POPEYES: Lauren Lee, Ian Jiang, Vivian Teo
softdev
p00 : Story Creation
2022-10-31
time spent:
target ship date: 2022-11-15

Scenario One: Your team has been contracted to create a collaborative storytelling game/website.

- 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).

Scenario 1:

Components:
- app.py – define routes of pages
- login.html – user can login or register
- db.py – python file for writing SQL and organizing the databases
- users.db - table storing user information
- edits.db - table storing contents of each story
- home.html - user can choose to create a new story or search for a story to edit.
- search.html - results page displaying story links after searching for a story on the home page

- stories.html - template that’s able to hold different topics/stories
- edit.html - users can edit old or write new stories on this page

How each component relates to each other:

- app.py - host the web pages locally.
- login.html - use db.py to check if user has an account stored in database
- home.html - displays all the stories a user has contributed to. There’s a search bar to search for any story as well.
- search.html - displays results after user inputs title into form submission on the home page.
- db.py – adds information into databases like users.db and edits.db
- edits.db – contains info on stories user edits after db.py is run
- users.db - have individual user information like their logins that will be checked when the user logs in or registers.
- edit.html - send information to db.py to be put in edits.db and users.db (list of stories a certain user contributes to)
- stories.html - get information from db.py to display the stories

Database Organization:

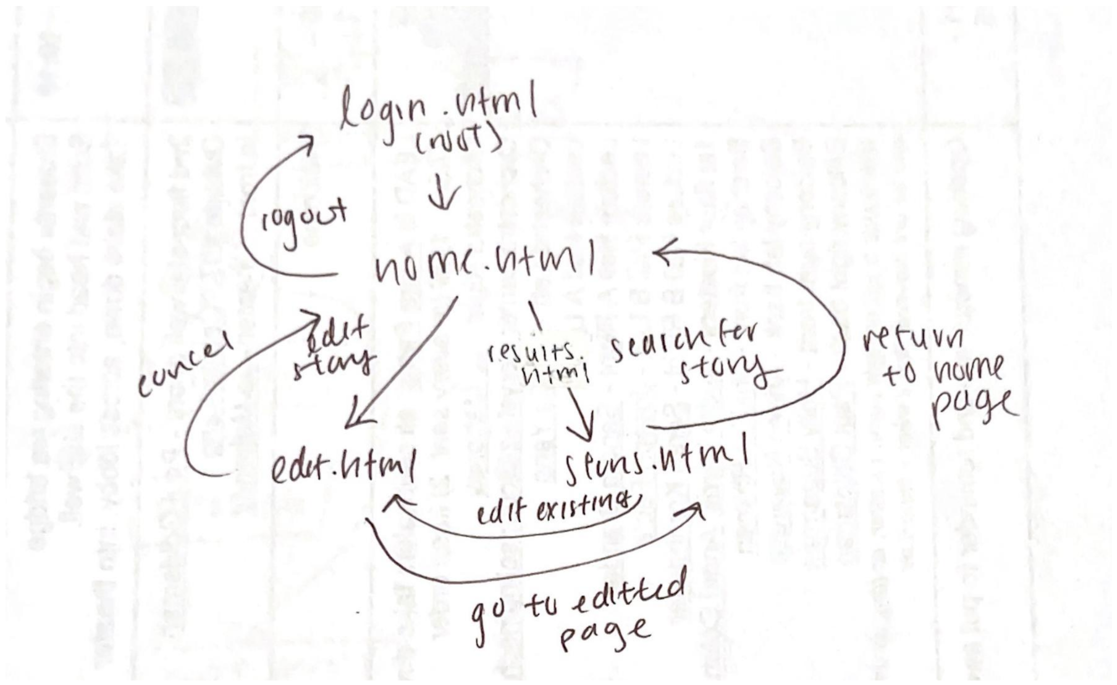
- users.db: user id | username | password | list of story ids user has contributed to | currently editing
- edits.db: story id | story title | content | time last edited | latest change

Site map for front end:

- /
    - login page - login.html
    - Form submission for logging in or registering
    - Registering creates new data in the table users.db
- /home

