STORYTELLER

By Amy Xu, Daniel Chiu, and Brian Yang

Guidelines

- Include:
  - component map **[FINISHED]**
  - sitemap **[FINISHED]**
  - database schema **[FINISHED]**
  - any necessary supporting documentation**[FINISHED].**
- Divide the tasks among your group members. Include a "Project Manager" **[FINISHED]**
  - Project manager should also have coding tasks, but will also make sure the group is consistently moving together.
  - Project manager should make certain the design document is coherent and that the group is adhering to the agreed design.
  - If changes need to be made to the design, project manager should be informed.
  - There will be a summary document at the end of the project created by the group, but the project manager will have certain duties pertaining to that document as well.

Components

User accounts

Users will register an account on the page with a unique username. Once their username and password are registered, the data will be encrypted and stored in the users database. Using the credentials they can login to their account and create a session. To close the session, the user has to logout. Once a user logs in, they will be able to create new stories, add lines to existing stories that they haven't already, and read the stories they have already contributed to.

Homepage

The homepage for logged in users will display a list of stories that the user has contributed to. Each row in the stories table will contain information about the story: the user who initially wrote the story (the creator), the date it was created, the title of the story, and the most recent update to the story. There will also be an option in the top corners of the page to create new stories ("Create" button) and add to existing stories ("Browse" button).

New Story

Users can submit a few sentences starting a story as a new "post." The story's title, date of creation, and initial lines will be recorded in a database called "stories." In addition, each story will be given an ID. Users can select the stories that they can contribute to from the browsing page.

Browse

Users can browse a list of randomly selected stories from the stories table that they have not yet contributed to. The date the story was created, the title, the creator, and the most recent update will be shown for each story. The user can select a story to contribute to it.

