Team Name: azrael (Mohtasim Howlader (PM), Jack Lu, Jason Tung)
SoftDev1 pd8
P #02
2019-01-08

Project Name: Testlet

Abstract:
Teslet is a Quizlet inspired project which revolves around duplicating Quizlet's core functionality: logging in, creating, saving, and viewing sets. Team azrael will try to also implement various ways to expedite the process of set creation and reviewing sets, most notably autofill definitions and enabled pronunciation on terms/definitions. Using javascript, we will try to update a container in while reviewing sets to imitate flipping an index card.
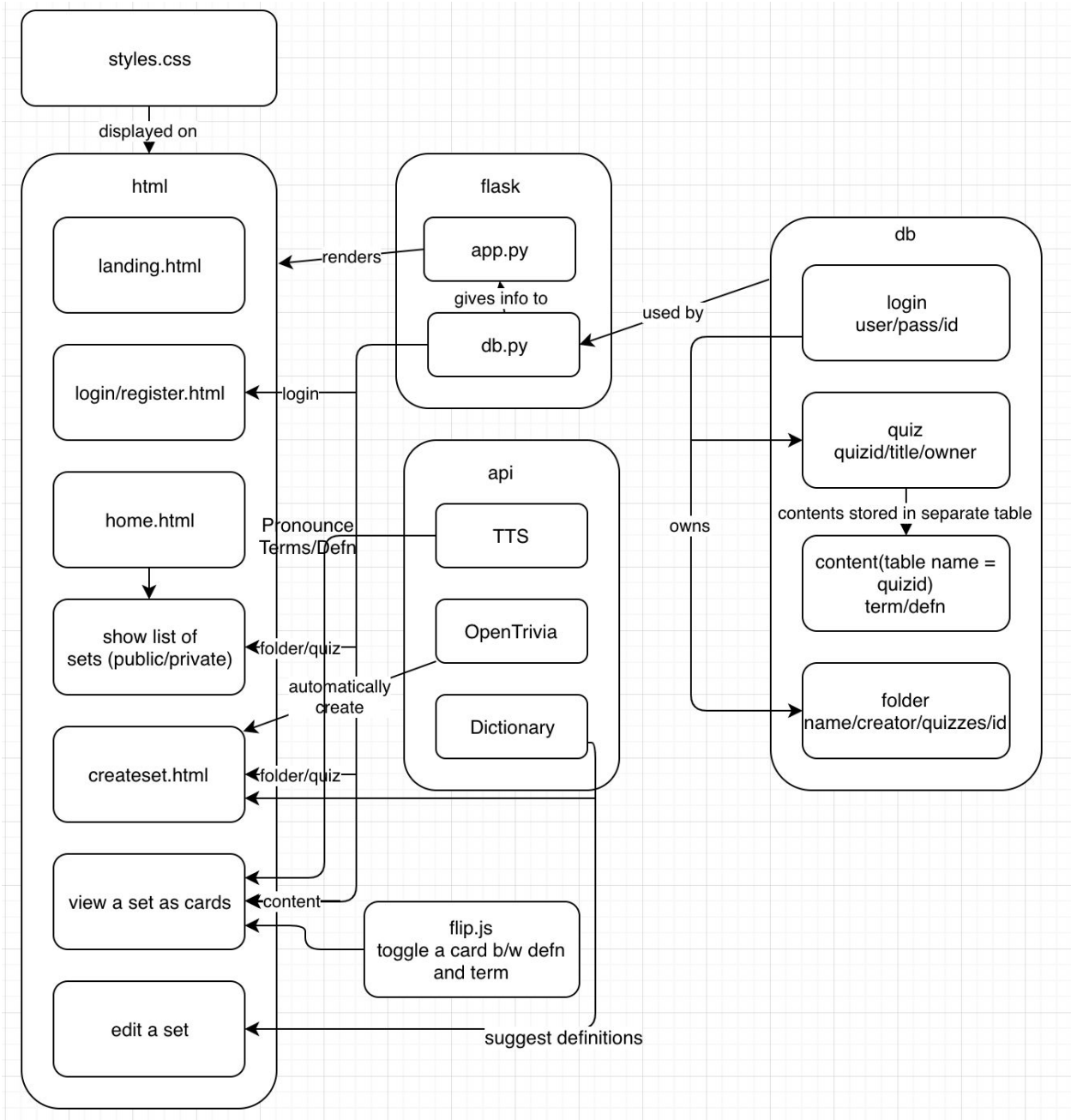
Front End Framework: Bootstrap
- All team members have more experience developing with Bootstrap than Foundation (including in previous projects)
