Design Document:

Outline:

      Sentiment analysis coupled with post generation using data from reddit posts of two subreddits. The user types in two subreddits and chooses a sentiment from a dropdown menu. The website pulls posts from the subreddit and passes them to a sentiment analysis API. Subsequently a post is created that incorporates emotion expressing keywords of the two personalities.

      Go ahead and confuse yourself with the mixed qualities of /r/jokes with /r/antijokes, /r/theonion and /r/nottheonion, or /r/the_donald and /r/pyongyang.

Planned Components:
🟨 **Base functionality**
🟧 **+** **Stretch goals**
🟥 **Extra stretchy goals**

🟨 **Python:**
    🟨 **Accessing Data from APIs:**
        🟨 Reddit API
        🟨 IBM BlueMix API (Sentiment Analysis)
    🟨 **sentiment.py**: functions for interacting with the BlueMix API
    🟨 **readit.py**: functions for interacting with the Reddit API
    🟨 **wordmash.py**: functions for generating posts/comments using sentiment.py and tweet.py
    🟨 **app.py**: define routes and associated templates
    🟧 **+** **sql.py**: functions for interacting with the SQL database

🟧 **+** **SQL:**
    🟧 **+** Store past mashes created by users. Allow browsing of this data on the homepage.

Database Schema

Subreddit1 TEXT	**Subreddit2** TEXT	**SubredditMash** TEXT

🟨 **Frontend:**
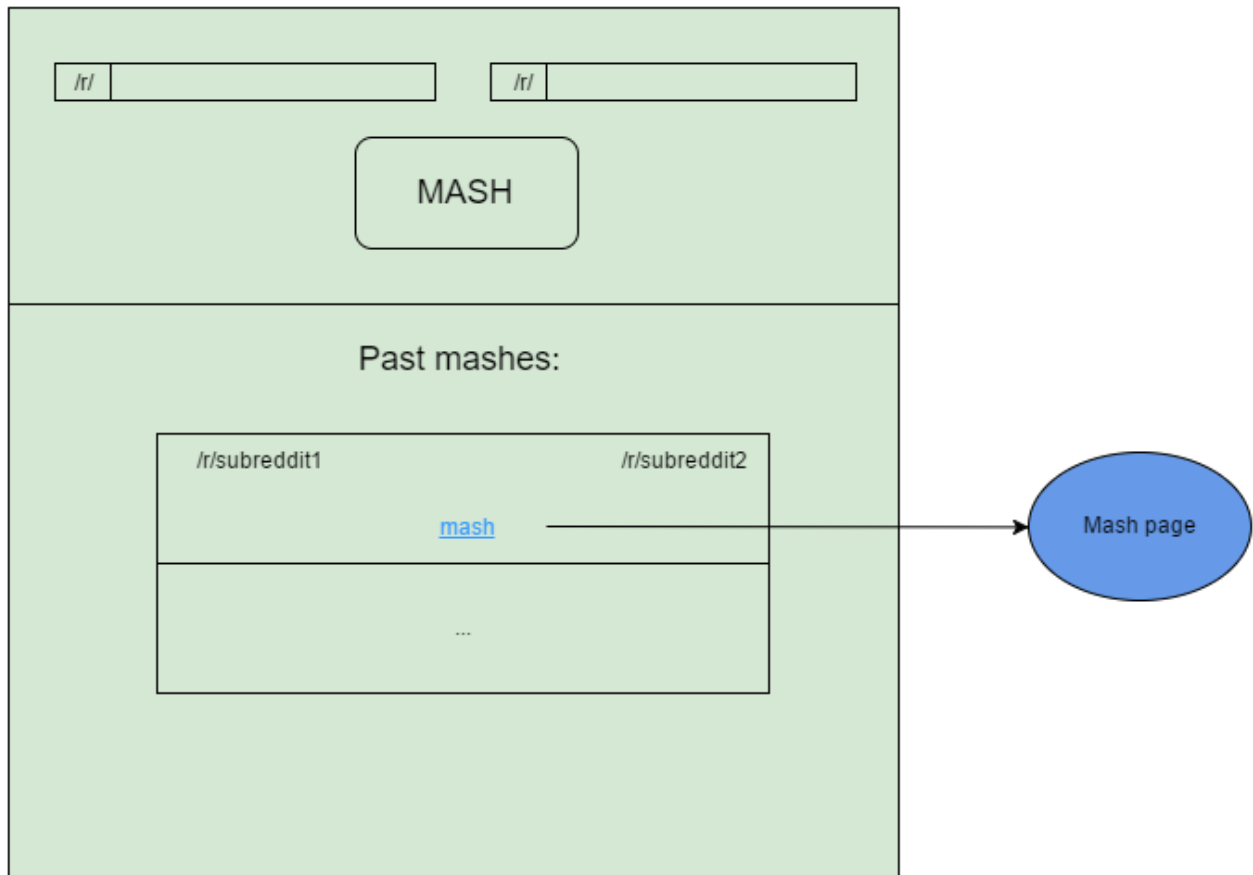   🟨 **Routes**
      - 🟨 /
        - 🟨 Place for users to generate madlibs
        - 🟧 **+** Has inputs for options, like what words to look for
        - 🟧 **+** Homepage: display recent posts and queries
        - 🟧 **+** Acquires data from SQL database.
      - 🟥 /mashed/<mashid>
        - 🟥 Post is displayed

Design with Bootstrap. There are basically two pages, so the site map is simple:

Timeline of Completion:

   12/6 - Simple folder structure, skeletons of app.py and python files, bootstrap files added

   12/7 - Bootstrap website visible, integrated with app.py, API research finalized

   12/8 - APIs working, can retrieve data, begin integration of Flask with frontend and python

   12/9 - Begin work on mashing

   12/10 - Implementation of major elements, making bootstrap look good

   12/11 - Implementation of reach features

   12/12 - Final morning bug testing

Division of Labor:
Gabriel - Word mashing
Kenneth - APIs
Wilson - app.py and SQL part
Karol - Project Manager and front-end

Component Map

browser
Interprets and displays content.

user
Expendable component.

bootstrap
Formats HTML.

HTML output

app.py
Forwards mash to user, manages redirection and template rendering.

backend masher
Receives data and combines content from two subreddits, creating a mash.

backend data retreiver
Retrieves content from APIs.

reddit API
Provides content.