Emergency Management Victoria Website

Production Support Manual

Sean Liew & Others

Table of Contents

[1. Setting Up 2](#_Toc436039381)

[1.1. Git Setup 2](#_Toc436039382)

[1.2. Continous Integration 7](#_Toc436039383)

[2. Source Code 12](#_Toc436039384)

[2.1. Repositories 12](#_Toc436039385)

[2.2. Front End Repository 12](#_Toc436039386)

[1.1.1. Incidents & Warnings 14](#_Toc436039387)

[1.1.2. Map 14](#_Toc436039388)

[1.1.3. Sidebar 16](#_Toc436039389)

[1.1.4. Text view 16](#_Toc436039390)

[1.1.5. Filter 16](#_Toc436039391)

[1.1.6. Sorting 16](#_Toc436039392)

[1.1.7. Locate Me 17](#_Toc436039393)

[1.1.8. Refresh 17](#_Toc436039394)

[1.1.9. Business Rules 17](#_Toc436039395)

[1.1.10. Accessibility 17](#_Toc436039396)

[1.1.11. Responsive 17](#_Toc436039397)

[1.1.12. Prepare and Get Ready 18](#_Toc436039398)

[1.1.13. Relief and Recovery 19](#_Toc436039399)

[2.3. Data Feed 19](#_Toc436039400)

[1.1.14. Warnings 19](#_Toc436039401)

[1.1.15. Incidents 22](#_Toc436039402)

[2.4. Osom Processor 23](#_Toc436039403)

[2.5. LeafletJS 27](#_Toc436039404)

[2.6. Gulpfile.js 27](#_Toc436039405)

[2.7. Handlebar Templates 28](#_Toc436039406)

[2.8. Test Driven Development 28](#_Toc436039407)

[2.9. Deployment 28](#_Toc436039408)

[2.10. Testing 29](#_Toc436039409)

[3. Contacts 29](#_Toc436039410)

# Setting Up

## Git Setup

# EMCOP EM-Public

Public information site for a state overview warnings and advisory site (Vic Emergency replacement).

## git setup

You can clone using ssh with a preshared SSH key with the following command:

git clone ssh://git@gitlab7.tools.vine.vic.gov.au:2022/empublic/web-src.git

## Setup

You will need `NodeJS` installed. At time of writing the minimum node version used is `0.10.31`. You can either install \*\*node\*\* from [`brew`](http://brew.sh/) if you are on a Mac OS or download a binary from [NodeJS download page](http://nodejs.org/download/).

### Windows

On Windows you will need to install some additional tooling, including the following:

\* [Python 2.7.10](https://www.python.org/downloads/windows/)

### All platforms

After you install you will need a couple of other things, namely [Bower](http://bower.io/) and [Gulp](gulpjs.com). To install them run the following:

npm install -g bower gulp

Then you will have to run the below commands in the `em-public` folder:

bower install

npm install

The above will install the dependencies necessary to run the public information site. \*\*Bower\*\* will install front end dependencies such as jQuery and \*\*NPM\*\* will install build tools needed such as SASS preprocessor, JSHint, Javascript compression tools, etc.

## Running

Gulp will execute SASS preprocessor, run JSHint against the Javascript files and compress both of them. It generates a `dist` folder which contains the minified version of both production ready.

To build everything just run

gulp

## Serving

As part of the development workflow you should be able to change the files and automatically see the changes on your browser. To run a lightweight webserver that will serve the content built simply run

gulp watch

## Building a Distribution Package

There are various optimisations performed when building for distribution. The dist target renders, uglifies, compresses, etc:

gulp dist

The package target creates a zip:

gulp package

## Testing

There are two testing targets: \*\*unit\*\* and \*\*functional\*\*.

### Unit Testing

The unit testing lives under `test/unit` and is done using [KarmaJS](karma-runner.github.io) + [Mocha](http://visionmedia.github.io/mocha/) + [Chai](http://chaijs.com/) + [Sinon](sinonjs.org)... explaining:

\* Karma is an agnostic test runner

\* Mocha is a testing library very similar to Jasmine, without the assertions

\* ChaiJS is an assertion library

\* Sinon is a mocking / stubbing / spying framework

To run the unit tests execute the following in your terminal:

gulp unit-test

The above will run the tests only once. If you prefer to keep them alive while making changes, simply run..

gulp unit-watch

### Functional Testing

The functional testing is done using RSpec and lives under `test/functional`. You will need Ruby 2.1.1. NodeJS is wiring the run of `rake` under its belt by spawning a process.

Before you can run the NodeJS task to execute the functional testing you will need to install dependencies yourself. Just run...

cd test/functional

bundle install

Then back to the root of the project run:

gulp functional-test

And a Firefox browser should come up with automated testing.

## Continous Integration

### Pushing to the main repo

All commits to the upstream repo will trigger a build on the Jenkins server. The build will be triggered by a [Git Trigger Job](http://jenkins.tools.vine.vic.gov.au/view/EMPublic/job/EMPublic-Git-Trigger/), which will result in a server-side build happening in the [Build no tests job](http://jenkins.tools.vine.vic.gov.au/view/EMPublic/job/EMPublic-Build-No-Tests/).

