Spring Term Final Project

“GOOGLE KEEP”

TEAM MiLKShake

Kristin Lin, Levi Olevsky, Sonal Parab, Masha Zorin

DESCRIPTION

A recreation of Google Keep, an online post-it note taking application that allows a user to create and edit notes by typing, drawing, uploading an image or other things. The notes can be labeled or color coded the way the user chooses for better organization. Also, the notes can be arranged/rearranged on the dashboard.

COMPONENTS

HTML files

See 'Routes’ section for details

CSS Stylesheets

Bootstrap’s as well as our own

app.db

Database to hold information on users and notes taken. See ‘DB Schema’ section for details.

database.py

Functions that will interact with the database from the Flask app. Functions may include add_user, add_note, find_note, find_user, set_note_color etc.

app.py

Flask framework to connect the backend and the frontend.

notes.js

Helps user interact with notes present on the home page. Will use AJAX calls to update user’s choices in the database. Will use d3 with svg to manage notes.

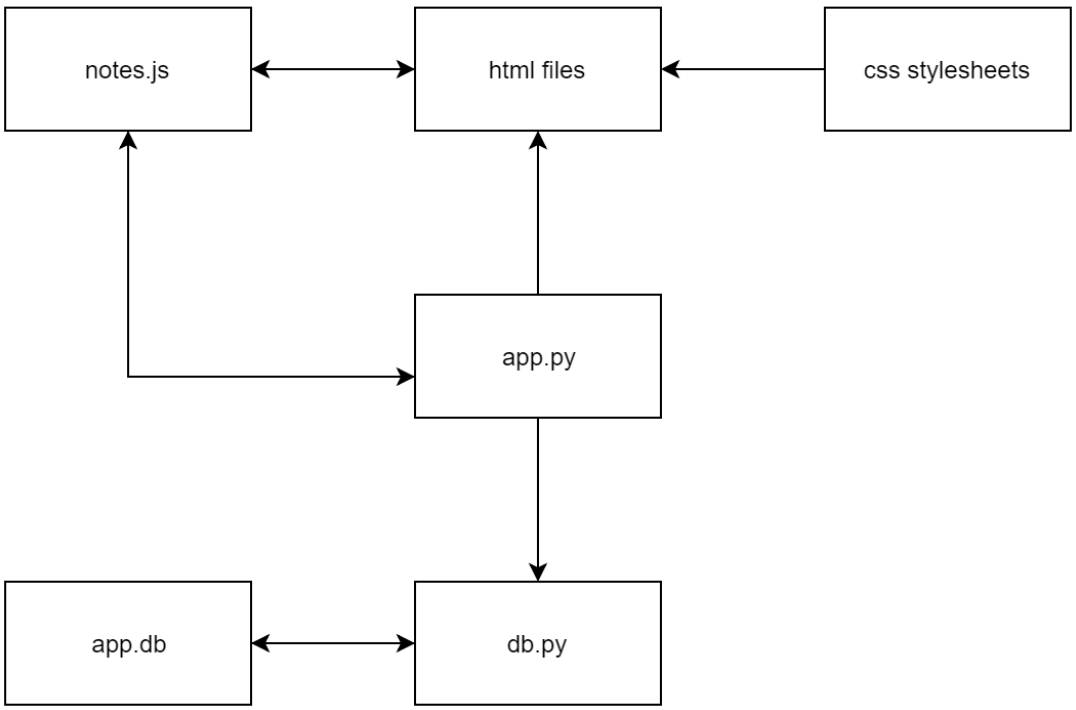
Possible Actions that Functions Must Cover:

- Create note
- Change note to checklist
- Change color of note
- Add images
- Drag/reorder notes
- Pin notes to top
- Archive notes (sends to archive page)
- Delete notes
- Select notes
- Label notes

