically open. In the event that it didn't, simply open the browser of your choice and navigate to the following URL: `localhost:3000`.

4.1 Logging In

Once the Dashboard loads you will be presented with the following screen:

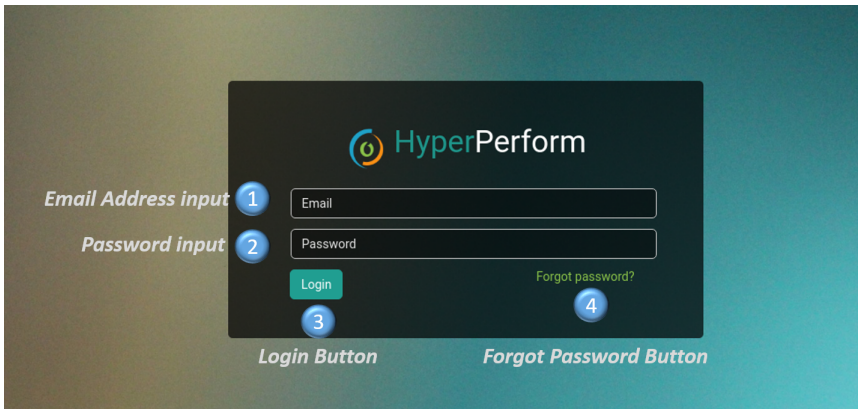

Figure 7: Login screen

1. **Email Address Input:** The input field for the email address with which an employee or manager was registered.

2. **Password Input:** The input field for the password with which an employee was registered or the password that was changed by "forgot password".

3. **Login Button:** Logs the user into the system, but first validates the input into the above mentioned fields.

4. **Forgot Password Button:** A button that should be pressed when a user has forgotten their password. This will bring up a window which asks for the users' email address. Once submitted a new password will be sent to you.

   The default email for login is *admin@hyperperform.me* and password *1234*.
   Note: This account should be removed, manually, once another admin is created using this default administrator account.

4.2 Your Dashboard

Once logged in the user will be presented with the following screen:

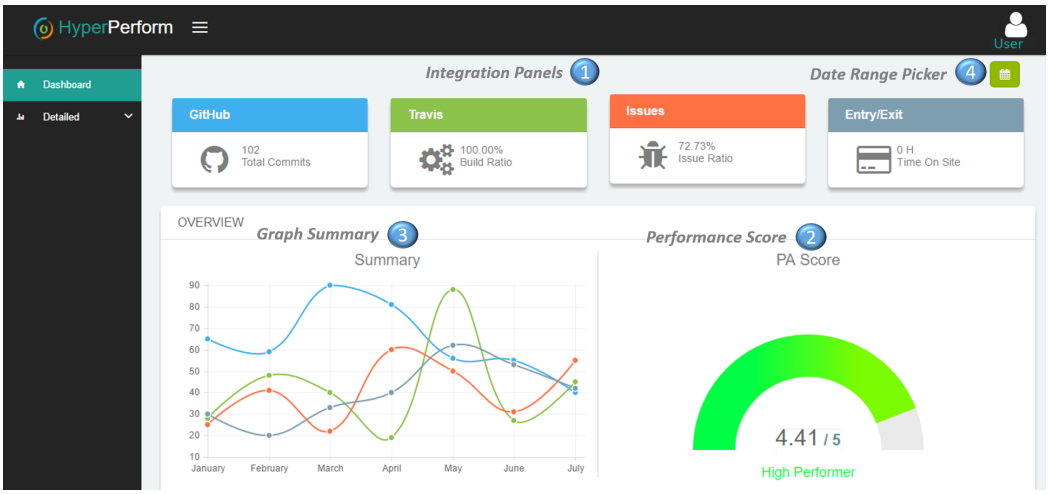

Figure 8: Dashboard

1. **Integration Panels:** Note the four colour-coded panels on top, each of these panels represents an integration. Each of the panels show your cumulative information for your specific integrations during the time period specified (discussed in point 4 below). These panels are click-able and will direct you to a details screen which will be discussed in the next subsection (subsection 4.3). This can also be done via the sidebar under the "Detailed" dropdown.