### Creating a release

When a series of changes are ready for release, the develop branch can be promoted to release through the [Git Release jenkins job](http://jenkins.tools.vine.vic.gov.au/view/EMPublic/job/EMPublic-Git-Release/). When triggered, it will:

\* Create a new release based on the current develop branch

\* Push this release into master

\* Bump the version number by a patch level (\*.\*.1)

\* Push the version bump back into the develop branch

## Git Flow

Git flow has been configured on this repo. This means that the git-flow command should be installed before performing any development work.

brew install git-flow

Once installed, set up your local repo and go for it.

To start developing, use the feature commands to work on a feature:

git flow feature start [featurename]

...

git flow feature finish [featurename]

To test a feature during development, or to share with other developers, publish the feature using:

git flow feature publish [featurename]

When changes are pushed to the origin git server via publish or finish, jenkinsci will automatically trigger a CI build and publish the result to S3. You can see the published build at:

http://public-info.dev.devcop.em.vic.gov.au/em-public/origin/develop/index.html (develop branch)

http://public-info.dev.devcop.em.vic.gov.au/em-public/origin/feature/[featurename]/index.html (published feature branch)

Once the feature has been completed (and pushed to the origin), it can be staged for release:

git flow release start [releasename]

...

git flow release finish [releasename]

Releases get merged into the master branch, and should get a new build via the jenkinsci CI job at:

http://public-info.dev.devcop.em.vic.gov.au/em-public/origin/master/index.html

# Hotfixing a production environment

To start a hotfix, checkout master and pull to get the latest commits:

git checkout master

git pull

Then create a hotfix feature:

git flow hotfix start emegencychangename

Make your changes and commit.

git add .

git commit -m'description of changes'

git push --set-upstream origin hotfix/emegencychangename

This will trigger a build on the JenkinsCI server, an automatic publish to the feature site, and allow a QA build to be generated.

To view the hotfix change in your browser, go to:

http://em-public.ci.devcop.em.vic.gov.au/em-public/origin/hotfix/emegencychangename/

Once you are happy with this change, generate a QA build by manually starting the JenkinsCI job:

Job Name: "EMPublic-Deploy-QA"

WEB\_SRC\_GIT\_TAG parameter: "hotfix/emegencychangename"

This will generate a new QA build for verification.

Once QA has been completed, the change can be merged into the master branch with git flow:

npm version patch

git flow hotfix finish emergencychangename

This will update the Master branch with the hotfix changes, and increment the patch version (\*.\*.1).

The production environments can then be released using the following JenkinsCI job:

Job Name: "EMPublic-Deploy-Prod"

Environment: PVT or Prod

WEB\_SRC\_GIT\_TAG: master

AWS\_ACCESS\_KEY\_ID: [as provided]

AWS\_SECRET\_ACCESS\_KEY: [as provided]

It is recommended that a PVT release is made first and verified at:

http://pvt.public.em.vic.gov.au/

The PROD release can then be made and verified with:

http://prod.public.em.vic.gov.au/

# Source Code

## Repositories

The repositories are hosted in Gitlab – click on the link below to access the repositories

<http://gitlab7.tools.vine.vic.gov.au/groups/empublic>

To clone the repositories into your local machine, type

Git clone <SSH or HTTP key>

Read the attached readme.md files in the respective repositories and follow the instructions listed in it

web-src – front end code for the main website

test-src – test data used to create custom datasets for testing purposes

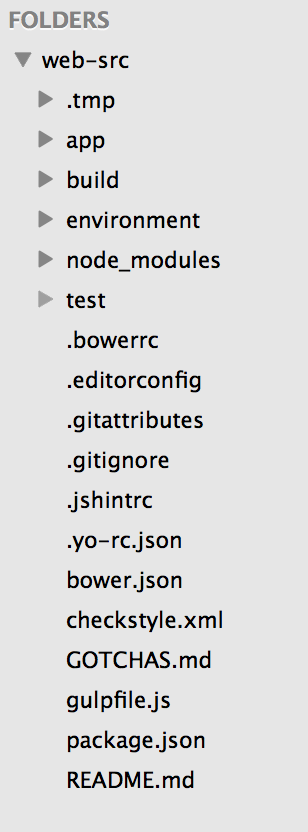
feed-src – Osom processor that aggregates the incoming data feed to be presented in a standardized JSON format that is displayed on the main website