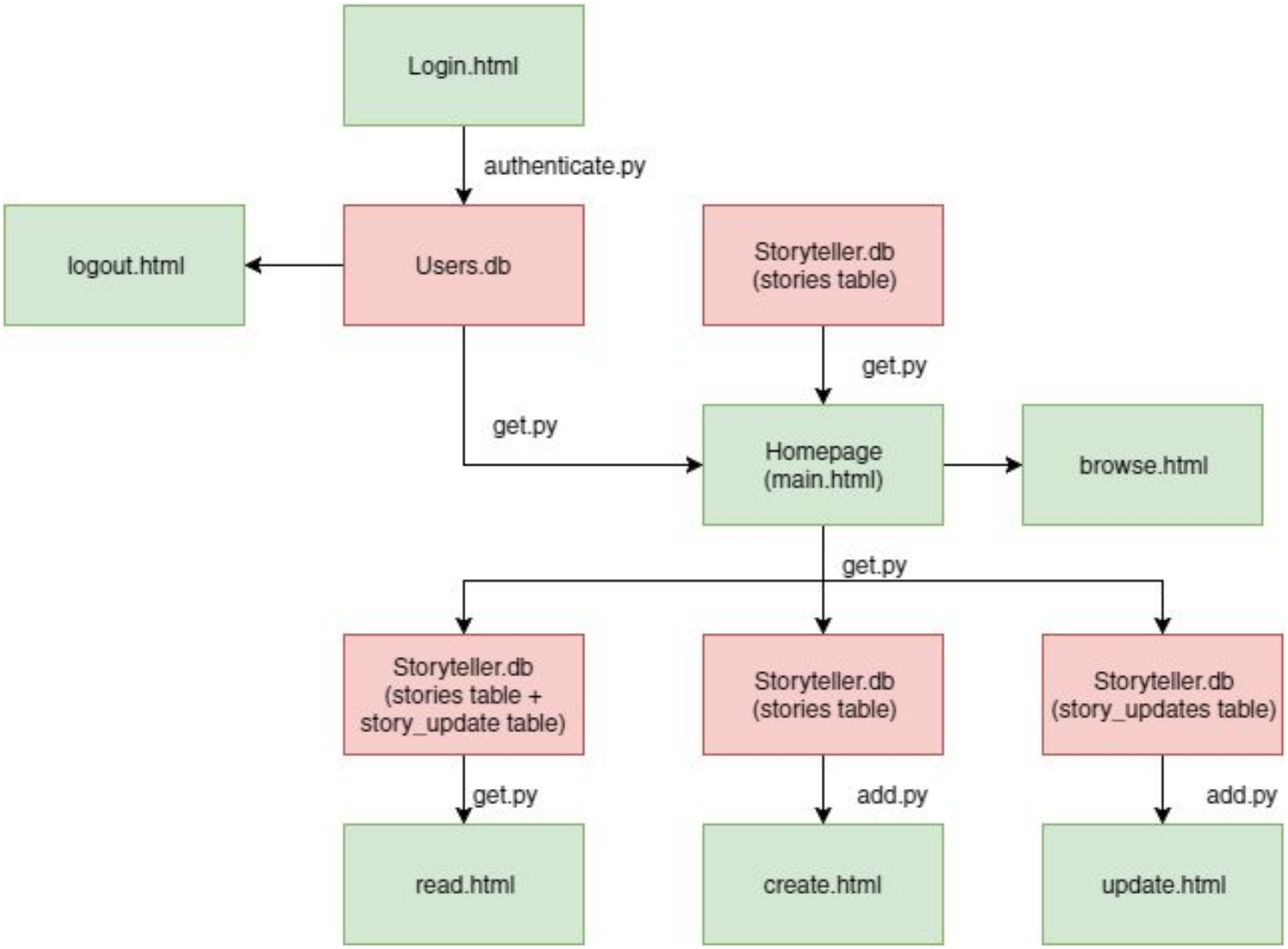
Add line

Once the user is on the page of the story, they can see the latest update to the story. Then, they can add a line to the story below the latest update. The input will be limited to a 150 character count. When the user submits the line, it will be recorded in the story_updates table. The table will record the line the user submitted, the ID of the story they are contributing to, the username of the contributor, and the time/ date of the contribution. The story id will also be added to the user_profiles table so that the user can see the story on their homepage.

Read story

On the homepage, if the user chooses to read a story that the user has contributed to, they can click on the story entry, and then they will be brought to a separate page, read.html, to read it.

Component Map



File structure

app.py

app.py contains the Flask code necessary to get the website up and running as well as manage sessions and send data back and forth between the frontend and the backend

utils/

add.py

Add.py will process all the data we get through POST or GET from the website. It is responsible for adding new stories to the stories table in storyteller.db, and adding updates to story_updates in storyteller.db.

get.py

Get.py is responsible for processing data from the database files and getting them to the website. It pulls the information about the stories (title, date, etc) from the storyteller.db and the updates from the story_updates table. The data obtained from the databases is processed and put into html templates. For example, once a user logs in, they will be redirected to a page displaying all their contributed stories. Get.py is responsible for getting those stories they have contributed to and then putting them into the display template.

authenticate.py

Authenticate is responsible for three main things: adding new users to the users database (checking that the username isn't taken), authenticating the login information, and dealing with sessions. The session is over once the user logs out.

