RSS Feed Implementation Plan

Devon Compton, James DelVesco, Justyn Harriman,
Lucia Lu, Michelle Ross, and Jawwad Zakaria

October 31, 2014

1 Iteration 1

1.1 Use Cases

In iteration 1, we intend to implement front-end design and the core feed aggregator functionalities.

Feed creation and editing

- Add feed
- Search for feed
- Delete feed

Feed browsing

- Expand feed
- Expand different feed
- Expand post
- External website
- Sort chronlogically

Topic creation and editing

- Create topic
- Add feeds to topic
- Delete feeds from topic
- Rename topic
- Delete topic

1.2 Division of Labor

Tasks are divided across several high-level components:

- the database that keeps track of user accounts and feeds;
- the model that manipulates data in the database;
- the view that prints the HTML; and
- the controller that receive and processes input.

In week 1, Justyn and Lucia will be setting up the server with Django and the database with PostgreSQL, James and Devon will implement core model classes, and Michelle and Jawwad will work on the HTML view classes and to some degree the controller. In week 2, we will continue these tasks and start bridging our assigned components.

2 Iteration 2 Use Cases

In iteration 2, we plan to implement additional functionalities on top of our core feed reader. Specifically, we will implement:

Logs/Register

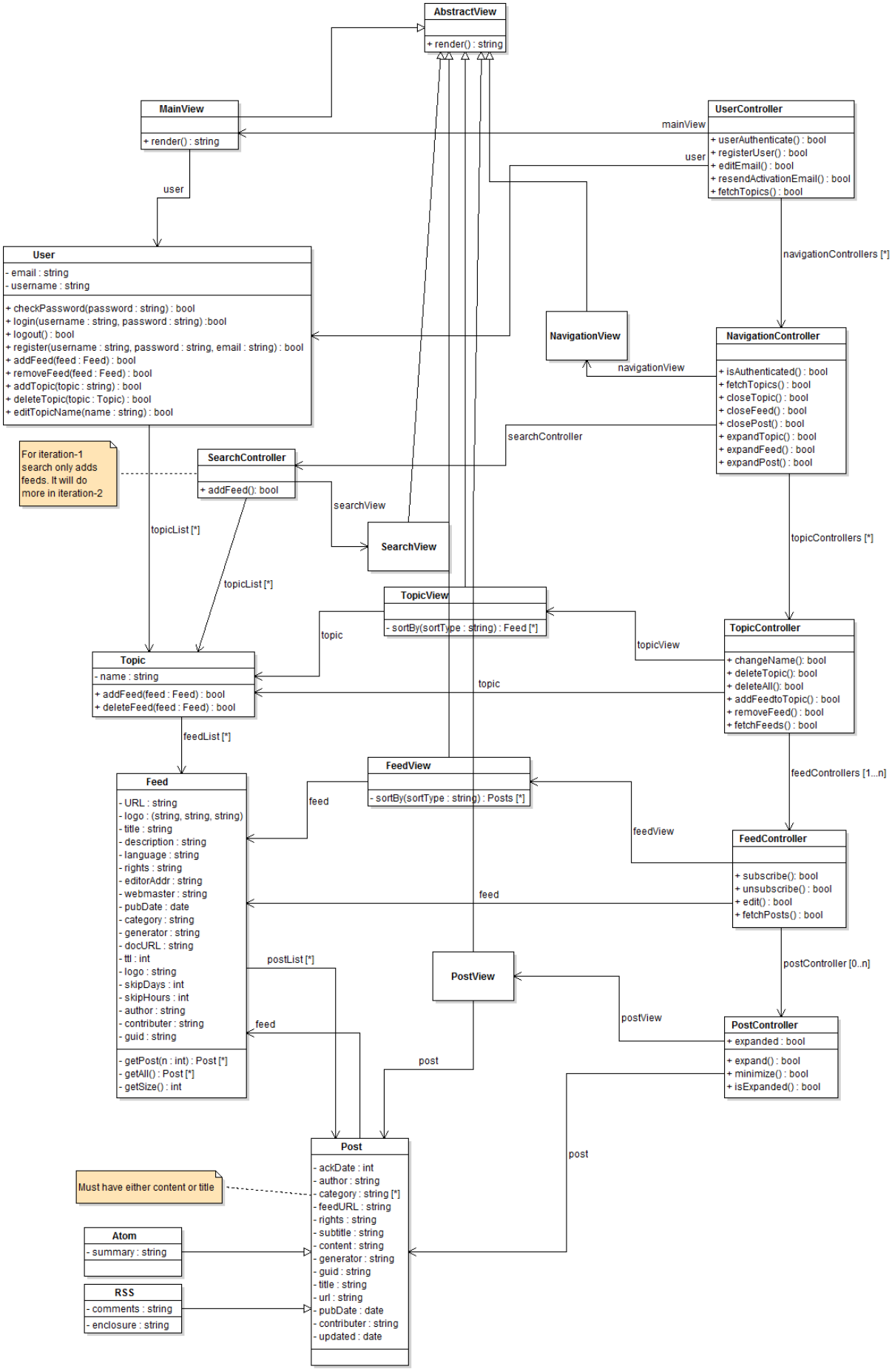
- Log-in
- Logout
- Register

Additional Functionalities

- measuring reading speed
- feed rating system
- generating feed list for a given time interval
- sorting posts
  - by length
  - by time
  - unread/read
- tracking browsing patterns
- user statistics page
- queued feeds
- time-dependent post deletion
- search by keyword
- mark posts read/unread

3 Iteration 1 Design

We plan to implement the RSS reader using a model-view-controller architecture.


AbstractView
+ render() : string
MainView
+ render() : string
UserController
+ userAuthenticate() : bool
+ registerUser() : bool
+ editEmail() : bool
+ resendActivationEmail() : bool
+ fetchTopics() : bool
mainView
user
user
User
- email : string
- username : string
+ checkPassword(password : string) : bool
+ login(username : string, password : string) :bool
+ logout() : bool
+ register(username : string, password : string, email : string) : bool
+ addFeed(feed : Feed) : bool
+ removeFeed(feed : Feed) : bool
+ addTopic(topic : string) : bool
+ deleteTopic(topic : Topic) : bool
+ editTopicName(name : string) : bool
navigationControllers [*]
NavigationView
