

Team Green Monkeys; Daniel He, Faiyaz Rafee
SoftDev
P08-SCENARIO ONE
2022-11-2
time spent: 2 hrs

Target ship date: {2022-11-15}

Components of SCENARIO ONE:

_Users and login/logout functionality using SQLite

_Sign Up and Login pages:

- Usage of cookies to keep users logged in unless they log out.

_Displaying stories, adding on/ starting stories with the following features:

- Full stories can only be read after contributions
- Only last contribution displayed when adding on to a story
- Categorized into a predetermined set of genres (selected from a predetermined dropdown list using HTML select tag), alphabetical, etc
- Word limit when adding on, not applicable when creating a story
- Only contribute to a story once
- Must be logged in to access stories
- Title, body, authors displayed, date and time

_Homepage: body displays all stories the user has contributed to (display in a card form, clickable to view in its entirety)

_Explore Page: Displays all the stories the user hasn't contributed to. Shows the last contribution as well as author, genre, and date. Same card format as homepage stories. Users can click on a card to contribute to it.

_Contribute Page: Allows users to add body text with a limit of 1000 characters. Submitting it changes the latest contribution shown and also removes it from explore page and into the homepage. Also now allows users to read the entire story.

_CreateStory Page: Allows the user to create a story. The body has no word limit. Users must choose from the list of genres.

Software COMPONENTS to achieve:

_Flask web server serving different routes/ webpages

_Databases storing all stories and affiliated resources, users

_Python handling:

- Editing SQLite databases, (updating existing story databases)

_Links acting as buttons for navigation and activating methods

- Html and CSS for UI

_Forms for story writing (error handling for unfilled forms)

_Templates using jinja to insert dynamic, changing data into stagnant html, e.g. user contributed

stories, available stories, etc

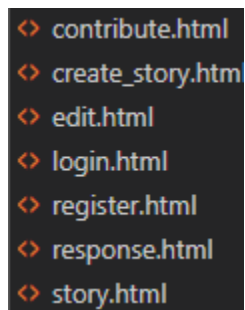
FLASK; work allotment: BOTH [flask], DANIEL [database], FAIYAZ [CSS]

App.py

Rudimentary **ROUTES** logics

- Homepage (Faiyaz):
 - Accesses argument: username
 - Access table Users to specify selection in Contributed_Stories table
 - Display all contributed stories in HomePage Template
- Stories (Faiyaz):
 - Accesses argument: genres, alphabetical, or none—to specify returned data
 - Accesses database table of all stories' title and table Stories, but making sure to avoid titles already in Contributed_Stories table
 - Display data in Stories Template
- Adder_logic (adding logic) [Daniel]:
 - Accesses arguments: username, story title, date and time
 - Updates corresponding Stories table
- Creator_logic (creating logic) [Daniel]:
 - Accesses arguments: username, date, genre, title of story, story body
 - Appends data to corresponding table: ContributedStories, Stories, Table of all story titles
- Login
 - Accesses arguments: username, password and somehow sends that to the rest of the python
- Signup
 - Accesses arguments: username, password and appends data to Users table

Templates:



```
<> contribute.html
<> create_story.htm
<> edit.html
<> login.html
<> register.html
<> response.html
<> story.html
```

__init__.py

db.py

world.py

DATABASE Tables:

“Users” (2 columns):

- Username TEXT P.K. | password TEXT not null

“<Story Title>”, each story is linked to a unique table (5 columns):

- Story Title TEXT, Author username TEXT P.K , date TEXT not null, body of TEXT not null, genre TEXT not null

RELATIONSHIPPING:

__init__.py will handle all the logic of navigation, as well as the logic of actions, such as editing or updating forms.

_Updating or editing forms will call upon methods of db.py

_When navigated to the correct routes, articles will be displayed by fetching data in the database (world.db) through methods in db.py which is called upon in corresponding app.routes in

__init__.py

- Each circle is a new webpage
- Arrows represent navigation pathways

