EPICS

1. Do Ipython notebooks for API
    a. Twitter (done) - Kevin
    b. Youtube - Garry
    c. Reddit/Yelp - Larry
    d. Facebook - Garry
    e. Hacker news - Neil
    f. RSS general - Igor?

2. RSS technology feeds
    a. Choose the correct technology feeds (spike) - Neil
    b. Parsing the RSS feeds - Igor
    c. Match sources to twitter account (like New York times RSS & Twitter account)
    d. ~~Create basic home page~~
    e. ~~Display any RSS feeds on UI~~
    f. Display correct feeds on UI (dependent on part a)
    g. Selenium test to ensure 10 stories on page

3. User Accounts (Larry/Kevin)
    a. Get tokens from twitter
    b. Create User Model in Django (store only twitter cred)
    c. Create Django admin interface
    d. Store user credentials in DB
    e. Add login button
    f. Add logout button
    g. Allow editing of users from Django admin
    h. Selenium test of user account login
    i. Selenium test of admin?

==

4. DB Model of Article (Garry / Neil)
    a. Create a django model for article
    b. Choose between redis and postgres to store articles(spike)

    c. Store articles in the database

5. Store Keywords
    a. Create a model for keywords
    b. Store keywords in database

6. NLP Extract keywords/topics
    a. Manual step to predefine list of keywords (spike) - Kevin
    b. Do NLP on articles and extract keywords - iPython Notebook?
    c. Associate keywords with articles
    d. Generate a list of articles based on keywords

==== UP to here is foundation =======

7. Notice how RSS feeds change (spike) - how often do we refresh, can we refresh individual sections

8. Twitter stream (depends on Epic 3)
    a. Update Twitter Notebook to try stream and search
    b. Decide on to sort tweets (spike)
    c. Use keywords from Epic 3
    d. Filter tweets based on keyword
    e. Sort tweets by relevancy
    f. Show top 3 tweets from first article
    g. Selenium test to check display of top 3 tweets

9. Twitter search (depends on Epic 3 (c,d)/4)
    a. Decide on to sort tweets (spike)
    b. Use keywords from Epic 3
    c. Filter tweets based on keyword
    d. Sort tweets by relevancy (LOS: and most recent)
    e. Show top 3 tweets from first article
    f. Selenium test to check display of top 3 tweets

10. User Topics (depends on Epic 3 (c,d))
    a. Create a model for user topics/preferences
    b. Allow users to choose topics from our predefined list
    c. Store chosen preferences for user
    d. Allow admin to update topics
    e. Create basic selenium test case for this

11. Store a Users' Clicked Articles
    a. Scoring system for ranking topics (spike)
    b. Add to user preferences to store topics related to clicked articles
    c. System to record clicks on articles

12. UI overlay initial setup
    a. Choose UI toolkit(spike)
    b. Create a list of anchor links beside RSS links (#)
    c. Create basic overlay page (template)
    d. Have a separate overlay for each article, activated with anchor link
    e. Create summary of article placeholder(lorem ipsum)

13. Summarisation of articles (Top section of overlay)
    a. Investigate reddit summarisation bots (spike)
    b. Create sample summaries from articles stored
    c. Add summary to django model for article
    d. Show on overlay (replace lorem ipsum)

14. Customise twitter feeds to be per article (Bottom left section in overlay)
    a. Store twitter archive (search) tweets - model
    b. Associate tweets with articles in database
    c. Show on overlay
15. Customise youtube feeds to be per article (Bottom right section in overlay)
    a. Search youtube API for article keywords
    b. Store youtube links
    c. Associate youtube links with articles in database
    d. Show on overlay