Software Design Specification

for

Soothsayer

Version 2.0 approved

Prepared by Ruslan Kolesnik, Caleb Hellickson, Michael Moss,

Ignacio Sáez Lahidalga, and Paul Gentemann

UAF

April 30, 2014

Table of Contents

Revision History

Date	Notes	Version
02/15/2014	**Rough draft.**	**1.0**
02/16/2014	**Writing session.**	**1.1**
04/30/2014	**Writing session.**	**1.2**
05/01/2014	**Final version.**	**1.3**

1. Introduction

1.1. Purpose

This document describes the system architecture and design of Soothsayer, a framework meant to replace the functionality of the current forecast system, database, and web services of the Aurora Forecast.

1.2. Scope

Soothsayer's main goal is to provide forecast data graphically in a web browser. Soothsayer also allows web clients to upload photos to a gallery, receive notifications, and download forecast data in a standard format. Soothsayer is meant to be an easily distributed platform that can be modified and evolved throughout its lifetime.

1.3. Overview

This document will describe the internal workings of Soothsayer, from a high level to a low level, decomposing all its modules.

1.4. Definitions and Acronyms

AASRG – Auroral Activity by Solar Rotation Graph
CGI – Common Gateway Interface
CMS – Content Management System
Drupal – An open-source content management system.
JSON – Javascript Object Notation
KP – An index value that quantifies disturbances in the horizontal component of Earth's magnetic field with an integer in the range 0-9, which leads to a roughly logarithmic scale.
NOAA – The National Oceanic and Atmospheric Administration
PNG – Portable Network Graphics
WebGL – (Web Graphic Library) is a JavaScript API for rendering interactive 3D and 2D graphics within any compatible web browser.
ZIP – Archive file format that supports lossless data compression

2. System Overview

Soothsayer provides web services to clients that include Auroral phenomenon.

3. System Architecture

3.1. Architectural Design

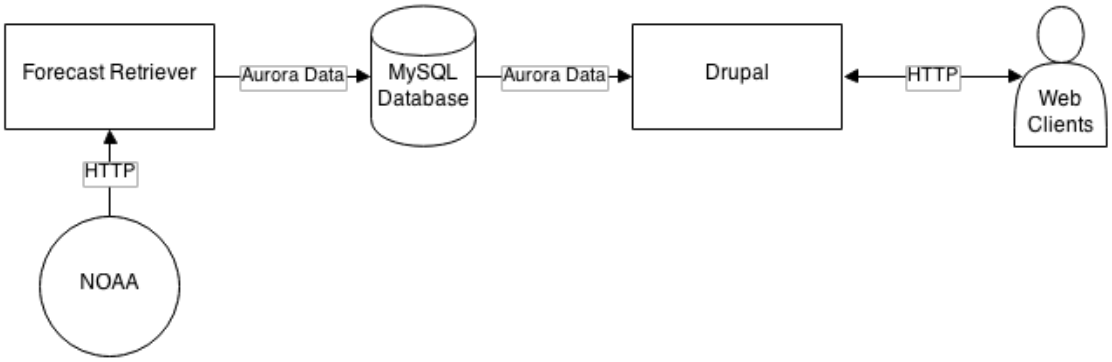
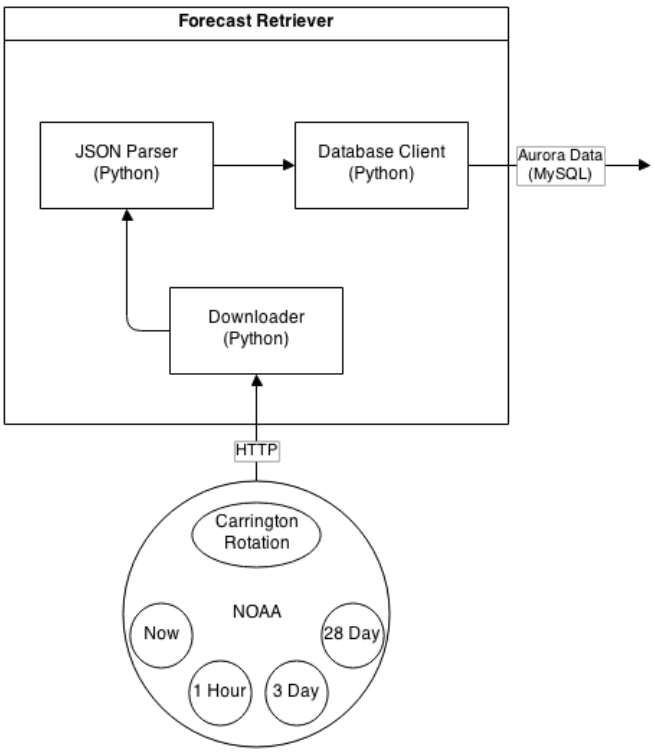


