Chalice: Anastasia Lee, Andy Shyklo, Jack Blair, Jady Lei
SoftDev
P00
2024-10-29
TARGET SHIP DATE: 2024-11-07

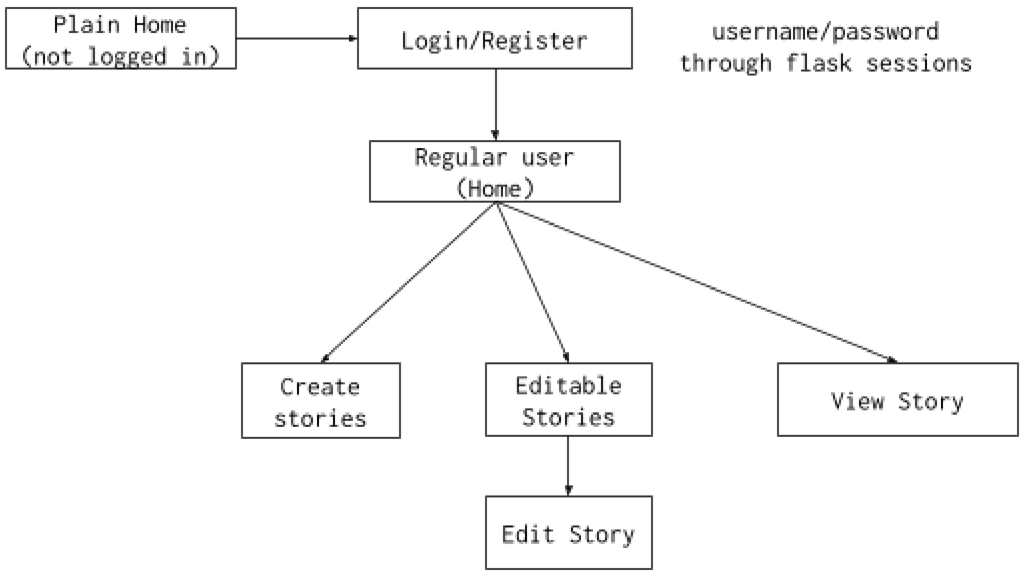
Scenario One: Your team has been contracted to create a collaborative storytelling game/website, with the following features:

- 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.
        - Limit roughly to a sentence (can be edited by the creator).
- 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).

Program Components:

- Login for registered users, register for new users, logout once logged in
    - Can only view the most recent edit while editing
    - Can only view a whole story after they edit it
- Stories:
    - List of story titles and IDs on homepage
    - Ability to add stories
    - Ability to edit stories
    - Ability to view stories

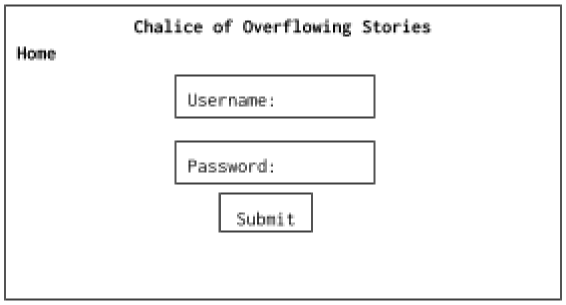
Component Map:



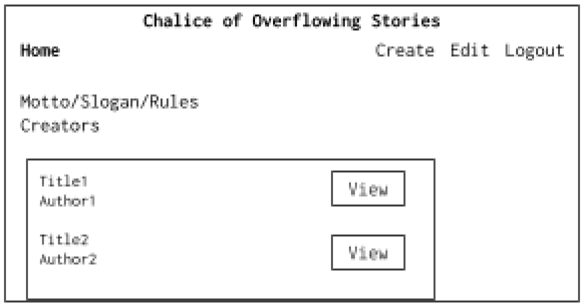
Front-end (HTML, CSS, Displaying info):

- **Home page**
  - Site logo (links to home page) in top left corner
  - NavBar: links to home page and login/register
  - Once logged in:
    - Displays logout button in top right corner
    - NavBar expands to include links to view stories, edit stories, create stories
    - On main screen, displays list of contributions (most recent at the top)
      - These stories are fully viewable to the user upon selection, since they have previously edited them
- **Login/Register (same from front-end perspective)**
  - Username, password, and submit button
- **Edit stories**
  - Last update, textbox, and submit button
- **Create stories**
  - Title, start of story, and submit button