2. **Performance Score:** A view of your current performance score along with your performance indicator. The current performance indicators are:

   - Non-Performer (Score Below 2)
   - Standard Performer (Score between 2 - 3)
   - Standard-Plus Performer (Score between 3 - 4)
   - High Performer (Score between 4 - 5)

3. **Graph Summary:** A summary graph view of your performance for each colour-coded integration, during the months shown in the bottom of the graph.

4. **Date Range Picker:** The date picker where you can filter your score between two dates and times. This is discussed in the 4.2.1 below. The default time period used for the calculation of your PA Score is three months.

4.2.1 Filtering Your Performance Score

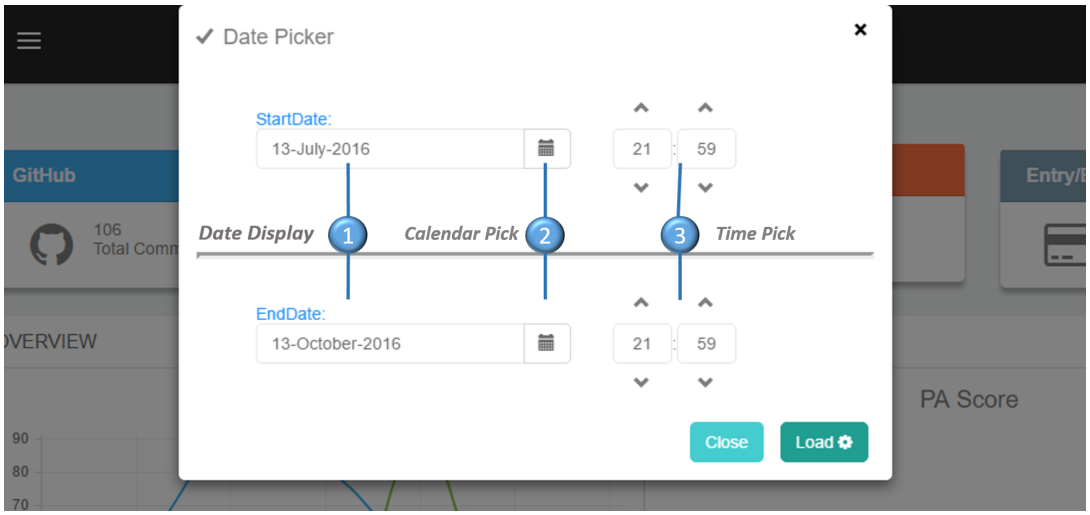

Figure 9: Date Picker

1. **Date Display:** A view of the selected dates for the start date and end date. These are directly modifiable by either typing a date of your choosing or using the calendar picker (described below).

2. **Calendar Picker:** When the button is clicked a calendar will be displayed above the modal window, allowing the selection of the date of your choice.

3. **Time Pick:** This gives you the ability to choose a specific time on the day of the date that has been chosen in the Calendar Picker.

4. **Load Button:** When this button is clicked it will take the date and time into consideration into the algorithms. Our system will then calculate the PA Score along with the values for each integration and display them to you within this time frame.