Figure 0 – Overview

3.2. Decomposition Description

There are four main components of Soothsayer.

3.2.1. Retriever



The retriever includes the following files:

- date_util.py
- db_util.py
- emailer.py
- file_util.py
- forecast_parser.py
- forecast_retriever.py
- json_util.py
- kp_cgi.py
- kp_util.py
- string_util.py
- unit_tests.py
- url_util.py
- forecast_retriever.cfg

All configuration options and variables are in forecast_retriever.cfg. This is the configuration file for all the the retriever components. This file contains the following:

[Contact Info]
receiver_email = Administrator <admin@alaska.edu>

[Data Resources]
now_forecast_link = http://www.forecast.com/now.txt
h1_forecast_link = http://www.forecast.com/1hour.txt
d3_forecast_link = http://www.forecast.com/3day_%m_%d.txt
d28_forecast_link = http://www.forecast.com/28day.txt
cr_link = http://www.forecast.com/carrington_rotation.txt

[Database]
forecast_database_name = forecast_db
forecast_database_password = password

The configuration file should not be readable or writable by anyone but root, since it contains the password to the forecast database.

Note the special syntax of the d28_forecast_link variable of the Data Resources header. To be flexible with filenames, the retriever allows use %m and %d to change the names of files dynamically. The %m escape sequence inserts the current month and the %d escape sequence inserts the current day.

Ex:

http://www.forecast.com/3day_%m_%d.txt

becomes

http://www.forecast.com/3day_05_23.txt

Forecast data is downloaded and passed through a parser, forecast_parser.py, that converts the data into a standard JSON format:

```
{
  "values":
  [
        {
              "download_time":
              {
                    "year":(integer 1970+),
                    "month":(integer 1 to 12),
                    "day":(integer 1 to 31),
                    "hour":(integer 0 to 23),
                    "minute":(integer 0 to 59)
              },
              "predicted_time":
              {
                    "year":(integer 1970+),
                    "month":(integer 1 to 12),
                    "day":(integer 1 to 31),
                    "hour":(integer 0 to 23),
                    "minute":(integer 0 to 59)
              },
              "kp":(float 0.0 to 9.0)
        },
        ...
  ]
}
```

The download time represents the time the data was downloaded from the online source. The predicted time represents the time the data is predicted for. The KP is the expected KP for the predicted time.

