Nanotwitter v0.4

Ashcloud: Andy Alexander, Mark Capobianco, Jinfeng Lin, Ruoyun Song

http://nanotwitter.herokuapp.com/
https://github.com/Ash-cloud/nanotwitter

Load Tests with Indexes:

Maintain Load:
1. Login and Loggedin root page load test. After login, we redirect to Loggedin root page to grab 100 tweets from timeline.
    http://ldr.io/1ywGq3c
2. Welcome page load test. Show latest 100 tweets.
    http://ldr.io/1aXbSwj
    http://ldr.io/1GUlfdY
3.unfollow
    http://ldr.io/1JQNSXO
4.follow
    http://ldr.io/1GUn0YG

Load Test summary - Implemented indexing, changed database design to minimize database calls when creating timelines, and switched server to Thin to address performance.

Noticed a benefit in performance for unlogged in root, follow and unfollow. Our login and loggedin root performance seems unchanged.

Significant Changes from v0.3:

1. Changed database: removed Tweet_User table

2. Shortened recommendations method from this:

```
#get all the users who aren't you
@other_users = User.where.not(id: user_id)
puts("number of other users is #{@other_users.size}")
#get all the people you follow
@followees = Follow_Service.followers(user_id)
@recommended = []
num_recommendations = 0;
while num_recommendations < 11 #want to get 10 random recommendations
        index = Random.rand(@other_users.size)
        potential_recommendation = @other_users[index]
```

```
followed = false
@followees.each do |followee|
        #if other_user is a followee of you, followed is true
        if potential_recommendation.id == followee.user_id
                followed = true
        end
        @recommended.each do |rec|
                if potential_recommendation.id == rec.id
                followed = true
                end
        end
end
#if you don't follow other_user yet, push to recommended
if followed == false
        @recommended.push(potential_recommendation)
        num_recommendations += 1
end
end
return @recommended
end
```

To this:

```
candidates = User.limit(10).order("RANDOM()")
```

3. Create tweet arrays that contain fields we need to display for tweet timelines (user name, user id, tweet text, & tweet time) from one database join rather than multiple individual database calls for each tweet shown to address performance.

4. Hide User Password

5. Perform add and remove of old tweets in logged in root timeline when follow and unfollow checked

6. Display time in words

7.Included ReadMe

8. Implemented Indexes
   User table (user_name, email)
   Tweet table (created_at, user_id)
   Follow table(user_id,follower)

9. Refactored Service into Tweet_Service, User_Service, and Follow Service

10.  Switched to Thin from WEBrick

11. Clean erbs, collect time function into API file, use template to reduce redundancy.

12. Fixed Tweet Box Indentation Bug