4.3 Detailed View

4.3.1 Graph View

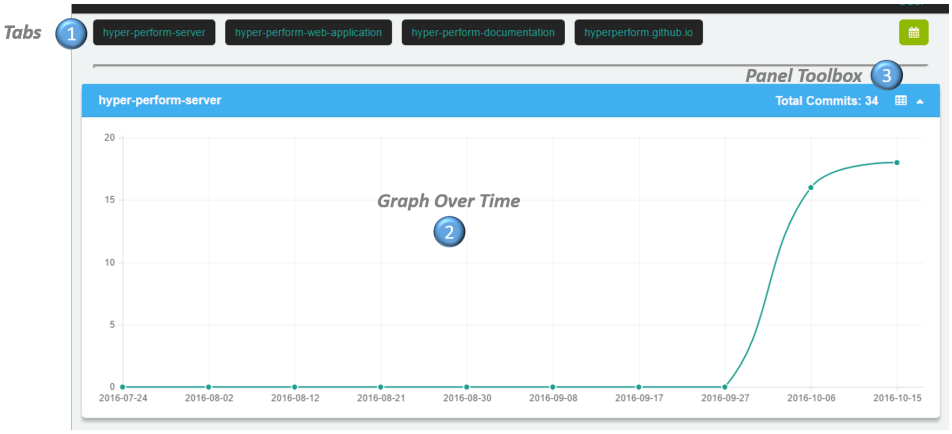

Figure 10: Detailed View Graph

1. **Tabs:** The names for each repository on which their is a webhook for our system (and the current user has contributions). Each of these tabs are clickable. When clicked it will scroll down to that panel on the page so you do not have to search for it yourself. **Note:** These tabs are not specified for integrations without different repositories (i.e. entry/exit).

2. **Graph Over Time:** A performance chart of the users performance regarding the integration information in the panel. This is shown within the timeframe specified in the date picker.

3. **Panel Toolbox:** On the left of this toolbox shows information regarding the panel. in the middle it allows you to click to change to the table view of this panel (explained below). The graph view can be toggled back again by clicking the same button. On the right allows you to minimise the panel to make it easier to see the other panels on the page (you may need to double click for this to work).

4.3.2 Table View

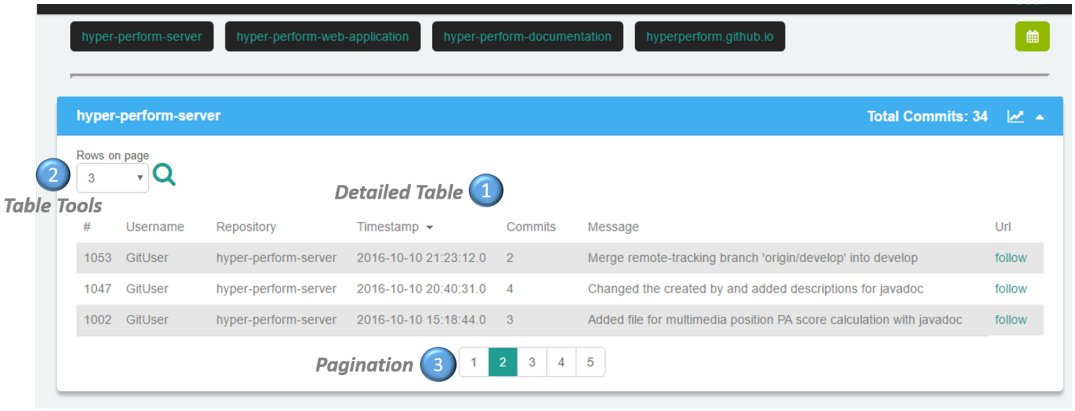

Figure 11: Detailed View Graph

1. **Detailed Table:** A table that shows the details of the panel in the integration being viewed. In this example above, each row in the table shows one push to your git repository (specified in the panel heading).

2. **Table Tools:** On the left you are able to filter out how many results you want to show in the table at a time. On the right, there is a button that, when pressed, will show search bars for each column in the table. This allows the ability to search each column individually to filter out the results required.

3. **Pagination:** The pagination shows the numbers of different pages in the table where the rest of your data will be shown.

4.4 Signing Out

If you wish to logout then you merely click on the profile icon in the top right corner. Once clicked you will be presented with a small menu.