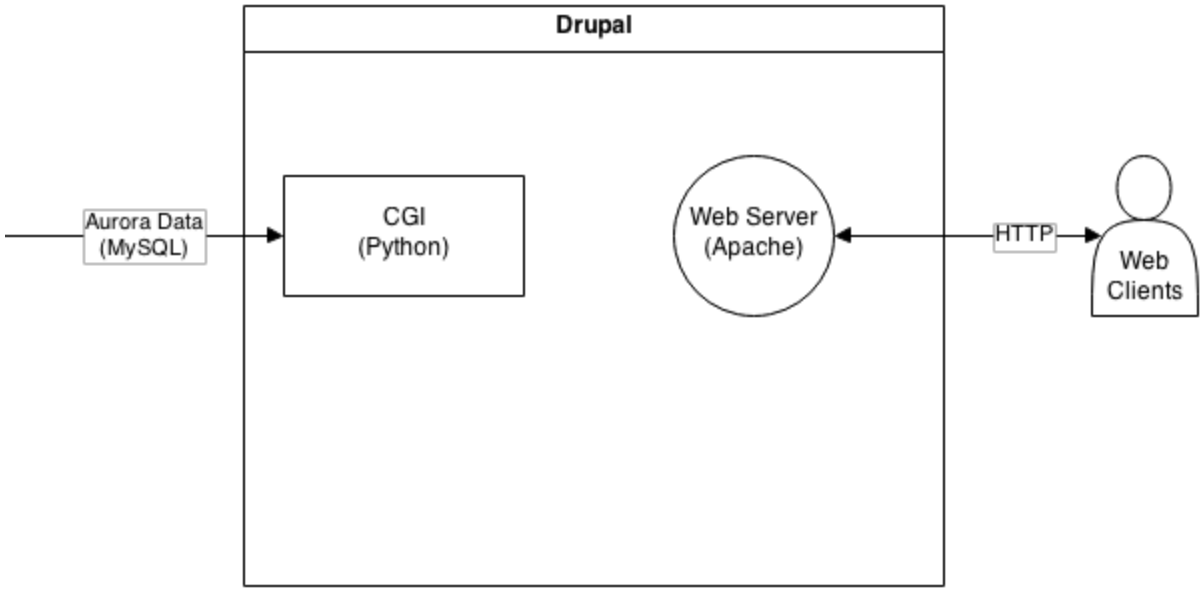
Carrington rotation data is downloaded in a similar JSON format:

```
{
  "values":
  [
        {
              "rotation_index":(integer),
              "year":(integer),
              "month":(1 to 12),
              "day":(float 0.0 to 32.0)
        },
        ...
  ]
}
```

The file forecast_retriever.py is ran as a cron job whenever it is required. The default cron settings are as follows:

- Now - Every 15 minutes.
- 1 Hour - Every 1 hour.
- 3 Day - Every 3 hours.
- 28 Day - Every 24 hours.

3.2.2. Drupal



Drupal Modules

Dupal will need the following modules installed and enabled:

- Chaos Tools Suite (Ctools) 7.x-1.4
  Required by Panels, Views, and Views Infinite Scroll modules
- Date version 7.x-2.7
  Provides a date picker for Drupal date fields. Used in the Special Forecast content type for the “Unpublish on” field.
- Entity 7.x-1.5
  Required by Simple News module.
- Panels 7.x-3.4
  Used to customize the layout of a page. Allows you to insert a view anywhere on the page.
- Rules 7.x-2.6
  Required by Simple News and Entity modules.
- Scheduler 7.x-1.2
  Allows nodes to be published and unpublished on specified dates. Need to create the Special Forecast content type.
- Simplenews 7.x-1.1
  Provides the ability to publish and send newsletters to lists of subscribers. Used to create the Notification content type.
- Views 7.x-3.7

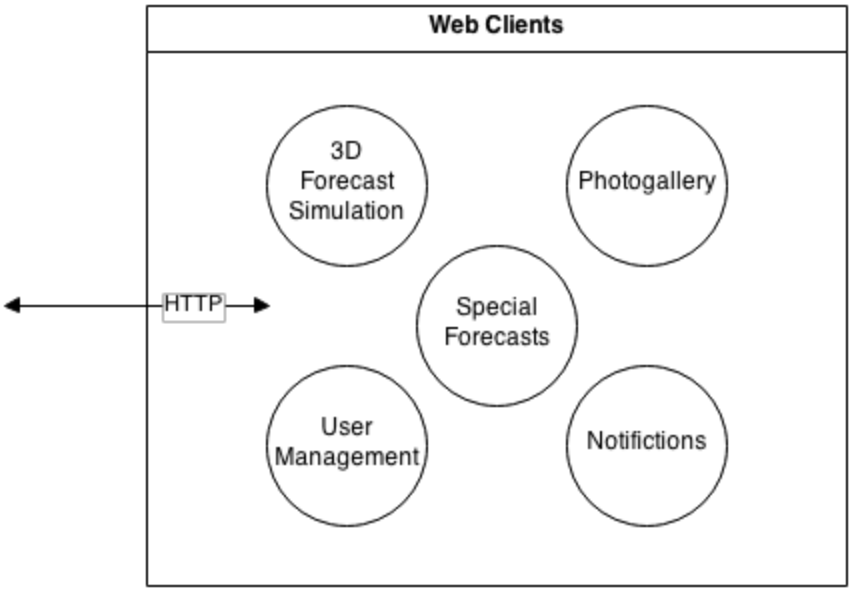
Used to customize how content is displayed. Used to create the Photo gallery page.