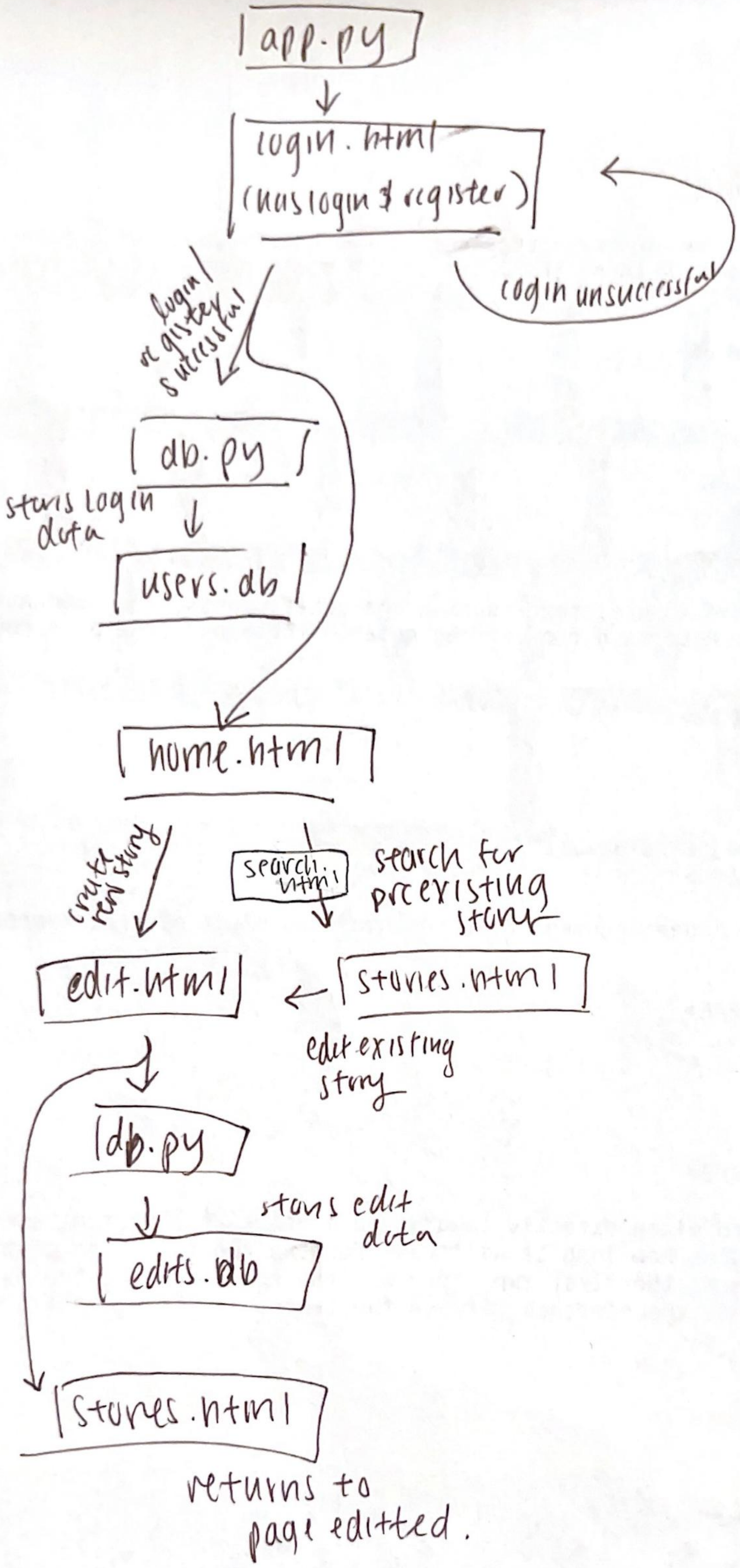
  - Once logged in, the home page is displayed (home.html)
  - Home page consists of all the stories the user has contributed to, scroll-based format
  - Able to search for a story via a form; submitting form takes user to the /results route
  - Option to create new story (button), which takes user to the /edit route
- /results
  - Page that displays search results after user submits inputs in the /home route
  - Hyperlinks will appear for specific stories. Clicking on a link will take the user to the /[title] route.

- /[title]
  - View story named [title] (this uses the stories.html template) after clicking on a story link in the /home route
  - There will be an option to edit story (button), which takes the user to the /edit route. Data regarding which story is being edited is stored in users.db (currently editing).
- /edit
  - Page where user can edit and submit story. If a user has already contributed to a story, they cannot contribute again (checked by looking at the list of story ids the user has contributed to in users.db). The data about what time the user submitted as well as the content is stored in the edits.db, and the story id is added to the list in users.db if not already in. User cannot delete content, only add.



Breakdown of tasks:

- Lauren: edits.html, stories.html, storing data into edits.db
- Ian: app.py, db.py, search.html
- Vivian: login.html, home.html, storing data into users.db


app.py
login.html
(has login & register)
login unsuccessful
login/register successful
db.py
stores login data
users.db
home.html
create new story
search.html
search for preexisting story
edit.html
stories.html
edit existing story
db.py
stores edit data
edits.db
stories.html
returns to page editted.