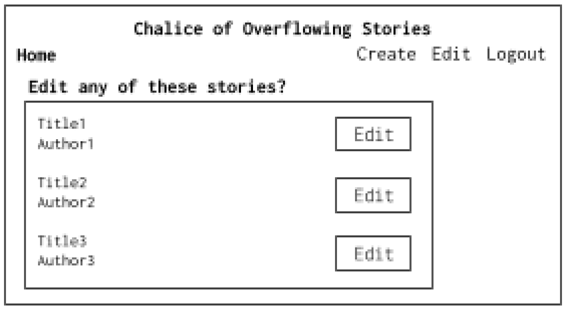
Login/Register



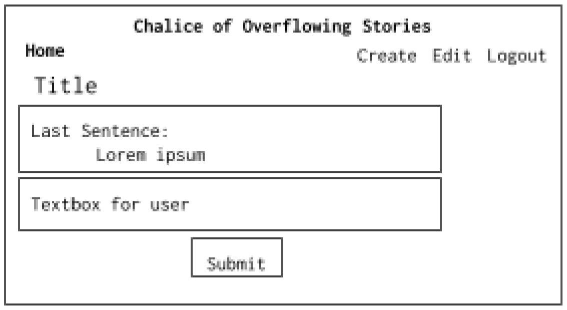
Home



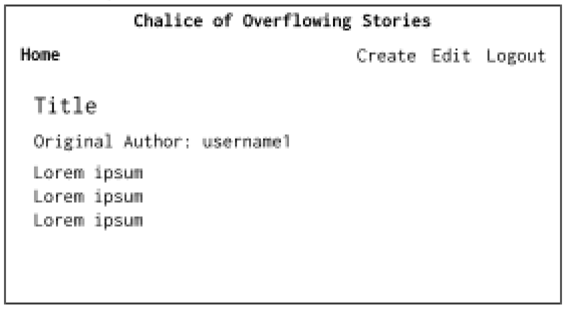
Editable Catalog



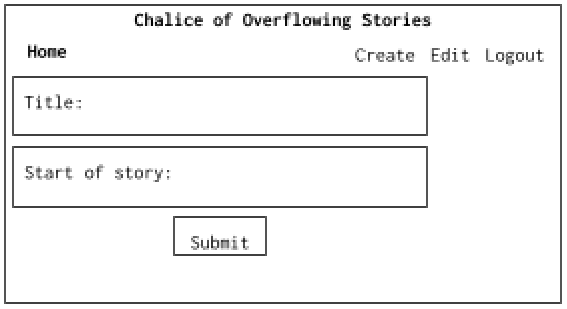
Edit Story



View Story



Create Story



Back-end (Flask Nav/Requests):

- Login information stored or retrieved via flask session request using POST. If the account doesn't exist, run code to ask if you want to make an account, otherwise retrieve login info from SQL database.
- Specific user data stored with SQL with user ID compared to posts/views, and depending on this status, some stories may be marked as locked (not contributed) or unlocked (contributed).
- Posts have version history, creation date, contributors, and other parameters that will be held in an existing, separate SQL database.
- All redirects to specific parts of the website, and their subpages, are routed via flask render_templates and specific website elements.
- Stories should be updated both locally and globally around the website as it is being changed, this could be done with update requests (needs more research).

Database Organization:

- What NEEDS to be stored? (EVERYTHING LISTED IS A TABLE):
    - login information,
        - Username
        - Password
        - Viewable stories
        - Editable stories
    - stories,
        - story_name , story_id(key), chapter_count
    - Chapters
        - Story_id, chapter_id, content, author, date
- Actions:
    - Account management:
        - User logs in
            - Check username in login table and authenticate user.
        - User registers new account
            - Adds new username and password to login table, session logged in.
    - User adds to a story
        - Select most recent update to display
        - Take request and enter it into the story
        - Store contribution metadata
        - Add story to user's viewable
    - User creates a story
        - Create entry in stories with title
        - Store contribution + its metadata
        - Add story to user's viewable
        - Adds story to all users' editable
    - User views a story
        - Select all story contents and display to viewer

Username	Password	Access	Viewable	Editable
admin	******	Admin	Full list	Full list
user_name	******	Normal		

Story ID	Story name
1	abc

Story ID	Story name
2	def

User Database

Story Database

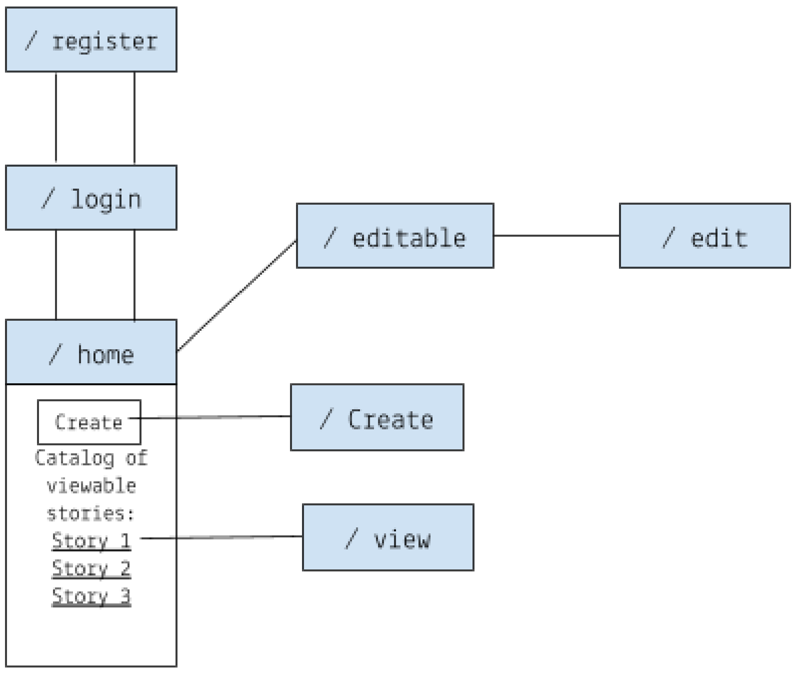
stories

Story ID	Story name
1	abc
2	def

chapters

Story ID	Content	Contributor	Date
1	"abc"	username1	10/24/2024 12:52 A.M.
2	"d"	username2	10/24/2024 12:52 A.M.
2	"e"	username3	10/24/2024 12:53 A.M.
2	"f"	username1	10/31/2024 10:41 A.M.

Site Map:



- Users can always reach the home page
- Once logged in, users can always log out

- All edit and view pages are rendered by a template, with arguments from the databases

Task Assignments:

Anastasia: PM + Python + HTML/CSS
- Deal with site flow (__init__.py)
- Account management (Flask sessions, login, register, logout)
- Create HTML templates for most pages
- Create CSS styling

Andy: User SQL Database + HTML
- Handle user tables
- Handle permissions
- Create HTML templates for viewing and editing stories

Jack: ?
- Was given task assignments but did not deliver on any of them

Jady: Story SQL Database
- Handle the story tables
- Make sure app requests are processed correctly into the database
- Debug if database goes wrong
- Whip design doc back into shape