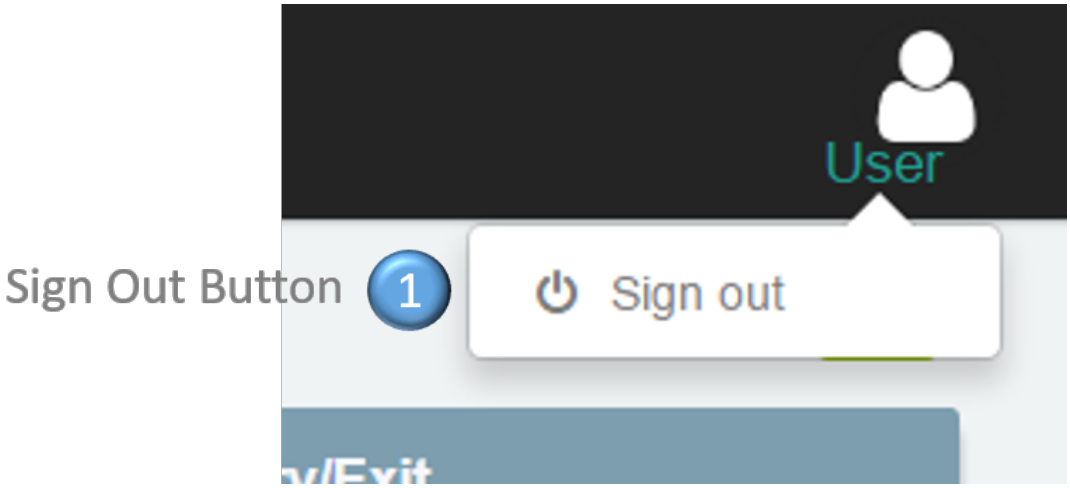

Figure 12: Signing Out

1. **Sign Out Button:**
   (a) For an employee - When this button is pressed it will sign you out of your dashboard and you will be redirected back to the login page (Figure 7).
   (b) For a manager - This button will sign you out of your employees' dashboard and redirect you to the Employee List Page (Figure 13) where you can view other employees' performance.

4.5 Managerial features

A variety of features are only made available to the manager within the organisation.

4.5.1 Managerial view

When a manager logs into the system, instead of being directed to a dashboard they will be directed to the following view.