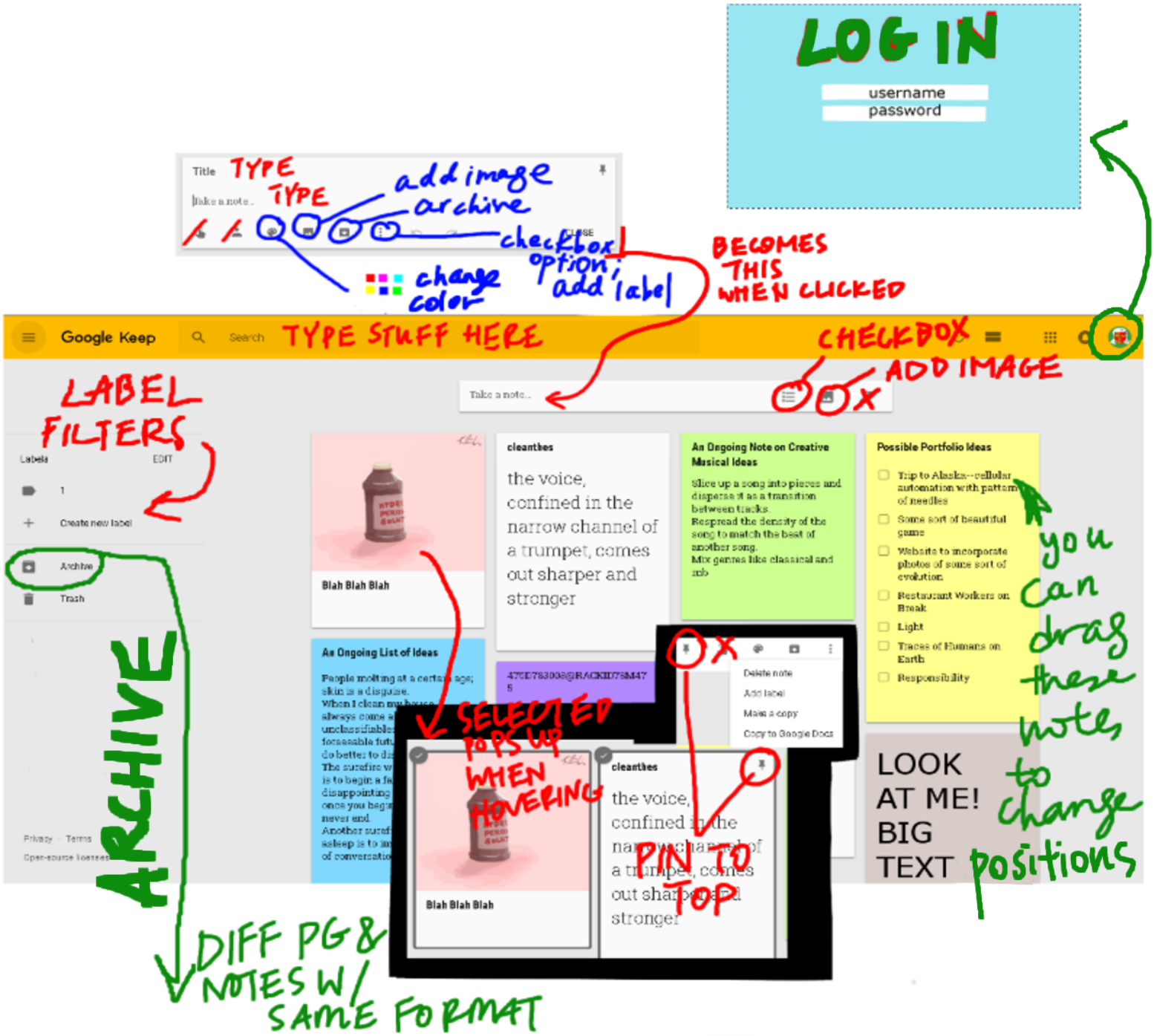
ROUTES

- **/**
  The front page, which will display a description of the website, along with some basic information you need to know and maybe a sample arrangement of notes.
- **/login**
  Login page for username and password input. Contains button to go to /signup.
- **/signup**
  Page to create an account, with username, password, and confirm password. Contains button to go to /login.
- **/home**
  Dashboard with all the notes
- **/search**
  Filters notes. Will call database function to find notes containing keyword in title or content
- **/archive**
  Dashboard with the notes that were archived