backend-src – Backend code currently in development

## Front End Repository

Live site URL – <http://emergency.vic.gov.au>

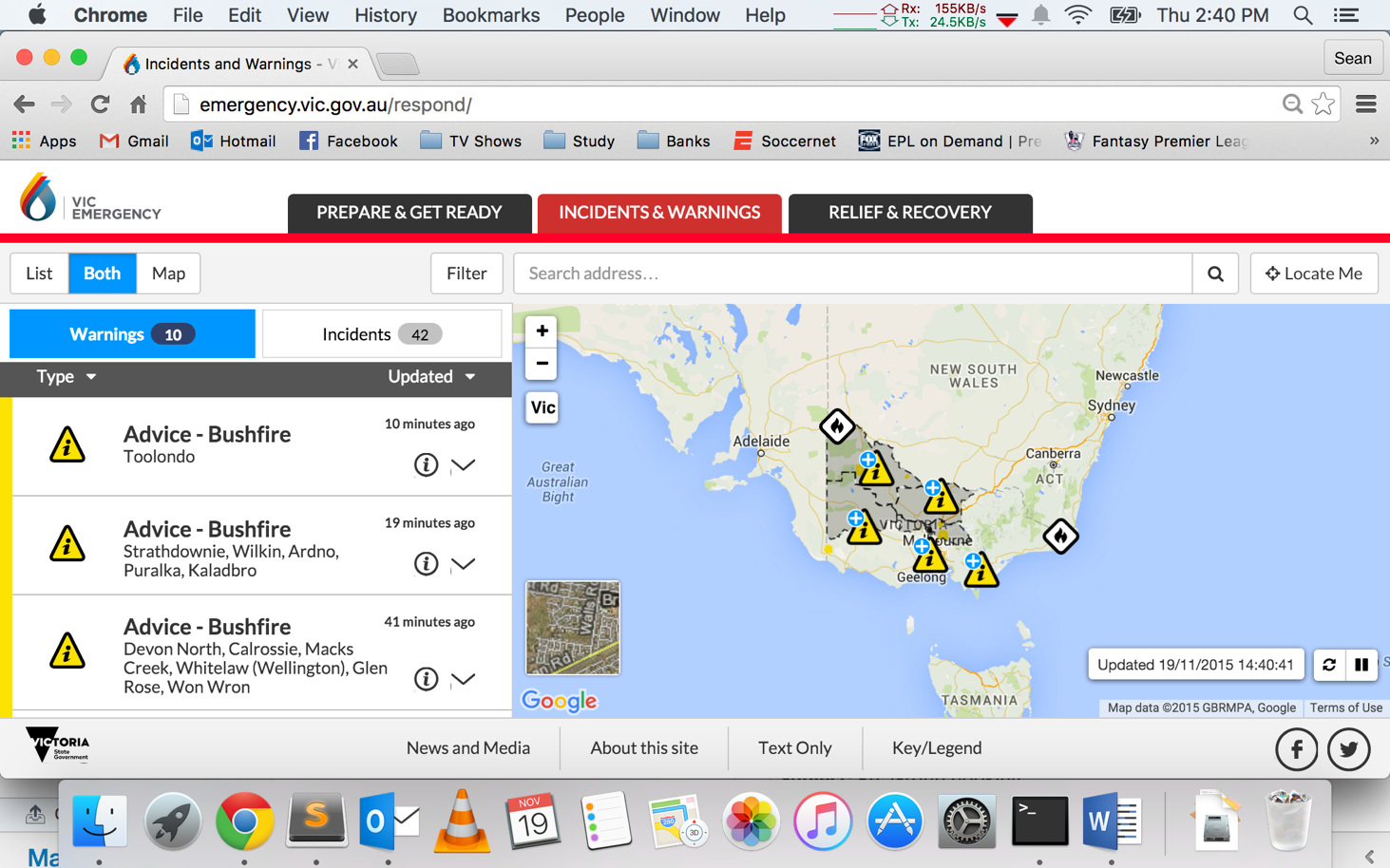
The structure of the repository is segregated into different folders as indicated in the screenshot below



Modification of the code will occur in the app directory where any changes to html, hbs, css, js, etc files will automatically be detected and built by the gulp build task (found in gulpfile.js) that compiles everything into the build directory.

The first release of this website is separated into 3 main pages as indicated below