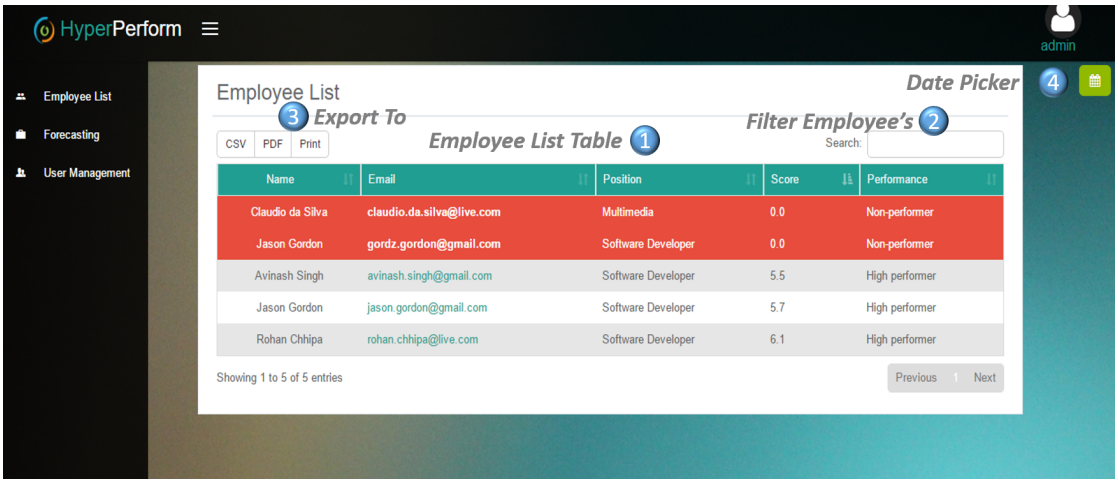

Figure 13: Managerial view

1. **Employee List Table:**
   (a) This table allows a manager to view all the current employees within the organisation. They can see all the necessary employee data such as name, email, performance score and performance classification.
   (b) A manager also has the ability to view an employees' dashboard. This can be achieved by simply clicking on the employees' name and the manager will be directed to the corresponding employees' dashboard.
2. **Filter Employee's:** This search bar allows a manager to filter the results by searching for any value in any field of the table. The resulting table will show the filtered results only.
3. **Date Picker:** This allows a manager to filter the scores between two dates and times resulting in a change in the table to those scores between that specific period.
4. **Export To:** Managers also have the ability to export the table to a PDF or CSV file or even directly print it out. This allows the manager to keep a copy with them at any time and can aid with record keeping by allowing them to print the

table at regular intervals. **Note:** After filtering the results (via point 2) only the filtered data will be exported and not the whole table.

4.5.2 User registration

The responsibility of adding users to the system is given to the manager. When registering a new user the manager must go through a three step registration process.

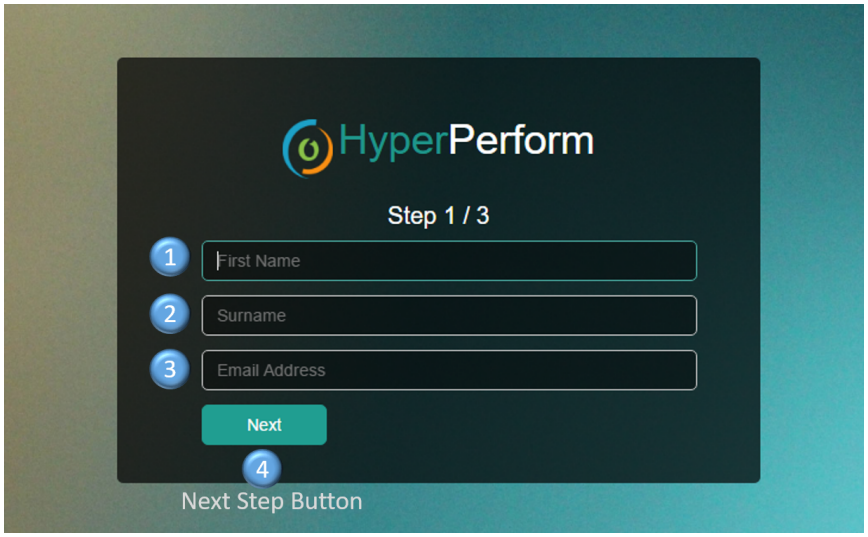

Figure 14: Registration step 1

1. **First Name:** The first name of the user being registered.
2. **Surname:** The surname of the user being registered.
3. **Email Address:** The email address with which the user being registered will login. **Note:** This must be a valid email address because the users password will be sent to them via email.
4. **Next Step Button:** Move from step 1 to step 2. Your input will be validated on click of this button.