- Views Infinite Scroll 7.x-1.1
  Provides infinite scroll capability to selected views and pages.

Drupal Content Types

- **Image**
  Description: Allows users with an account to upload images.
  Fields: Title, Photo, Comment, Location, GPS, Camera
- **Special Forecast**
  Descriptions: Allows administrators and privileged users to publish a special forecast comment for a set period of time.
  Fields: Title, Body, Notification
- **Notification**
  Description: Allows administrators and privileged users to send a notification.
  Field: Title, Body, Unpublish on

3.2.3. Web Client



To get data from the database, the kp_retriever.py and cr_retriever.py are used.

The kp_cgi.py file serves web clients forecast data using CGI. The CGI request is made via a GET request. The request is outlined as:

http://server.com/cgi-bin/kp_cgi.py?r=REQUEST_OBJECT

The REQUEST_OBJECT above is actually a JSON string. The string is as

follows:

```
{
   “forecast”:(string “now”, ”h1”, ”d3”, or “d28”),
   ”predicted_time”:
   {
         "year":(integer 1970+),
         "month":(integer 1 to 12, -1 = all months in year),
         "day":(integer 1 to 31, -1 = all days in month),
         "hour":(integer 0 to 23, -1 = all hours in day),
         "minute":(integer 0 to 59, -1 = all minutes in hour)
   }
}
```

The cr_cgi.py file serves web clients Carrington rotation data using CGI. The CGI request is made via a GET request. The request is outlined as follows:

http://server.com/cgi-bin/cr_cgi.py?r=REQUEST_OBJECT

The REQUEST_OBJECT above is actually a JSON string. The string is as follows (note at least one variable must be supplied):

```
{
   "year":(integer optional),
   "rotation_index":(integer optional)
}
```

There are three types of graphics that use the data above. The first is the 2D Fallback.

The 2D Fallback is an image based system for use on systems that do not support WebGL.

The second type of graphic is the 3D Simulation. The 3D Simulation is written in Javascript. The 3D Simulation uses THREE.JS for rendering. The rendering is done on the GPU using GLSL. The shaders draw a circular pattern on a hemisphere. There is one hemisphere per magnetic pole. The 3D Simulation is used for the now and 1 hour forecasts.

The third type of graphic is the Auroral Activity by Solar Rotation Graph (AASRG). The AASRG is used for the 28 day forecast. The AASRG uses Canvas.JS for its rendering.

Hammer.JS is used for touchscreen interaction.

JSZip.JS is used to create the 2d fallback pictures from the simulation. When used, the web page will render all the views required for the 2D

fallback in PNG format and download them as a ZIP file.

3.2.4. Database

The database must be MySQL compatible. The organization is as follows:

All data is put into a MySQL compatible database in the following format:

```
table now
(
   id integer auto_increment not null,
   predicted_time datetime not null,
   download_time datetime not null,
   kp float not null,
   primary key (id)
);
table h1
(
   id integer auto_increment not null,
   predicted_time datetime not null,
   download_time datetime not null,
   kp float not null,
   primary key (id)
);
table d3
(
   id integer auto_increment not null,
   predicted_time datetime not null,
   download_time datetime not null,
   kp float not null,
   primary key (id)
);
table d28
(
   id integer auto_increment not null,
   predicted_time datetime not null,
   download_time datetime not null,
   kp float not null,
   primary key (id)
);
table cr
(
   rotation_index integer not null,
   year integer not null,
   month integer not null,
   day float not null,
   primary key (rotation_index)
);
```

3.3. Design Rationale

This design was chosen because it makes the application easy to evolve and modify. All elements are separated in such a way that they are modular, and can easily be swapped out with a different solution with minimal modifications to their existing state.