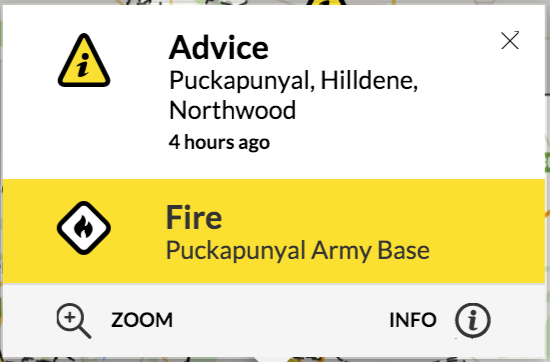
### Incidents & Warnings



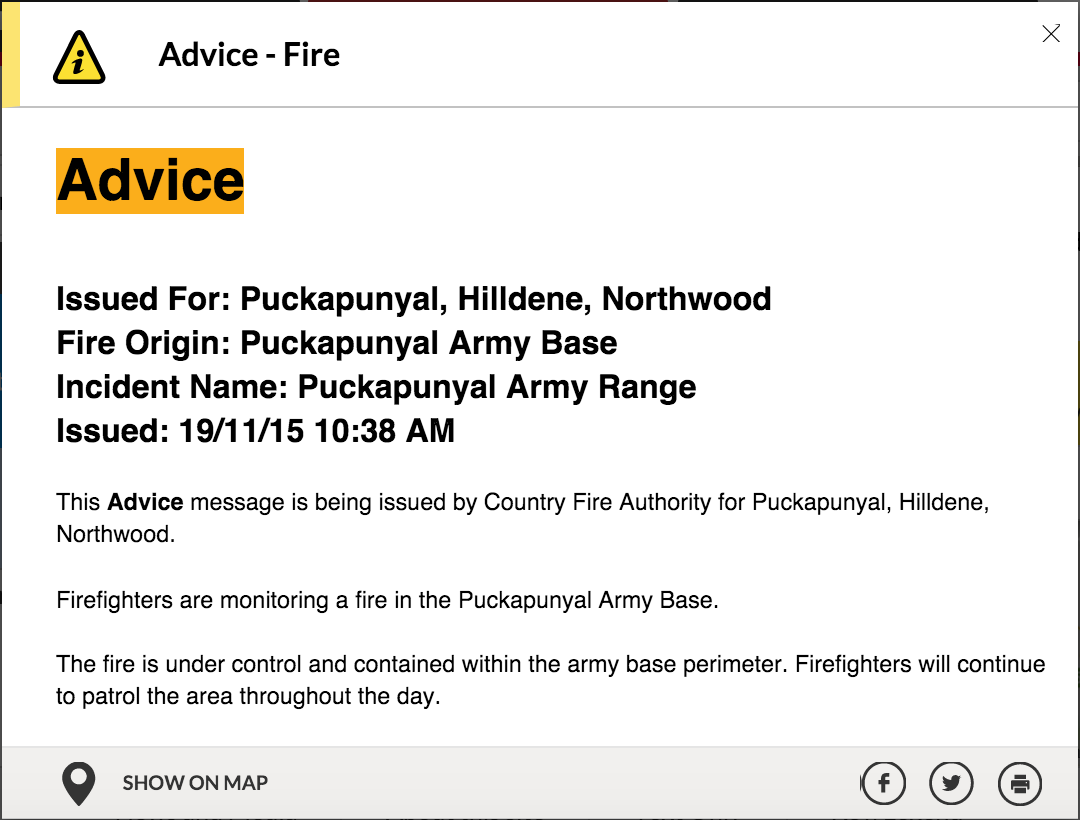
The index.html file for this page can be found in the respond folder within the web-src/app/html\_fragments/respond directory in the source code. This section will describe the main features of this page and highlights its file location in the source code.

### Map

All incoming data feed (see section on Data Feeds will be presented in this main page which is displayed in the map as SVG or PNG icons as shown in the screenshot above. Clicking on the relevant icon on the map will reveal a popup above the clicked icon containing dynamic live information as shown below



Clicking on the info link at the bottom right of the popup will reveal additional information as shown below



The content of the information is attached in the “webbody” property of the incoming JSON data feed which is handled by the backend system.

### Sidebar

A text based description of the incidents/warnings are also displayed in the sidebar table on the left hand side of the website. The sidebar table displaying the data is constantly rebuilt following any changes to the map’s field of view, zoom or data refresh which presents all the data that is currently visible on the map. There are two separate tables (one for warnings and one for incidents) that contains their respective datasets determined by the “feedtype” property in the data JSON structure.

Clicking on the list view will display hide the map and display the same sidebar tables in an alternative view. The source code for these tables can be found in the following files

App/templates/sidebar.osom-incident.hbs

App/templates/sidebar.osom-warning.hbs

App/templates/sidebar.other.hbs

### Text view

Please contact [Michael.Jenkins@vine.vic.gov.au](mailto:Michael.Jenkins@vine.vic.gov.au) for further information

### Filter

Filtering is presented as a dropdown to enable users to determine which categories of incidents to be displayed on the map and sidebar. The source code can be found in app/scripts/ui/filter.js. In mobile view, the filter is displayed as a modal window which leverages the same Javascript functions that are synchronized with the desktop version of filters.

The filter list also enables users to choose between the map overlays that the users chooses to be displayed on top of their map. The functions can be found in the same app/scripts/ui/filter.js file.

### Sorting

Sorting is done using the library ListJS. The function governing the sorting rules can be found in the app/scripts/sidebar.js file.

Further documentation can be found on <http://www.listjs.com/>

### Locate Me

The locate me functionality is leveraged off the library leaflet-locatecontrol. Customised controls have been created to perform additional functions (such as calculating distances between incidents and location of user) can be found in the app/scripts/ui/leaflet/ directory. Further documentation on this library can be found on <https://github.com/domoritz/leaflet-locatecontrol>

### Refresh

The refresh functionality is used to refresh the data on the map that is set to occur every 60 seconds by default. There is a pause functionality that is built to negate the refresh functionality as part of accessibility standards. The source code for the refresh functionality can be found in

App/scripts/ui/refresh.js

App/scripts/ui/refresh.manager.js

App/scripts/ui/leaflet/control.refresh.js

### Business Rules

The business rules to govern how warnings and incidents are classified and displayed on the main website can be found in the app/scripts/rules/osom.js

For further information on business rules, please contact [Reegan.Key@emv.vic.gov.au](mailto:Reegan.Key@emv.vic.gov.au)