- We prefer Bootstrap's default design aesthetic and structure over Foundation's. Specifically, Bootstrap's grid system is simpler to use and implement (we don't want to deal with gutters) for our project.
- Has a much larger collection and variety of templates than Foundation. And its templates are typically more fleshed out and "complete" , so we don't have to spend as much time on the front end and could devote more to the back end.
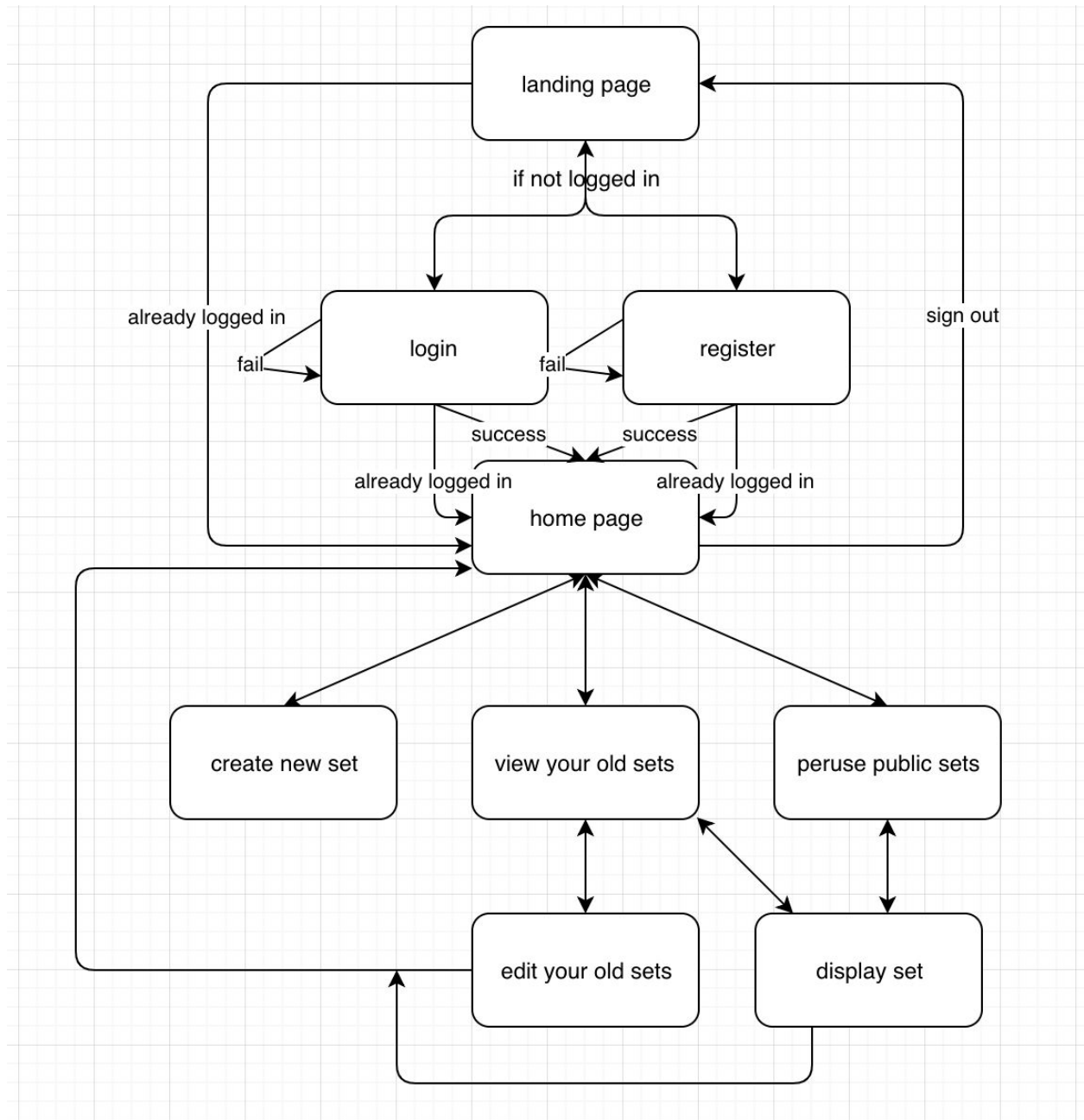
APIs:
1. MSAzureTTS API - pronounce terms/definitions from sets
2. Oxford Dictionaries API - generate suggested definitions for terms when creating sets
3. OpenTriviaDB API - generate trivia questions to make sets automatically
4. (stretch goal) OAUTH2 - lets other companies handle the login system

