**STEP Pod Feature List Doc Template**

Authors: <[Danya Gao](mailto:danyagao@google.com), [Ankita Mitra](mailto:mitraan@google.com), [Adrian Manhey](mailto:amanhey@google.com)>

Used to create timeline and split workload

Feedback from 06/17/2020:

* focus on a couple things that we want to focus matching on
* need to find a way to do offline processing (e.g. tagging, labeling) so that the results are ready before the request arrives.
  + have a certain amount of books pre-processed and match with those
  + processing independent of the user request; can do this in the cloud with compute instances

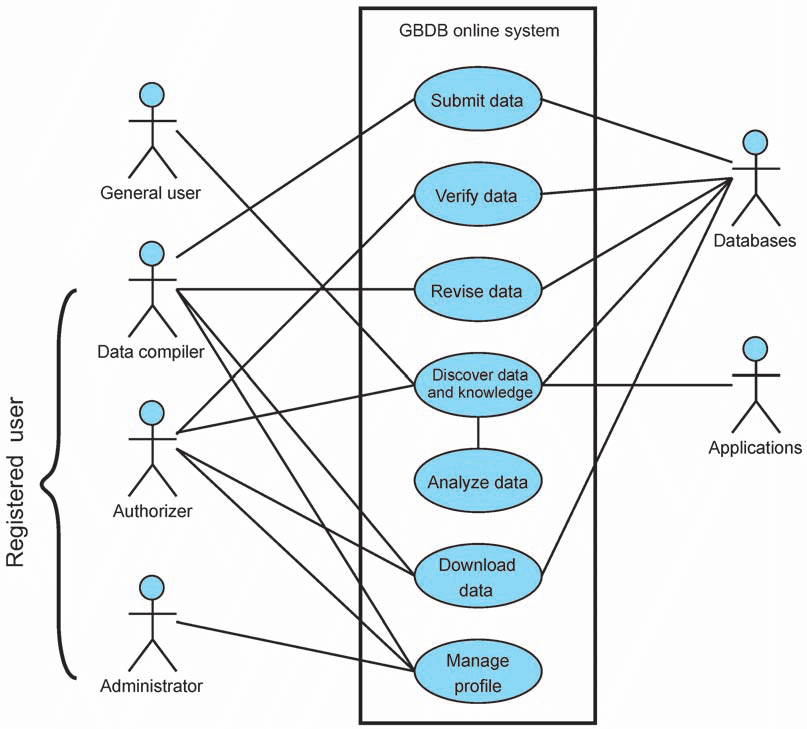
-authentication, books, Youtube, blobstore image upload, google cloud vision, charts, sentiment analysis(?), google news

Notes from Ankita’s brain:

* Pre-Processing Data:
  + Books API gives us ~1 paragraph short description
  + Goodreads dataset has SEVERAL long reviews
  + Get all the Sentiment Analysis Values
  + Make a CSV file
    - Title
    - SA for books API description
    - {SAs for goodreads dataset reviews}
    - Normalize SA to a single value representation

Core Outline

* P0 (PROTOTYPE, week 6)
  + Books API
  + Home page with all books, and be able to loading different book pages
    - Start with 20 books
* P1 (MVP, week 8)
  + Comments
  + Authentication
  + Sentiment Analysis (possibly using the “common words” part of google books)
    - Entities rather than sentiment
    - Helps recommend books, sad book = sad background
* P2 (Week 9 - end)
  + Blobstore & image analysis (if possible)
  + Youtube API
  + Google News
  + Charts API

List the independent work items/features we want (e.g. functionality, pages)

* We’ll need a basic html page that displays the books (homepage; 20 books)
  + How are we going to choose the books that appear on the homepage. Same for everyone?
    - (time permitting)
  + Expand to contain query to search for books later
* Create reusable html page for each book
* Display comments/reviews for each book
* Store the books that we’ve processed in a csv file
  + Information to process (title, author, description, reviews) comes from Books API and Goodreads dataset
    - Pulls information from Books API
    - Pulls information from Goodreads dataset
    - Places in a csv file
  + Admin console: admin can 1) use the interface of the deployed site and input the title of the book which calls the servlet to call the apis and fetch the data or 2) run the servlet locally (if takes more than 30 secs to get book data)
* Recommend other books offline, can store in a map/graphs (closer books more related)
  + =Take csv file and get one book
  + Analyze the book’s associated info with [Entity Analysis](https://cloud.google.com/natural-language/docs/basics#entity_analysis)
    - <https://cloud.google.com/natural-language/docs/basics#entity_sentiment_analysis_responses>
    - Create groupings from the Books API genre tag
      * Time permitting, recommend books by entity analysis within the genre
    - Create (separate) groups to recommend by sentiment analysis
      * Positive - negative
    - Sample output doc (when working)
* User profile: need to identify the user and be able to get the comments that user has made (shows that which the user owns)
* Authentication -- ability to log in

[saur@: Notes]

* Design Datastore
  + What tables, columns, information going to be stored.
    - Entities:
      * Book
        + Fields (as of 06/26): ["title", "genre", "categories", "author", "language", "description", "infoLink", "pageCount", "publishedDate", "publisher", "maturityRating"]
      * User (name, email, etc)
      * Comment/Review(name, time, body)
  + Datastore access layer
* **P0**: Generate Book information - **Ankita**
  + For offline processing - csv format?
  + Calling books api
  + Admin console/webpage to upload the book information.
* **P0**: Home Page with all books from the datastore - **Adrian**
  + Each book with hyperlink the display page
  + Defining the datastore (columns)
* **P0**: Display page for a given book - **Danya**
  + Title, Author, Comments, Reviews
  + Comments/Reviews page (in case reviews/comment don’t fit on a page).
* Widget:
  + Recommendations
    - What are the inputs? Current book being displayed
  + Can be independently integrated into many pages
    - Designed as an independent HTML page that can be embedded (like Spotify iframe)
* Page design
  + To have common template across the webapp (reusable page)
* Lower priority:
  + User authentication
  + User profile page:
    - See comments/reviews by a given user

Adrian

* Datastore

Ankita

* Books api - look at the attributes
* Pre-processing the entity analysis response to make it usable.

Danya

* Entity Sentiment Analysis