### Accessibility

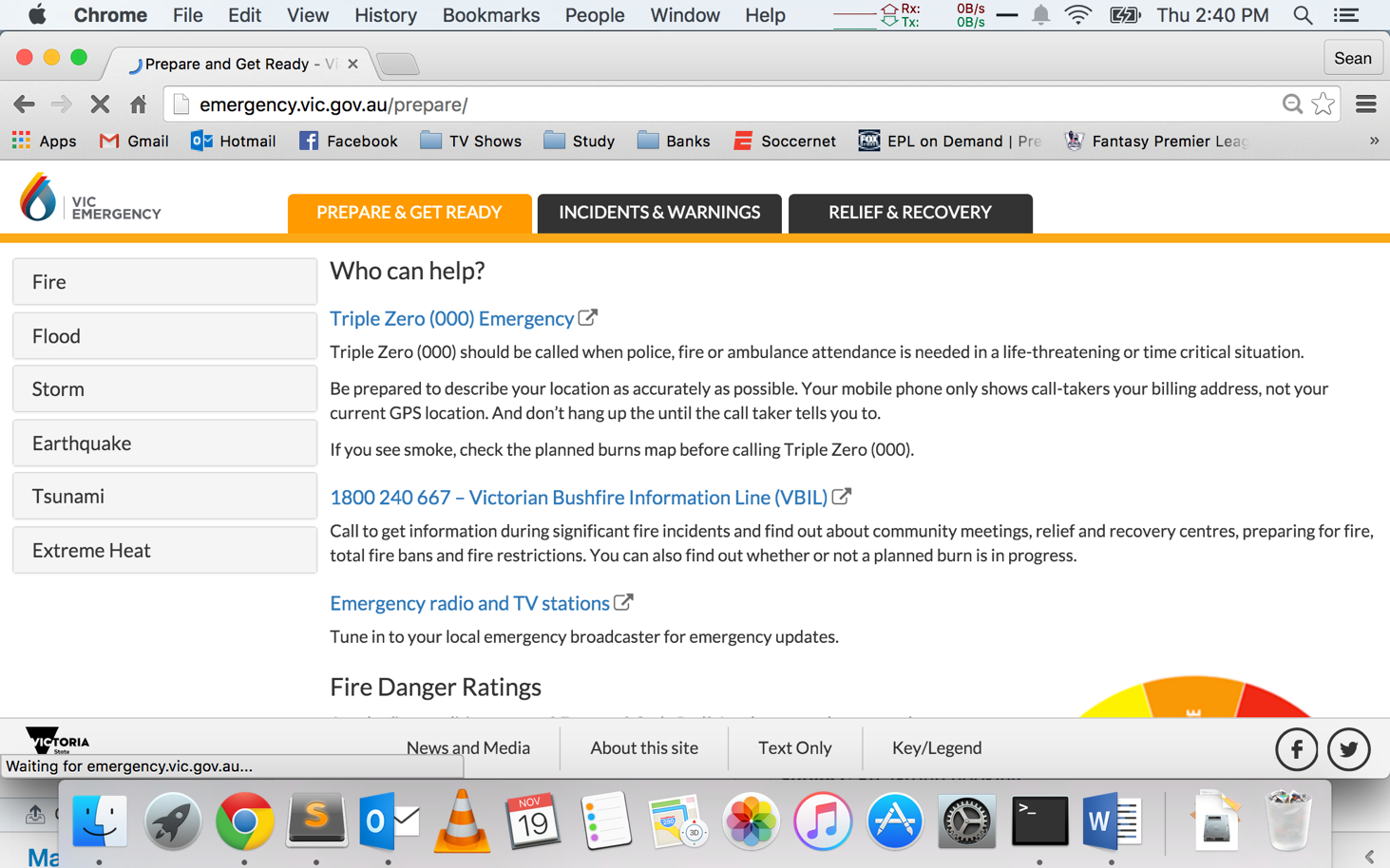
The site has gone through significant WCAG 2.0 AA assessment by Vision Australia. Please consider the impact on accessibility when making changes to the source code. For further information on accessibility standards, please contact [Sean.Liew@smsmt.com](mailto:Johnathan.Gardner@smsmt.com)

The latest accessibility testing report can be found [here](../Downloads/Accessibility%20Review%20VIC%20Emergency%20website%20V1.0.docx)

