Always in School Squad--Sharon Lin, Asher Lasday, Sachal Malick, Felix Rieg-Baumhauer
SoftDev PD 8
HW #10: Da Art of Storytellin'
Due F 2016-10-28, 8:00am, EST.

Introduction

Purpose

Create a collaborative storytelling game/website.

Definitions/Constraints

- Users will have to register to use the site.
- Logged in users can either start a new story or add to an existing story.
- When adding to a story, users are shown only the latest update to the story, not the whole thing.
- A user can then add some amount of text to the story.
- Once a user has added to a story, they cannot add to it again.
- When creating a new story, users get to start with any amount of text and give the story a title.
- Logged in users will be able to read any story they have contributed to on their homepage (the landing page after login)

Design Overview

Site Components
Login/Logout
Registration
Start story (title, initial text)
Add to story (has added?)
Display stories

Site Pages
index.html - Homepage
auth.html - Login/Registration
feed.html - Feed of stories to which the user has not contributed
edit.html - Adding new story/Adding to story

Python functions
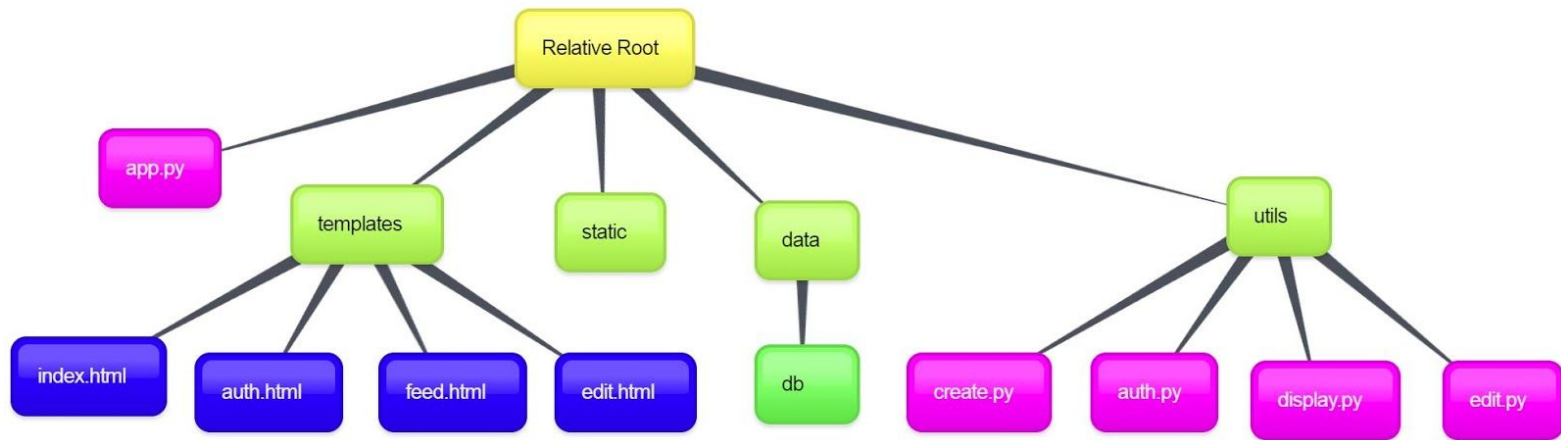auth.py : authentication for user signin/registration
create.py : adding stories to database
display.py : displaying feed with stories to contribute to / stories that have been completed

edit.py : adding a new story or editing a story on the feed

Diagrams
File Structure

Sequence Diagram/Site Map

Data Design

Description:

- story.db: holds every story's ID and information along with its most recent edit, this allows for quick serving of information, as we only call for the full story when we need it
- full_story.db: holds the actual text of any story along with its unique ID
- users.db: holds users' info

story.db

story_id INT	story_title TEXT	most_recent TEXT	number_edits INT	time_since INT	timestamp INT
000	"Doggy"	"The doggy went home."	2	0	5000000000
256	"Sharon"	"Sharon went home."	3	100000	4999999999

full_story.db

story_id INT	story_title TEXT	full_text TEXT
000	"Doggy"	"I got a doggy and the doggy went home.
256	"Sharon"	"Sharon went to a tech conference and then Sharon went home."

users.db

user_id INT	username TEXT	password (hashed) TEXT	email TEXT	story_ids BLOB
000	sharonlin	rejhi34t8	sharonlin@hotmail.org	{000,001,002}
371	sachalthemachal	i824378fh	smalick@earthlink.gov	{256}

Tasks

Project Manager (Sharon): makes sure the rest of the group is on task

Front-end (Felix): Creating views, displaying stories (HTML/Jinja2), implement the login/registration system

Database management (Asher): Creating python functions for managing db (story and users) (SQLite3)

Back-end (Sachal): link the functions that deal with databases to the front end, get the right information to be presented from the backend in the appropriate part of the front end

Quality testing (all group members may test the product): try to break it, different user logins, story components, etc.

Pages We Need:
- Register
  - Writes out your information to db (users)
- Log in
  - Redirects from the Home page if not logged in, must be logged in in order to access other parts
- Home
  - Has stories you've added to
  - The stories are presented in their entirety, along with title
  - Has buttons/links, (ie logout, make a story, add to a story, perhaps view stories that the user has not seen)
- Stories you haven't added to
  - This is where you go if you want to add something to a story
  - Will look similar to Home, but users will only see the last addition
- Create a story
  - Text boxes for name of story, text, will write out to db (story)

Additional Features
- Multiple users editing a story - story auto updates once a user pushes, adds to story in order of latest timestamp

Future Features:
- Sorting stories in feed.html by number of edits, most recent edit
- Liking, wowing, sadding, thankfuling, adding images, relationship status
- Venmo information (sending kudos for stories)
- Sorting stories by author, etc.
- Tags for stories, auto-tagging (genres, themes)
- Searching for certain stories to add to