4. Data Design

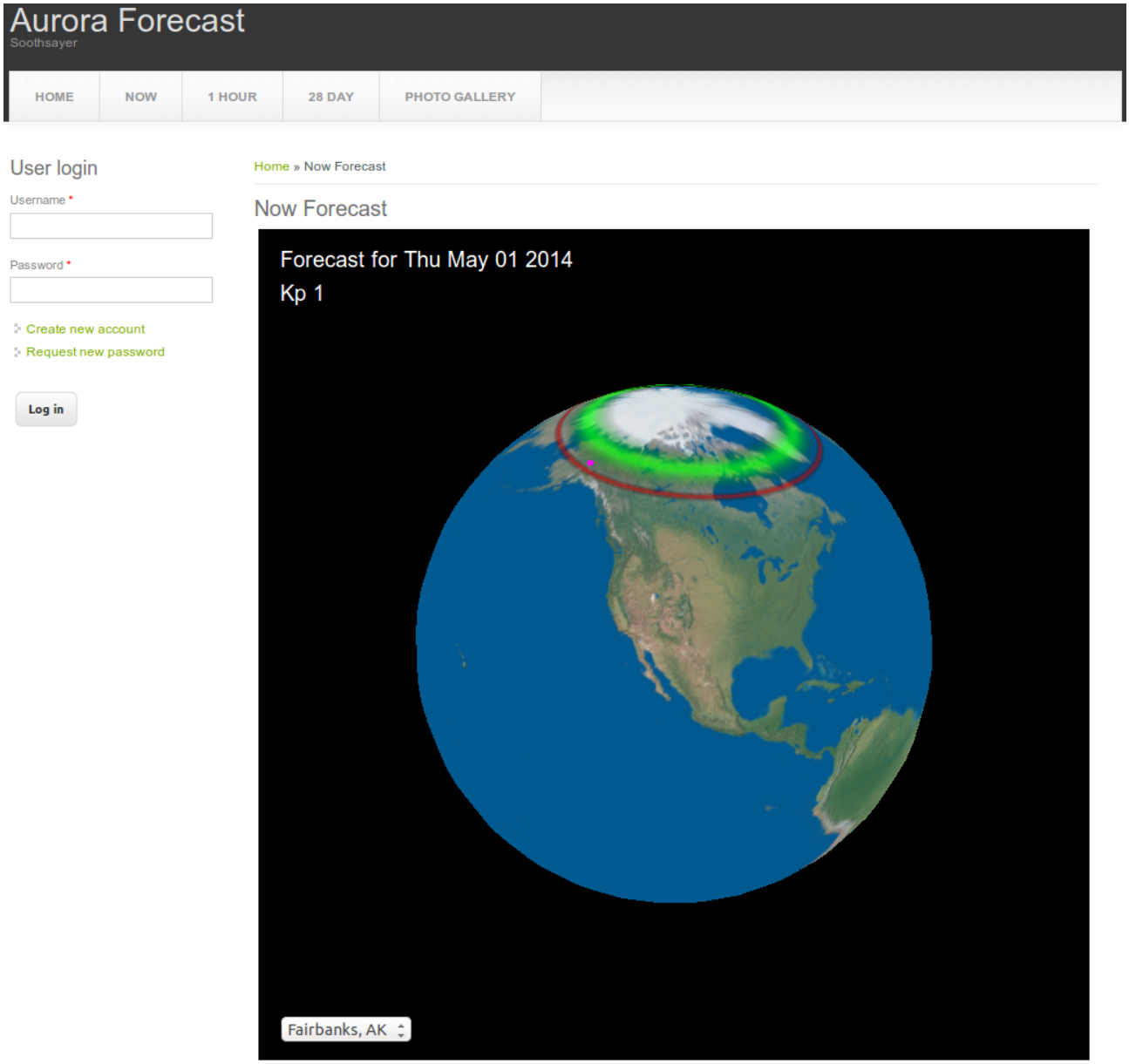
The main source of data Soothsayer will be pulling from is NOAA. Once the data is pulled a python script will be used to convert that data into several JSON strings. Those JSON strings will then be parsed for each day's KP values, which will be stored into the database.For displaying the different aurora forecasts, the website will retrieve the information from the database.