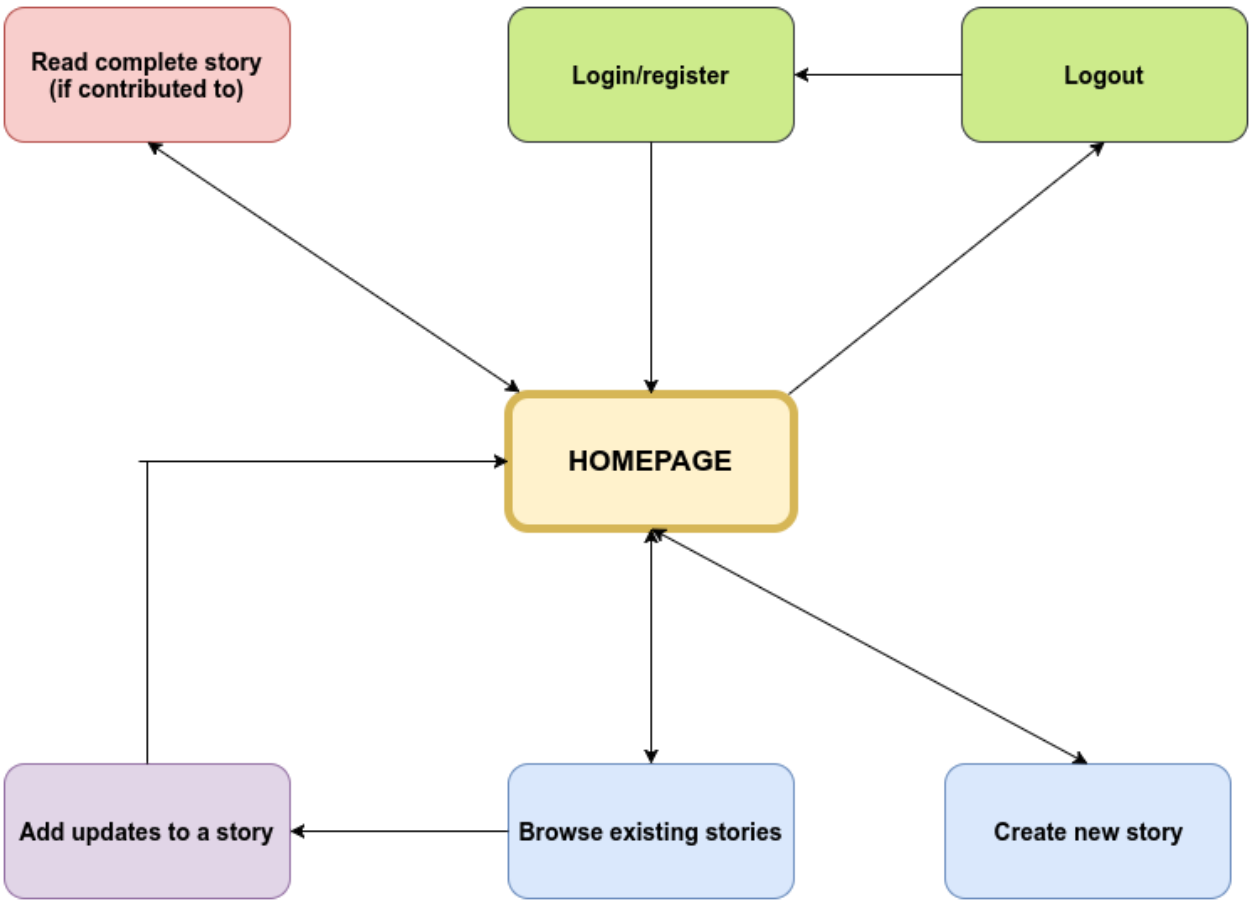
templates/

login.html: the login page users use to log into their accounts
logout.html: the logout page logs the user out of their accounts
main.html: the homepage of the user once he/she logs in
read.html: displays the story up to it's latest update
browse.html: lists the stories the user has never contributed to before
create.html: allows a user to start a new story. Contains html form.
update.html: the user can add a line to an existing story

data/

Storyteller.db: contains information about thestories and stores data about users such as credentials and list of stories that they have contributed to

Sitemap


Read complete story
(if contributed to)
Login/register
Logout
HOMEPAGE
Add updates to a story
Browse existing stories
Create new story

Database Schema

storyteller.db

users table

- Will contain the usernames and salted passwords of users as well as the ids of the stories that they have contributed to (the list of ids will be a string)
- The salt will be in a separate plain text file (there'll be an example salt file on GitHub, but for actual deployment, the salt will be different and not added to Github)

user	(salted) password	story_ids (string)

stories table (contains all the story information)

- Will contain basic information for each story
- Each entry will contain the id of the story, the time the story was initially created, the title, the username of the user who initially created the story (AKA the creator), and the base content of the story

story_id	date	title	creator	base_content

story_updates table

- Will contain the updates users make for each story
- Each entry will contain the id of the story, the time the update was made, the user who made the update, and the update itself

story_id	date	user	update

Task Breakdown

- **Daniel (front and back end)**
  - Make templates for the web pages
  - Ensures the signup and login are secure.
  - Transfers information securely using POST
  - Codes the authentication process
  - Sessions
- **Brian (back end)**
  - Store the credentials of a user who has registered in users.db
  - Check the credentials of a user trying to log in
  - Store the newly created stories in storyteller.db
  - Get random stories from storyteller.db that the user can contribute to and that the user has not already contributed to
  - Store updates to stories in storyteller.db
  - Get data such as the title of a story from storyteller.db
  - Combine the base content of a story from the stories table with user-contributed content/updates to the story from the story_updates table in storyteller.db
- **Amy (front end and project manager)**
  - Design the look of the web pages
  - Create forms that allow the user to register/log in and contribute/create stories
  - Ensuring all the members are on task
  - Resolving member conflicts