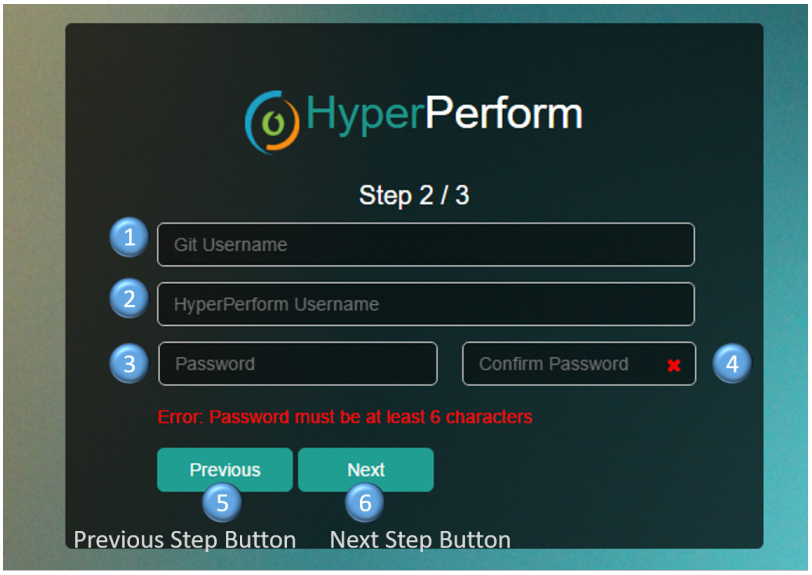

Figure 15: Registration step 2

1. **Git Username:** This is the users' exact git username which will be used in collecting information from their GitHub account.
2. **HyperPerform Username:** A username given to the registered user.
3. **Password:** The password the user will use to login to their dashboard.
4. **Confirm Password:** A confirmation of the password that was typed in the above field, used for security reasons.
5. **Previous Step Button:** Takes you back to step one where you can edit the information.
6. **Next Step Button:** Continues to step 3. Your input is validated again on click of this button.

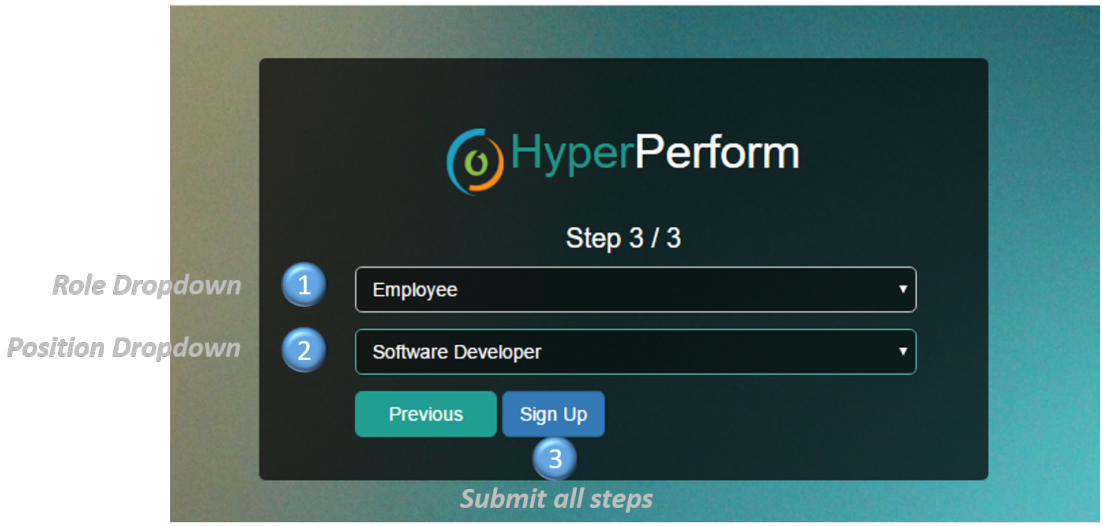

Figure 16: Registration step 3

1. **Role Dropdown:** A list of role types where the role of the user being registered is chosen. Currently existing roles are:
    - Administrator
    - Employee
2. **Position Dropdown:** A list of positions in the system where the position of the user being registered is chosen. Currently existing positions are:
    - Web Developer
    - Software Developer
    - Multimedia
3. **Submit all steps:** A button that submits all the fields that were input in all 3 steps. This will then send an email to the registered user which includes their login details. so that they can view their individual dashboard.

Currently existing roles are:
Currently existing positions are:

- Web Developer
- Software Developer
- Multimedia

4.5.3 Forecasting

The forecasting section allows for an administrator to add forecast values to the system. A forecast value is a prediction made by an administrator with regards to employee performance.