5. Human Interface Design

5.1. Overview of User Interface

The user interface shall consist of a collection of web pages.

5.2. Screen Images



Home page mockup.

User login

Username *

Password *

Create new account

Request new password

Log in

Home » Photo Gallery

Photo Gallery

Aurora Canada

Uploaded by: rpkolesnik

Aurora Test

Uploaded by: rpkolesnik

Comment:
Aurora Test
Location:
Alaska Fairbanks
GPS:
64.8436° N, 147.7231° W
Camera:
Nikon

Photo gallery mockup.

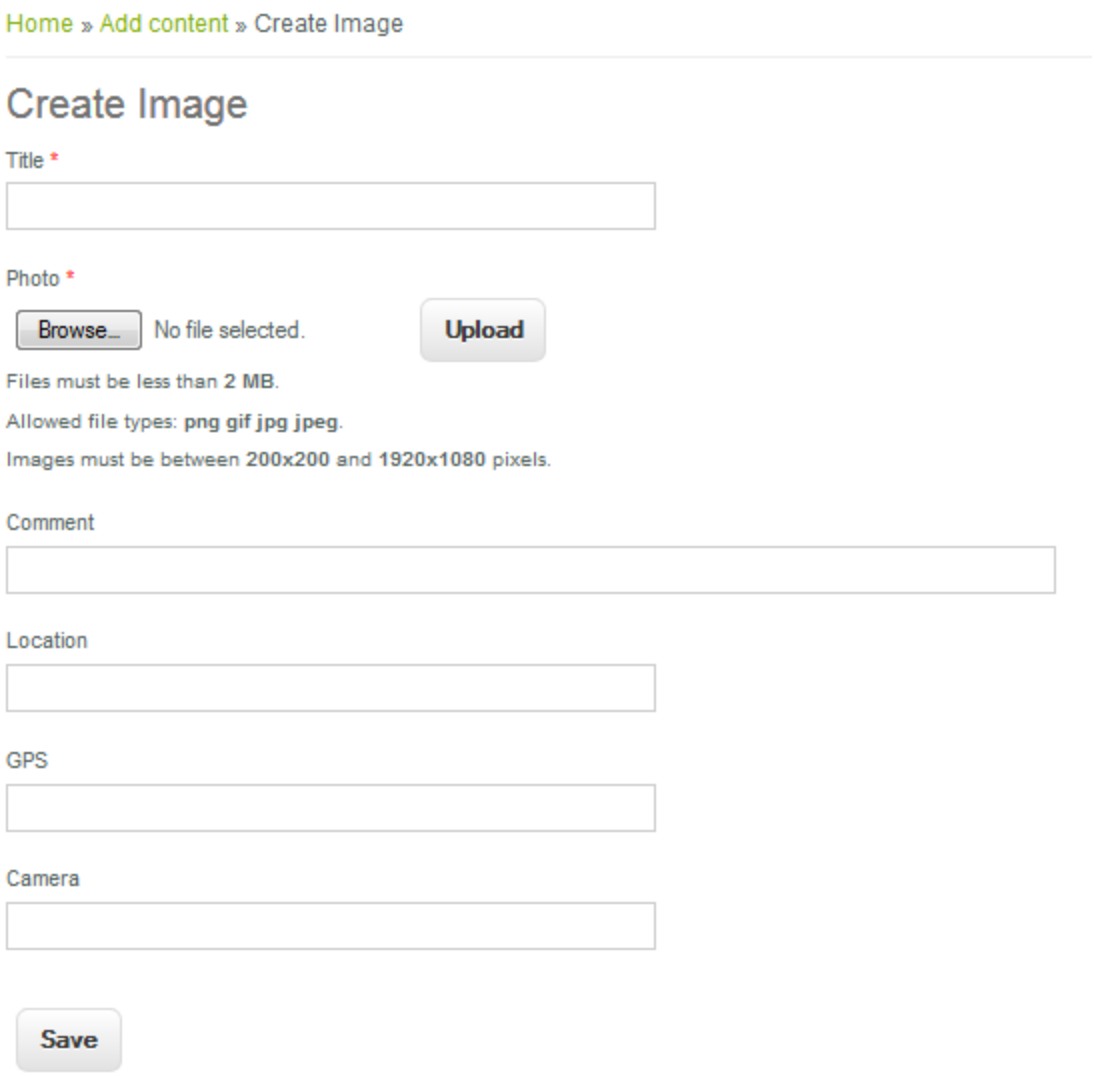
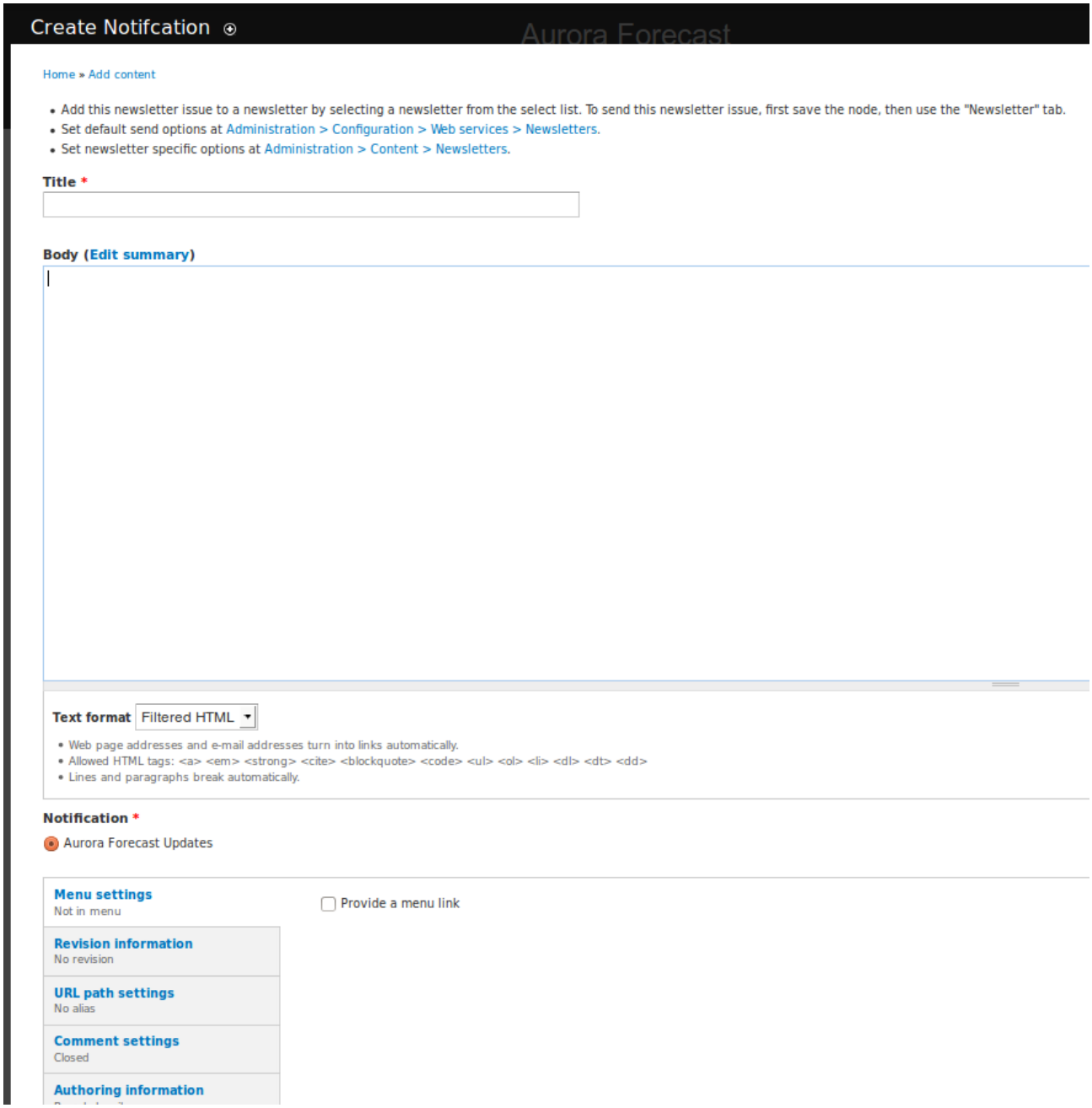
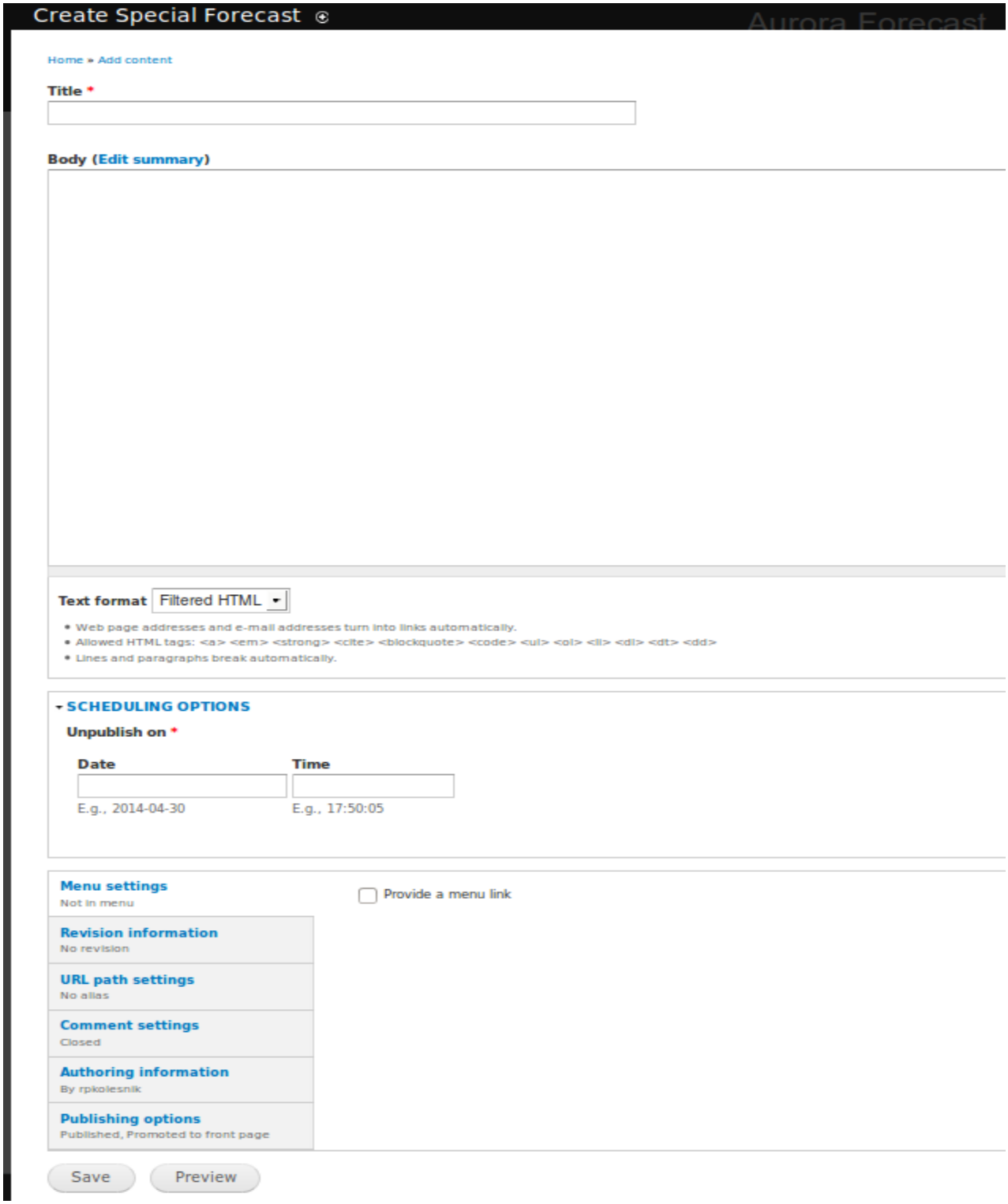


Image upload mockup.



Notification mockup.



Create special forecast mockup.