NavigationController
+ isAuthenticated() : bool
+ fetchTopics() : bool
+ closeTopic() : bool
+ closeFeed() : bool
+ closePost() : bool
+ expandTopic() : bool
+ expandFeed() : bool
+ expandPost() : bool
navigationView
searchController
For iteration-1 search only adds feeds. It will do more in iteration-2
SearchController
+ addFeed(): bool
searchView
topicList [*]
SearchView
topicControllers [*]
topicList [*]
TopicView
- sortBy(sortType : string) : Feed [*]
topic
topicView
TopicController
+ changeName(): bool
+ deleteTopic(): bool
+ deleteAll(): bool
+ addFeedtoTopic() : bool
+ removeFeed() : bool
+ fetchFeeds() : bool
Topic
- name : string
+ addFeed(feed : Feed) : bool
+ deleteFeed(feed : Feed) : bool
topic
feedList [*]
FeedView
- sortBy(sortType : string) : Posts [*]
feedControllers [1...n]
Feed
- URL : string
- logo : (string, string, string)
- title : string
- description : string
- language : string
- rights : string
- editorAddr : string
- webmaster : string
- pubDate : date
- category : string
- generator : string
- docURL : string
- ttl : int
- logo : string
- skipDays : int
- skipHours : int
- author : string
- contributer : string
- guid : string
- getPost(n : int) : Post [*]
- getAll() : Post [*]
- getSize() : int
feed
feedView
FeedController
+ subscribe(): bool
+ unsubscribe() : bool
+ edit() : bool
+ fetchPosts() : bool
feed
postList [*]
PostView
postController [0..n]
feed
postView
PostController
+ expanded : bool
+ expand() : bool
+ minimize() : bool
+ isExpanded() : bool
post
post
Post
- ackDate : int
- author : string
- category : string [*]
- feedURL : string
- rights : string
- subtitle : string
- content : string
- generator : string
- guid : string
- title : string
- url : string
- pubDate : date
- contributer : string
- updated : date
Must have either content or title
Atom
- summary : string
RSS
- comments : string
- enclosure : string

4 Coordination Methods

1. For version control, we have created a Github organization and plan to use a git repository. https://github.com/CombustibleLemons
2. For discussing implementation and asking each other questions, we have a Discourse page. http://discourse.combustil lemons.bitnamiapp.com/
3. For meeting notes and other materials, we are using Google Drive.

We plan to meet after week 1 of iteration 1, and before the due date during week 2 of iteration 1.

5 Major Changes

We will use Python with the Django web framework. Django is implemented around the MVC design pattern, and will help us structure our classes.