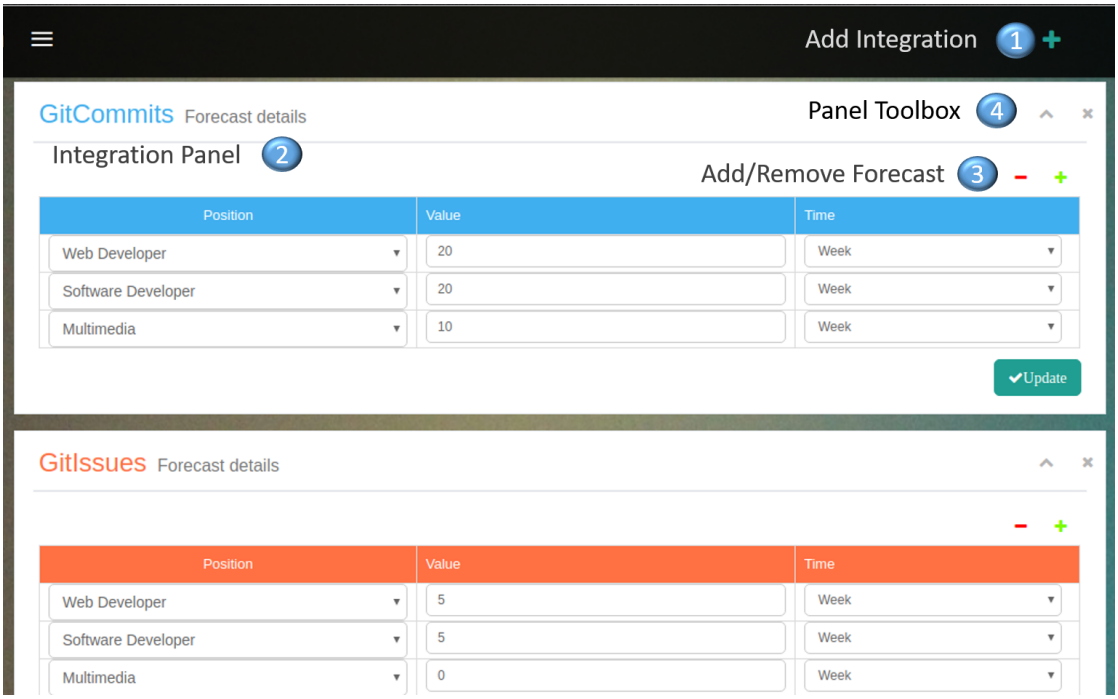

Figure 17: Dashboard errors

1. **Add Integration:** New integrations can also easily be added. Simply click on the + on the navbar to bring up a modal(Figure ?). On this modal a new integration can be added to the system. **Note:** when adding a new integration, at least one position and forecast value must be added.

2. **Integration Panel:** Each integration is displayed to the administrator in its own panel. Each panel consists of a table which shows each position along with its corresponding forecast value. All the values in the table are editable; this allows for different employee positions to have different forecast values in the same integration.

3. **Add/Remove Forecast:**

   (a) If the administrator does not wish to directly modify the table then simply click on '+' button to add a new row to the forecast table.

(b) If you want to remove a forecast from the table simply click the '-' button and it will remove a single row at a time.

(c) **Note:** you will not be able to add more rows in the table than there are positions in the position dropdown. Also you will not be able to remove all rows in a table, there must be at least 1 at all times.

4. **Panel Toolbox:** This toolbox allows you to do the following:

   (a) Minimise the panel with the ⋀ button so that you can see the rest of the integrations more clearly.

   (b) Delete the current integration with the 'x' button. When this is pressed a confirmation box will be shown to make sure that you want to delete the integration.

4.6 Errors

The dashboard provides error messages in the event there's an issue.