COMPONENT MAP



SITEMAP


LOG IN
username
password
Title
TYPE
TYPE
add image
archive
checkbox option; add label
change color
BECOMES THIS WHEN CLICKED
Google Keep
Search
TYPE STUFF HERE
CHECKBOX
ADD IMAGE
Take a note...
LABEL FILTERS
Labels
EDIT
Create new label
Archive
Trash
ARCHIVE
DIFF PG & NOTES W/ SAME FORMAT
Blah Blah Blah
cleanthes
the voice, confined in the narrow channel of a trumpet, comes out sharper and stronger
An Ongoing Note on Creative Musical Ideas
Slice up a song into pieces and disperse it as a transition between tracks.
Respread the density of the song to match the beat of another song.
Mix genres like classical and rnb
Possible Portfolio Ideas
Trip to Alaska--cellular automation with pattern of needles
Some sort of beautiful game
Website to incorporate photos of some sort of evolution
Restaurant Workers on Break
Light
Traces of Humans on Earth
Responsibility
An Ongoing List of Ideas
People molting at a certain age; skin is a disguise.
Delete note
Add label
Make a copy
Copy to Google Docs
SELECTED POPS UP WHEN HOVERING
PIN TO TOP
LOOK AT ME! BIG TEXT
you can drag these notes to change positions

DATABASE SCHEMA

users table

The users table will store the user's username, password.

notes table

The notes table will store the username, note id, note type (note, list), note content (list items separated by backslash, image file separated from message by backslash), whether it is pinned or not, order id (keeps track of what order notes are displayed in), color, whether is is archived or not, labels

labels table

The labels table will store the label name and note id (to keep track of labels of notes, and all labels in general)

note table

The note table will store the content of a note with type “note”. It will store content, image (can be null), and note id.

list table

The list table will store the content of a note with type “list”. It will store list item, item order, whether it is checked off, image (can by null), and note id.

reminders table

The reminders table will store the note id, the date, time, and repeats (once, daily, weekly, monthly, annually).

users	**notes**	**labels**	**note**	**list**	**reminders**
user *PRIMARY KEY (string)	note_id *PRIMARY KEY (int)	label_name (string)	note_id (int)	note_id (int)	note_id (int)
pass *encrypted (string)	user (string)	note_id (int)	content (string)	item_content (string)	date (string)
	note_type (string)		image (string - the link to image)	item_order (int)	time (string)
	pinned (bool)			checked (bool)	repeat (string)
	order_id (int)			image (string - the link to image)	

	color (string)				
	archived (bool)				

ROLES:

- Project Manager & CSS: Kristin
- Database: Masha
- Flask App, HTML Templates & JS: Sonal
- More JS & Ajax: Levi\

TENTATIVE GUIDELINE

First week:

- DB tables created. DB functions for user login half way done.
- Bare bones of Flask app routes connected to HTML
- General background CSS completed
- Research how to do AJAX

Second week:

- Essential DB functions for user login completed
- DB functions for notes, note, and list nearly finished (wiggle room for debugging)
- Flask working on connecting DB to HTML to JS
- JS able to create notes, pin notes, archive notes, delete notes
- CSS figured out how to place notes in tile format

Third week:

- DB finished the labels functions; maybe start on reminders (depending on status of other teammates
- Finished connecting all DB functions to JS
- JS able to change note to checklist, change color of note, add images, label notes
- JS able to send edits user made via AJAX to Flask to edit database info

Last week:

- Debug and fine-tune all of the above if unable to stick to schedule
- Attempt reaches (listed below)

REACHES:

- Google Authentication instead of our login
- Add drawing
- Reminders
- Storing custom images the user wants to upload