Team Spicy Churros William Soe, Max Zlotskiy, Jenny Gao, Michael Cheng

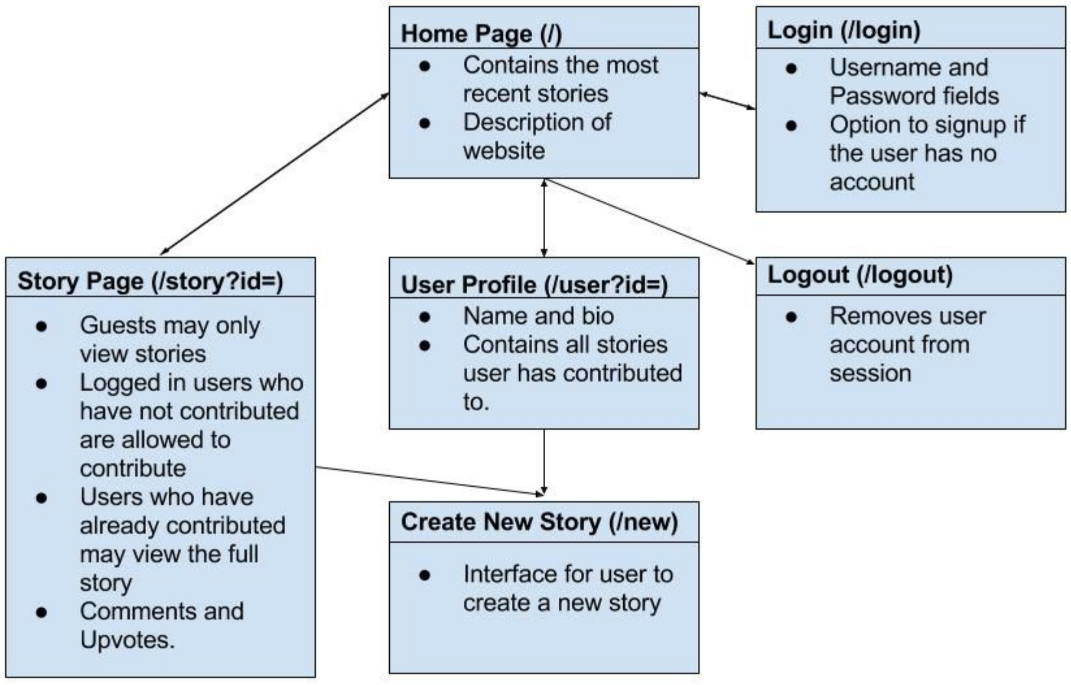
Project 0: Our Story Period 9 Software Development

Components:

- Home Page (`/`)
  - A description of this website and what it does.
  - A login form and a signup form
- Help page that lets people know how to use the site (`/help`)
- About the User Page (`/user?id=`)
  - A list of stories the user has created, as well as their full name and bio
  - Users who log in through the home page will be redirected to their own page
- List of Unfinished Stories (`/stories/unfinished`)
  - A list of links to story pages
  - Can be filtered using query strings. Adding `?genre=` will filter by genre. Adding `?sort=a` will list them alphabetically, or to sort by popularity add `?sort=p`, or to sort by most views add `?sort=v`, or `?sort=d` for recent
- List of Finished Stories (`/stories/finished`) [See above for query data]
- List of All Stories (`/stories`) [See above for query data]
- Story Page (`/story?id=`)
  - For guests(users who have not logged in) the story is set to *viewable* if the story is finished (*viewable* has to do with html). If the story is not finished, they will only see the title.
  - For users logged in who have not yet contributed to the story, only the introduction and the part added by the last contributor is shown
  - For users who have logged in and have contributed to the story, the whole story is displayed, as well as the option to create a spinoff/branch the story
  - Will have a button to like/upvote the story. Each user can only do this once per story, and can rescind their decision at any time by clicking it again.
  - The creator of the story can always see the whole story and will have access to a button that closes the story for additions, marking it finished.
- Logout Page (`/logout`)
  - Deletes session and brings you to the home page.
- The database (`/ourDB.db`)
  - See the schema for which tables this will contain
- New Story Page (/create)

    ○ Prompted for a title, genre, introduction, word limit and cooldown (see schema for details)