Component Map

styles.css

— displayed on →

html

landing.html

← renders — **app.py**

login/register.html ← login —

gives info to

home.html

db.py ← used by —

show list of sets (public/private) ← folder/quiz —

Pronounce Terms/Defn

createset.html ← folder/quiz —

automatically create

api

TTS

OpenTrivia

Dictionary

view a set as cards ← content —

flip.js toggle a card b/w defn and term

edit a set ←

suggest definitions

flask

db

login user/pass/id

quiz quizid/title/owner

contents stored in separate table

content(table name = quizid) term/defn

owns

folder name/creator/quizzes/id

Site Map

landing page

if not logged in

already logged in

sign out

login

fail

register

fail

success

success

already logged in

home page

already logged in

create new set

view your old sets

peruse public sets

edit your old sets

display set

Database Schema

The tables are formatted as follow:
** TABLE NAME**
COL-NAME | VALUE-TYPE

User	
username (key)	string
pw	string

Quiz	
quiz id (key)	integer
quiz name	string
quiz owner	string

Content(title = quizid)	
Definitions	String
Terms	String

Folders	
Folder name	string
Creator name	string
List of quizzes	string
Folder ID (key)	integer
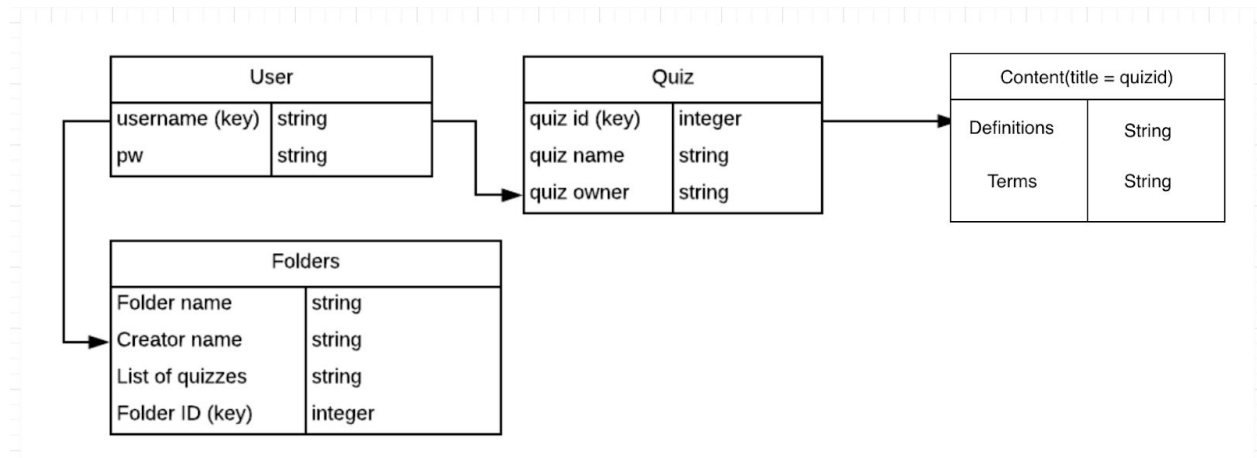
db

- User
    - Table of username/pw combos
- Quiz
    - Table of every quizid, its name, and its owner
- Content
    - Every quizid will have a content table with a corresponding name
    - Contains defn/term pairs
- Folders
    - Table of all lists with owner, id, name
    - List of quizzes will be in a parsable csv format
    - Users can organize their quizzes into folders, making them easier to group into categories like Chemistry vs. Spanish

TO-DO LIST
- search function
- search for quizzes/folder/class by name or by creator
- dictionary API - option to automatically define entered terms
- User can create quizzes and save others quizzes
- Quiz: each quiz has tags to search by
- Folders: group quizzes into folders

- Obtain quizzes from APIs to display on website
- User quiz history and stats
- Text to speech for flashcards and quizzes

Reaches:
- translations
- premade quizzes
- Class: contains sets and members
- Oauth2

Roles

Project Manager - Mohtasim Howlader
- Make sure the group is consistently working on the project
- Update and maintain devlog
- Update and maintain design document
- Work on front end HTML and Javascript with Bootstrap
- Handle coding tasks as necessary for back end and front end
- Work on user quiz history and stats
- Test website for errors, bugs, crashes, and instability (API requests running out….)

Jason "Snitch" Tung
- API work
    - Implement all APIs and procure API keys
- Flask routing
- Login setup
- Templates
- JS
- Databases
    - Writes functions to more easily work with DB

Jack Lu
- Database work
    - queries to retrieve and upload info to DB
- Javascript
- search function
- API work

Timeline

Jan9 - finish templates and a little bit of css

Jan10 - login system done

Jan11 - finished all routing with flask

Jan12 - all db components done and ready for implementation

Jan13 - implemented db to display on page

Jan13 - add js "card flipping"
Jan13 - finished css
Jan14 - add oauth2/other stretch goals
Jan15 - finish stress testing/debugging