### Responsive

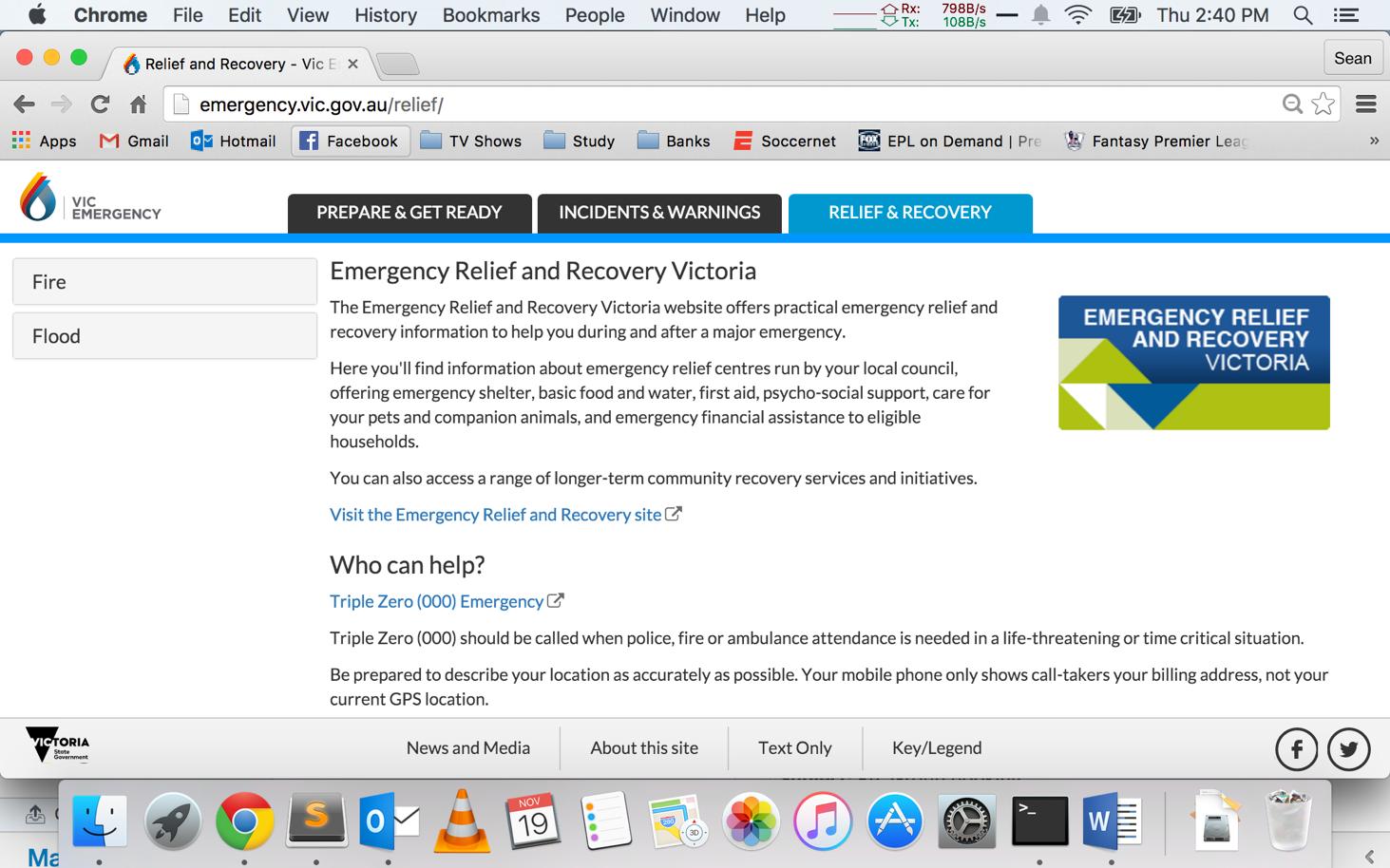
The website is designed to cater for mobile responsiveness to ensure proper functionality in portrait and landscape view for Android and iOS devices. For further information on the requirements for mobile capability, please contact [Johnathan.Gardner@smsmt.com](mailto:Johnathan.Gardner@smsmt.com)

### Prepare and Get Ready



The index.html file for this page can be found in the respond folder within the web-src/app/html\_fragments/prepare directory in the source code

### Relief and Recovery



The index.html file for this page can be found in the respond folder within the web-src/app/html\_fragments/relief directory in the source code

## Data Feed

### Warnings

There are predefined business rules implemented to handle the incoming datafeed which is kept in app/scripts/rules/..

These might be a good place to study the rules that govern how the data is processed and presented in the website

Sample of incident datafeed

{

"type": "Feature",

"properties": {

"feedType": "warning",

"id": "11244261",

"category1": "Watch",

"category2": "Flood",

"location": "Werribee, Maribyrnong, Yarra, Dandenong And Bunyip",

"created": "2015-05-13T01:53:35.000Z",

"incidentFeatures": [

{

"properties": {

"feedType": "incident",

"category1": "Flood",

"category2": "Flood",

"location": "Greater Melbourne Catchments (Werribee, Maribyrnong, Yarra, Dandenong And Bunyip)"

