08

**Fall**

**Darcy Laycock | Nicholas Barret | Robert Gillman | Sam Winterton**

CITS4230 - Project Documentation

# Functionality and Design Decisions

## Functionality *provided* in final submission:

* Picture caching of the stories related to a feed.
* Caching stories.
* Summarizing stories.
* Like/Disliking stories.
* ‘Favoriting’ stories.
* Viewing favorites of other members.
* Commenting on stories.
* Viewing member profiles.
* Profile statistics.
* User Authentication.
* Search.
* Automatic Categorization.
* Search suggestions.
* Resizing of cached images.

## Functionality *not* *provided* in final submission:

* Complete test suite.
* HTML5 and CSS3 compliance.
* User respect or achievements.

## Weighty design decisions bearing on final functionality:

* Authentication subsystem
  + Using devise for the authentication subsystem was a valuable decision as it fast tracked much of the learning revolving around implementing an authentication system. Devise is also fast because it acts without loading the whole rails stack.
* HAML for creating views
  + HAML has significantly reduced our prototyping time by removing the redundancies when writing html.
* Extensive use of plugins
  + The use of significant plugins available in the rails community reduces the utility code required from the start of a project and offloads the maintenance to other developers.
* View hierarchy re-use
  + By creating generic aspects of the site such as footers and headers in separate views it helps maintain a DRY methodology.

# Review and Further Development

## Knowledge Development

* Teamwork and communication skills.
* RSS under rails.
* Feed fetching with e-tags for efficient updating.
* Pismo keyword extraction for categorization (as used on coder.io).

## Issues and Problems

* Query efficiency
  + For example the front index page has a query complexity of n+1 where under load the site would probably become unusable in its current state.
  + Because we chose to use a premade library for dealing with likes and dislikes this has already implemented this problem within it. Because we were trying to fast track development we chose to use it anyway.
* Unexpected time for implementation of aspects of site (underestimation of time required).
* Time management.
* Being attacked by zombies.

## Further Development

* ‘Ajaxify’ the site.
* Real-time comments.
* Respect or adding a game aspect for users.
* Profile integration with github or other code related sites (stack overflow).