Site Map:



Home Page (/)
Contains the most recent stories
Description of website
Login (/login)
Username and Password fields
Option to signup if the user has no account
Story Page (/story?id=)
Guests may only view stories
Logged in users who have not contributed are allowed to contribute
Users who have already contributed may view the full story
Comments and Upvotes.
User Profile (/user?id=)
Name and bio
Contains all stories user has contributed to.
Logout (/logout)
Removes user account from session
Create New Story (/new)
Interface for user to create a new story

Database Schema:

Table of all users. The name of this table is *users*. Columns:

- **username:** TEXT Alphanumeric string that serves as the identifier for individual users. Can be seen publicly through the user's profile or through their finished stories.
- **password:** TEXT Alphanumeric and special character string that serves as the key to a user's account. Must have a minimum of 6 characters. Would be encrypted.
- **full_name**: TEXT Alphabetic string that contains the user's full name. Can be left blank.

- **likes:** TEXT a space-separated list of numerical story IDs. This is a list of every story the user has liked/upvoted. This will prevent the user from liking the same story more than once.

Table of all the stories. The name of this table is *AllStories*. Columns:

- **title:** TEXT, name of the story, given by the creator.
- **creator:** TEXT, username of the creator of the story.
- **Id:** INTEGER PRIMARY KEY, every story is given a unique, numerical id number that can be used to easily access the story on the back-end.
- **genre:** TEXT the genre of the story, as specified by its creator.
- **word_limit:** INTEGER the word limit for each contribution made. A positive value indicates that each contribution must be of that length, while a negative value indicates that contributions can be less than but at most the absolute value of that length.
- **cooldown:** INTEGER the number of contributions that have to be made in between consecutive contributions by the same user. Ex: if cooldown is 10, and user A already contributed, ten other users must add to the story before A can contribute again. Can be negative so that each user can only contribute once.
- **contributions:** INTEGER the number of contributions made to this story. It should be equal to the length of the table that contains the list of contributions for this story.
- **finished:** INTEGER either 1 if the story is finished or 0 if additions can still be made.
- **likes:** INTEGER the number of likes this story has received.
- **views:** INTEGER the number of times the story has been viewed.

For each story, there will be a table containing every contribution made to it. Every contribution will be a record. The name of the table should be *story_id* where “id” is the numerical id of that story, eg *story_42*. Columns:

- **version_num**: INTEGER PRIMARY KEY is the index of that contribution in the list that comprises the story. So the record with a version of 0 would go first, since it is the beginning of the story; then one with version=1, which would be the next addition to the story; etc.
- **contributor:** TEXT the username of the person who was responsible for this addition.
- **text_contributed:** TEXT the text that a contributor writes
- **when:** TEXT the date and time when this contribution was made. Will use the standard python datetime formatting.

Features

Yellow highlighted parts = necessary everything else will be considered at the end

- Choose number of words every contributor gets to add
- Creator can end the story whenever they want to, or set a word limit, or contributors can ask the creator to end the story there
- Option to filter profanity
- Infinite intro length for new stories
- Report misbehaving users
- Branch story, include link to the branch in the original story
- Option to let contributors see the intro as well as the last addition
- Guests can view finished stories, must be logged in to do anything else
- Every story must have genres, almost like tags
- Finished stories have popularity (upvote/downvote). You can list stories by popularity
- Add comments to finished stories
- Users vote on Story of the Week/Day
- List of past stories of the week/day
- Users should be able to see which stories they contributed to
- Public user profiles. Should include a bio and their stories
- Stories can either be One-And-Done or let a user contribute again after a certain number of others have taken turns adding
- Help page and a Code of Conduct
- The creator of a story can specify a custom background color for it
- Add option to set limit on number of additions

Roles

William - Writing the databases and writing methods to facilitate reading and writing to and from the database. Ensuring that
Max - Project Manager. Will be coordinating deadlines and coding conventions, linking database functionality with template rendering. Doing some css.
Jenny - organizing display of stories (genre, popularity), designing user profiles, and making HTML forms related to story creation.
Michael - Flask app developing - Use templates and other python modules to create a flask app. Organize routing and sessions. Implement message flashing to give user feedback, such as warning a user about word count or incorrect login credentials.