}

}

],

"webBody": "Flood Watch for Greater Melbourne Catchments (Werribee, Maribyrnong, Yarra, Dandenong And Bunyip) ï¿½ ï¿½ Effective Date: 13/05/15 12:00 PMExpiry Date: 14/05/15 3:00 PMï¿½ï¿½ Current Situation ï¿½ In the 24 hours to 9AM Wednesday rainfall totals of up to 42mm have been recorded in the Greater Melbourne catchments. A further 5-10mm of rain, with locally higher totals up to 15mm, is forecast for the eastern catchments (Yarra, Bunyip and Dandenong) for the rest of Wednesday. Showers will continue on Thursday, but with only around 5mm forecast. Stream rises have been observed in response to the recent rainfall. If the forecast rainfall is in the higher range of the forecast, then isolated areas of minor flooding may develop in the eastern catchments (Yarra, Bunyip and Dandenong) overnight Wednesday into Thursday. The Bureau of Meteorology in conjunction with Melbourne Water will continue to monitor the situation and issue catchment specific warnings if and when required.Note: This Flood Watch means that people living or working along rivers and streams must monitor the latest weather forecasts and warnings and be ready to move to higher ground should flooding develop. Flood Warnings will be issued if Minor Flood Level is expected to be exceeded at key sites along the main rivers for which the Bureau of Meteorology provides a flood warning service. For detailed explanation see http://www.bom.gov.au/vic/flood/brochures/flood\_watch/flood\_watch.shtml ï¿½ Potential Impact In Your Area The potential local impact of this flooding in your area may include:ï¿½ River levels may reach the top of the bank or flow into low-lying areasLocal roads may be closed and low bridges may be underwaterCaravan parks and camping areas may be floodedWhat You Should Doï¿½ Stay informed - monitor local conditions and be aware of the situationFarmers should prepare to move machinery and livestock to higher groundï¿½ Tune in to your emergency broadcasters: ABC Local Radio, commercial radio and designated community radio stations, or SKY NEWS TelevisionEnsure your family members and neighbours are aware of the situationFloodwaterï¿½is dangerous - never drive, walk or ride through floodwaterFloodwaterï¿½is toxic - never play or swim in floodwaterEmergency Contactsï¿½ For flood or storm emergency assistance from the SES call 132 500ï¿½ For life threatening emergencies call Triple Zero (000) Additionalï¿½Information Road Closures and Alerts: vicroads.vic.gov.au phone 131 170 Weather Warnings and River Heights: bom.gov.auphone 1300 659 217 SES Information: ses.vic.gov.au Issued By Victoria State Emergency Service"

},

"geometry": {

"type": "GeometryCollection",

"geometries": [

{

"type": "Point",

"coordinates": [

"144.64899",

"-37.90653"

]

},

{

"type": "Point",

"coordinates": [

"144.87981",

"-37.76338"

]

},

{

"type": "Point",

"coordinates": [

"145.3759",

"-37.66127"

]

},

{

"type": "Point",

"coordinates": [

"145.23926",

"-37.95185"

]

},

{

"type": "Point",

"coordinates": [

"145.75188",

"-38.08209"

]

}

]

}

},

The code above is a sample of a flood warning (classified by the “feedtype” property) with a moderate severity (denoted by the “category1” property usually represented by a ‘watch’ or ‘moderate’ field). The attached incident that is presented as a dropdown in the sidebar is denoted by the “incidentFeatures” property and the “webbody” property displays the more info that is presented as a modal window. This warning is represented as 5 different orange points on the map (denoted by the 5 array items in the “geometries” property.

### Incidents

{

"type": "Feature",

"geometry": {

"type": "Point",

"coordinates": [

142.2067,

-34.1839

]

},

"properties": {

"feedType": "incident",

"category1": "Fire",

"category2": "Bushfire",

"created": "2015-11-03T12:00:00.000Z",

"cssClass": "fire-active",

"id": 1544196,

"location": "7KM NE OF MILDURA",

"resources": 1,

"size": "0.50 HA.",

"sizeFmt": "0.50 HA.",

"status": "Not Yet Under Control",

"updated": "2015-11-04T01:32:00.000Z"

}

},

The code above displays a normal bushfire incident that is usually displayed as a white icon on the map. The general structure of a fire incident is seen above where the incident is denoted by the “feedtype” property. The UI design for the incident popup and sidebar is kept in and “app/templates/popup.incident.hbs” and “app/templates/panel.incident.hbs” respectively. Referring to the other templates in the same folder, different templates are used for other types of data such as burn area, earthquakes etc.

## Osom Processor

The Osom processor is used to aggregate the incoming data feed to be presented in a standardized JSON structure to be presented in the main website site. The repository for the source code can be found in <http://gitlab7.tools.vine.vic.gov.au/groups/empublic> under the name feed-src.

The instructions for setting up is listed below or can be found in the readme.md file in the repo

# Local execution

To install dependencies, run

npm install

## Configuration

The processor uses layered configuration files to determine runtime operation. In typical use, one "input" config and one "output" config are specified.

For example, to run the processor against live data sources and generate to a local development instance of EM-Public, use:

node cli.js config/inputs/production.json config/outputs/localhost.json

This will write the resulting output files to:

../web-src/build/data

Once created, these data files can be referred to by the local EM-Public app using the testdata feature toggle:

http://localhost:9000/respond/?ft=testdata

## Local testing scenarios

Various scenarios are available for testing locally. These scenarios use historic and constucted data to generate data files and place them within the EM-Public build/data directory.

For example, Scenario 1a (BOM Severe Weather Incident without matching Ripe data) can be generated as follows:

node cli.js config/inputs/sample.json config/inputs/scenario-1b.json config/outputs/localhost.json

This combines the sample data set with the scenario data files and generates the output to localhost.

# Containerisation

To containerise the app with docker, use the following:

## Setup for OSX

boot2docker init

boot2docker start

$(boot2docker shellinit)

## Build a new image

docker build -t osom-processor .