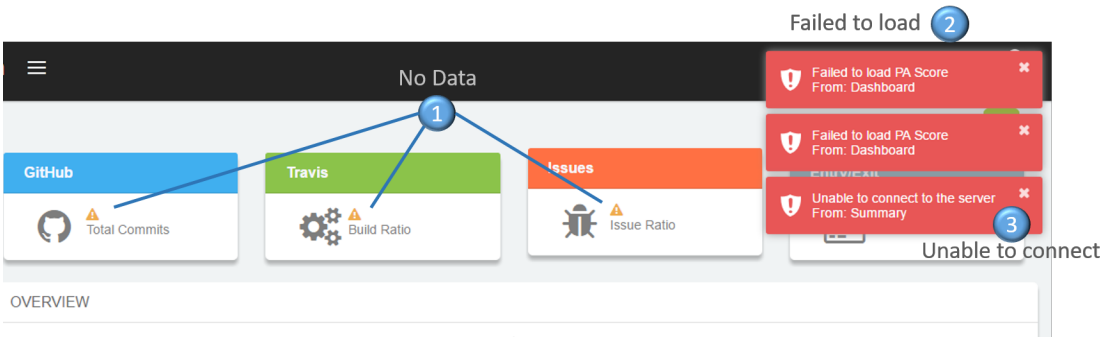

Figure 18: Dashboard errors

1. **No Data:** These icons in place of the values indicates that the data from the server was unable to be received by our dashboard. The reasons why are shown on the top right of the screen (also discussed below).

2. **Failed to load:** This indicates that the dashboard was unable to load the PA score due to an error. This is most likely that either you are not connected to the internet or that you have input an invalid date or time in the date picker.

3. **Unable to connect:** This error is due to the fact that you are most likely not connected to the internet or there is an issue on our side.

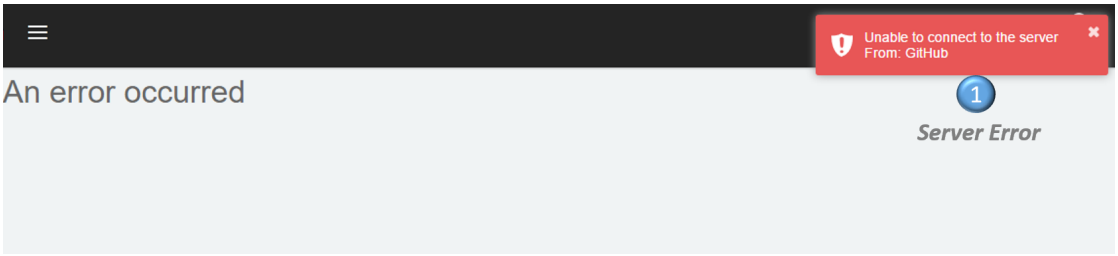

Figure 19: Server connection error

1. **Server Error:** If the system is not connected to the internet no data can be requested from the server.

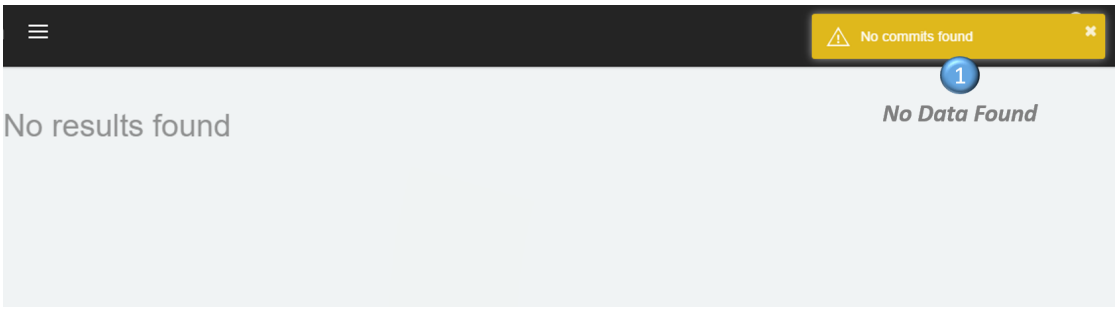

Figure 20: No data error

1. **No Data Found:** If a new integration has been added to the system and no data has been collected from that particular integration, then the user will see this error message.