## Run the image

docker run osom-processor

# Emergency Change

## Git flow process around making an emergency change

git checkout master

git flow hotfix start [change]

npm version patch

... changes and commits ...

git flow hotfix finish [change]

git push

git checkout master

git push

## Releasing to Lambda

Run the following jenkins-ci job to update the PVT environment for final validation:

Job name: EMPublic-Deploy-Feed-Prod

ENVIRONMENT: PVT

AWS\_ACCESS\_KEY\_ID: As provided

AWS\_SECRET\_ACCESS\_KEY: As provided

Once ready for a release, run the following jenkins-ci job:

Job name: EMPublic-Deploy-Feed-Prod

ENVIRONMENT: PROD

AWS\_ACCESS\_KEY\_ID: As provided

AWS\_SECRET\_ACCESS\_KEY: As provided

For further information, please contact the author Michael.jenkins@vine.vic.gov.au

## LeafletJS

The mapping feature of the site is delivered using the leafletJS library (we use Google maps at the moment but we may decide to convert to Vic maps in the future). The source code can be found in the app/scripts/ui/map.js file.

For further documentation on LeafletJS, please refer to <http://leafletjs.com/>

## Gulpfile.js

For further documentation on Gulp, please refer to <http://gulpjs.com/>

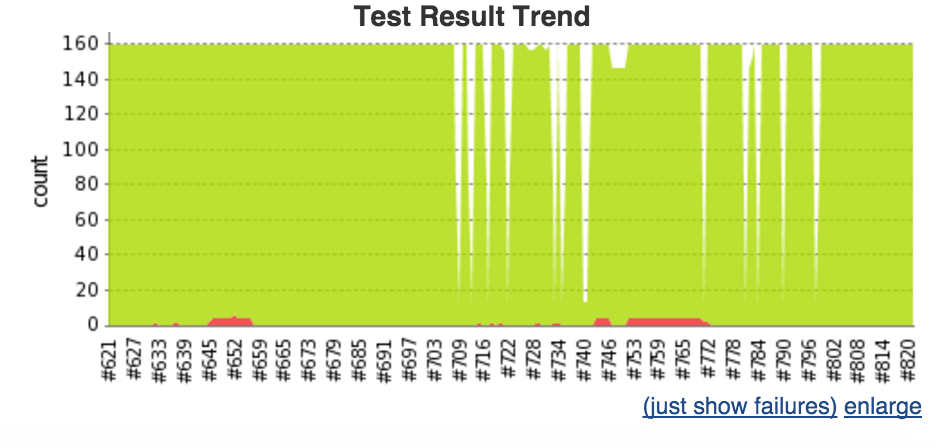
## Handlebar Templates

The site leverages handlebar templates to define the html structure of several items (e.g. popups, sidebar tables) in the site. The block helpers to create dynamic contents in the handlebar templates can be found in the app/scripts/patch/handlebars.js file.

For further documentation on handlebars and block helpers, please refer to <http://handlebarsjs.com>

## Test Driven Development

Unit and functional tests are written in the test directory in the web-src repository. The rule of thumb is to create unit tests for every JS function. Test coverage are monitored in the public-build-dist directory in Jenkins CI



where any failures in the functional testcases will result in a broken or unstable build which prevents Jenkins from updating the latest committed code into the develop environment (see ‘Deployment’ section.

## Deployment

Jenkins CI deployment

EM-Public Group Folder - <http://jenkins.tools.vine.vic.gov.au/view/EMPublic/>

EM-Public Dev Build Environment - <http://jenkins.tools.vine.vic.gov.au/view/EMPublic/job/EMPublic-Build-Dist/>

See <http://em-public.ci.devcop.em.vic.gov.au/em-public/origin/develop/respond/>

EM-Public Live Deployment Environment - <http://jenkins.tools.vine.vic.gov.au/view/EMPublic/job/EMPublic-Git-Release/>

## Testing

Jenkins Testing Environment - <http://jenkins.tools.vine.vic.gov.au/view/EMPublic/job/EMPublic-Deploy-QA/>

See <http://em-public.ci.devcop.em.vic.gov.au/em-public/qa/312/respond/>

Number in red should be changed depending on the QA build being generated

# Contacts

Chief architect – [Michael.jenkins@vine.vic.gov.au](mailto:Michael.jenkins@vine.vic.gov.au)

Lead developer – [David.Carroll@smsmt.com](mailto:David.Carroll@smsmt.com)

Developer – [Sean.Liew@smsmt.com](mailto:Sean.Liew@smsmt.com)

Developer – [Truong.N.Nguyen@smsmt.com](mailto:Truong.N.Nguyen@smsmt.com)

Project Manager – [Sean.A.Morrison@vine.vic.gov.au](mailto:Sean.A.Morrison@vine.vic.gov.au)

Product Owner – [Reegan.Key@emv.vic.gov.au](mailto:Reegan.Key@emv.vic.gov.au)

BA – [Johnathan.Gardner@smsmt.com](mailto:Johnathan.Gardner@smsmt.com)